

Automating Analysis of Spatial Grids Reference Manual

Generated by Doxygen 1.4.7

Fri Jan 27 09:58:25 2012

Contents

1 Automating Analysis of Spatial Grids Namespace Index	1
1.1 Automating Analysis of Spatial Grids Package List	1
2 Automating Analysis of Spatial Grids Hierarchical Index	3
2.1 Automating Analysis of Spatial Grids Class Hierarchy	3
3 Automating Analysis of Spatial Grids Class Index	7
3.1 Automating Analysis of Spatial Grids Class List	7
4 Automating Analysis of Spatial Grids File Index	13
4.1 Automating Analysis of Spatial Grids File List	13
5 Automating Analysis of Spatial Grids Namespace Documentation	17
5.1 Package edu	17
5.2 Package edu.ou	18
5.3 Package edu.ou.asgbook	19
5.4 Package edu.ou.asgbook.core	21
5.5 Package edu.ou.asgbook.datamining	22
5.6 Package edu.ou.asgbook.dataset	23
5.7 Package edu.ou.asgbook.distance	24
5.8 Package edu.ou.asgbook.filters	25
5.9 Package edu.ou.asgbook.geocode	27
5.10 Package edu.ou.asgbook.gmm	28
5.11 Package edu.ou.asgbook.histogram	29

5.12	Package edu.ou.asgbook.imgstat	30
5.13	Package edu.ou.asgbook.io	31
5.14	Package edu.ou.asgbook.linearity	32
5.15	Package edu.ou.asgbook.motion	33
5.16	Package edu.ou.asgbook.oban	35
5.17	Package edu.ou.asgbook.projections	36
5.18	Package edu.ou.asgbook.rasterization	37
5.19	Package edu.ou.asgbook.rbf	38
5.20	Package edu.ou.asgbook.segmentation	39
5.21	Package edu.ou.asgbook.thinning	41
5.22	Package edu.ou.asgbook.transforms	42
5.23	Package edu.ou.asgbook.usage	43
6	Automating Analysis of Spatial Grids Class Documentation	45
6.1	edu.ou.asgbook.motion.AlignAndDifference Class Reference	45
6.2	edu.ou.asgbook.transforms.AlignmentEstimator Class Reference	47
6.3	edu.ou.asgbook.usage.Assignment4 Class Reference	51
6.4	edu.ou.asgbook.rasterization.BoundingBox Class Reference	53
6.5	edu.ou.asgbook.rasterization.CatmullRom Class Reference	55
6.6	edu.ou.asgbook.datamining.CityCategories Class Reference	59
6.7	edu.ou.asgbook.datamining.CityGdiModels Class Reference	61
6.8	edu.ou.asgbook.segmentation.ContiguityEnhancedKMeansSegmenter Class Reference	64
6.9	edu.ou.asgbook.filters.ConvolutionFilter Class Reference	67
6.10	edu.ou.asgbook.dataset.CountryPolygons Class Reference	72
6.11	edu.ou.asgbook.oban.CressmanWeighting Class Reference	76
6.12	edu.ou.asgbook.motion.CrossCorrelation Class Reference	79
6.13	edu.ou.asgbook.histogram.CumulativeDistributionFunction Class Ref- erence	83
6.14	edu.ou.asgbook.dataset.DailyRainfall Class Reference	86
6.15	edu.ou.asgbook.rbf.DataSimulator Class Reference	88
6.16	edu.ou.asgbook.linearity.DataTransform Class Reference	90

6.17 edu.ou.asgbook.motion.Differencer Class Reference	92
6.18 edu.ou.asgbook.filters.DilateErodeFilter Class Reference	94
6.19 edu.ou.asgbook.filters.DilationFilter Class Reference	96
6.20 edu.ou.asgbook.motion.EdgeBased Class Reference	99
6.21 edu.ou.asgbook.projections.Ellipsoid Class Reference	102
6.22 edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter Class Reference	104
6.23 edu.ou.asgbook.histogram.Entropy Class Reference	107
6.24 edu.ou.asgbook.filters.ErodeDilateFilter Class Reference	109
6.25 edu.ou.asgbook.filters.ErosionFilter Class Reference	111
6.26 edu.ou.asgbook.io.EsriGrid Class Reference	114
6.27 edu.ou.asgbook.distance.EuclideanDT Interface Reference	117
6.28 edu.ou.asgbook.distance.EuclideanDTPropagation Class Reference . .	118
6.29 edu.ou.asgbook.distance.EuclideanDTRecursivePropagation Class Reference	120
6.30 edu.ou.asgbook.distance.EuclideanDTSaito Class Reference	122
6.31 edu.ou.asgbook.transforms.FFT Class Reference	124
6.32 edu.ou.asgbook.transforms.FFT2D Class Reference	127
6.33 edu.ou.asgbook.transforms.FFTBandpassFilter Class Reference . . .	131
6.34 edu.ou.asgbook.transforms.FFTConvolutionFilter Class Reference . .	134
6.35 edu.ou.asgbook.datamining.FuzzyCandidateMarket Class Reference .	137
6.36 edu.ou.asgbook.datamining.FuzzyLogic Class Reference	140
6.37 edu.ou.asgbook.datamining.FuzzyLogic.Rule Interface Reference . .	141
6.38 edu.ou.asgbook.gmm.GaussianComponent Class Reference	142
6.39 edu.ou.asgbook.gmm.GaussianMixtureModel Class Reference	147
6.40 edu.ou.asgbook.oban.GaussWeighting Class Reference	150
6.41 edu.ou.asgbook.datamining.GdiPattern Class Reference	152
6.42 edu.ou.asgbook.dataset.GlobalPopulation Class Reference	155
6.43 edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix Class Reference	158
6.44 edu.ou.asgbook.thinning.HilditchSkeletonization Class Reference . .	162
6.45 edu.ou.asgbook.histogram.Histogram Class Reference	165
6.46 edu.ou.asgbook.histogram.HistogramBinSelection Class Reference . .	169

6.47	edu.ou.asgbook.motion.HornSchunk Class Reference	174
6.48	edu.ou.asgbook.transforms.HoughTransform Class Reference	178
6.49	edu.ou.asgbook.motion.HungarianAssigner Class Reference	181
6.50	edu.ou.asgbook.motion.HybridTracker Class Reference	184
6.51	edu.ou.asgbook.segmentation.HysteresisSegmenter Class Reference	188
6.52	edu.ou.asgbook.filters.Inverter Class Reference	191
6.53	edu.ou.asgbook.motion.KalmanFilter Class Reference	193
6.54	edu.ou.asgbook.io.KmlWriter Class Reference	197
6.55	edu.ou.asgbook.segmentation.LabelResult Class Reference	201
6.56	edu.ou.asgbook.projections.LambertConformal2SP Class Reference	203
6.57	edu.ou.asgbook.core.LatLon Class Reference	206
6.58	edu.ou.asgbook.core.LatLonGrid Class Reference	209
6.59	edu.ou.asgbook.core.LevelSet Class Reference	219
6.60	edu.ou.asgbook.rasterization.Line Class Reference	220
6.61	edu.ou.asgbook.linearity.LinearityVerifier Class Reference	224
6.62	edu.ou.asgbook.linearity.LinearScaling Class Reference	227
6.63	edu.ou.asgbook.imgstat.LocalMeasures Class Reference	229
6.64	edu.ou.asgbook.filters.LoGEdgeFilter Class Reference	232
6.65	edu.ou.asgbook.linearity.LogScaling Class Reference	235
6.66	edu.ou.asgbook.dataset.MadisTemperature Class Reference	237
6.67	edu.ou.asgbook.filters.MatchedFilter Class Reference	239
6.68	edu.ou.asgbook.filters.MaxValueFilter Class Reference	243
6.69	edu.ou.asgbook.thinning.MedialAxisSkeletonization Class Reference	245
6.70	edu.ou.asgbook.filters.MedianFilter Class Reference	247
6.71	edu.ou.asgbook.motion.MotionEstimator Interface Reference	250
6.72	edu.ou.asgbook.filters.MultiFilter Class Reference	251
6.73	edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter Class Reference	253
6.74	edu.ou.asgbook.filters.NHighest Class Reference	257
6.75	edu.ou.asgbook.filters.NHighestLevelSetImpl Class Reference	260
6.76	edu.ou.asgbook.dataset.NighttimeLights Class Reference	262

6.77	edu.ou.asgbook.oban.ObjectiveAnalysisUtils Class Reference	264
6.78	edu.ou.asgbook.motion.ObjectTracker Class Reference	266
6.79	edu.ou.asgbook.motion.ObjectTracker.Assigner Interface Reference	270
6.80	edu.ou.asgbook.motion.ObjectTracker.CostEstimator Interface Reference	271
6.81	edu.ou.asgbook.filters.OrientedEllipseFilter Class Reference	272
6.82	edu.ou.asgbook.histogram.OtsuThresholdSelector Class Reference	275
6.83	edu.ou.asgbook.io.OutputDirectory Class Reference	278
6.84	edu.ou.asgbook.core.Pair< X, Y > Class Reference	280
6.85	edu.ou.asgbook.motion.PhaseCorrelation Class Reference	281
6.86	edu.ou.asgbook.core.Pixel Class Reference	285
6.87	edu.ou.asgbook.io.PngWriter Class Reference	289
6.88	edu.ou.asgbook.core.PointObservations Class Reference	294
6.89	edu.ou.asgbook.rasterization.Polygon Class Reference	296
6.90	edu.ou.asgbook.datamining.PrimaryCities Class Reference	300
6.91	edu.ou.asgbook.rbf.ProjectionPursuit Class Reference	302
6.92	edu.ou.asgbook.rbf.ProjectionPursuit.NextRBF Interface Reference	306
6.93	edu.ou.asgbook.motion.PyramidalCrossCorrelation Class Reference	307
6.94	edu.ou.asgbook.imgstat.Quantizer Class Reference	310
6.95	edu.ou.asgbook.filters.QuickSelect Class Reference	314
6.96	edu.ou.asgbook.rbf.RadialBasisFunction Class Reference	316
6.97	edu.ou.asgbook.segmentation.RegionGrowing Class Reference	318
6.98	edu.ou.asgbook.segmentation.RegionProperty Class Reference	319
6.99	edu.ou.asgbook.projections.Remapper Class Reference	325
6.100	edu.ou.asgbook.filters.SaturateFilter Class Reference	327
6.101	edu.ou.asgbook.core.ScalarStatistic Class Reference	329
6.102	edu.ou.asgbook.segmentation.Segmenter Interface Reference	332
6.103	edu.ou.asgbook.filters.SeparableConvolutionFilter Class Reference	334
6.104	edu.ou.asgbook.dataset.SeviriInfraredTemperature Class Reference	338
6.105	edu.ou.asgbook.filters.SimpleThresholder Class Reference	341
6.106	edu.ou.asgbook.segmentation.SnakeActiveContour Class Reference	344

6.107	edu.ou.asgbook.segmentation.SnakeActiveContour.Snake Class Reference	348
6.108	edu.ou.asgbook.filters.SobelEdgeFilter Class Reference	350
6.109	edu.ou.asgbook.filters.SpatialFilter Interface Reference	353
6.110	edu.ou.asgbook.filters.SpeckleFilter Class Reference	355
6.111	edu.ou.asgbook.usage.Sprawl Class Reference	358
6.112	edu.ou.asgbook.imgstat.StructuralMeasures Class Reference	361
6.113	edu.ou.asgbook.dataset.SurfaceAlbedo Class Reference	363
6.114	edu.ou.asgbook.segmentation.ThresholdSegmenter Class Reference	367
6.115	edu.ou.asgbook.geocode.UsaZipcode Class Reference	370
6.116	edu.ou.asgbook.imgstat.VectorQuantizer Class Reference	372
6.117	edu.ou.asgbook.segmentation.WatershedSegmenter Class Reference	377
6.118	edu.ou.asgbook.oban.WeightedAverage Class Reference	380
6.119	edu.ou.asgbook.oban.WeightedAverageOptimized Class Reference	384
6.120	edu.ou.asgbook.oban.WeightFunction Interface Reference	387
6.121	edu.ou.asgbook.dataset.WorldBankGDI Class Reference	389
7	Automating Analysis of Spatial Grids File Documentation	393
7.1	AlignAndDifference.java File Reference	393
7.2	AlignmentEstimator.java File Reference	394
7.3	Assignment4.java File Reference	395
7.4	BoundingBox.java File Reference	396
7.5	CatmullRom.java File Reference	397
7.6	CityCategories.java File Reference	398
7.7	CityGdiModels.java File Reference	399
7.8	ContiguityEnhancedKMeansSegmenter.java File Reference	400
7.9	ConvolutionFilter.java File Reference	401
7.10	CountryPolygons.java File Reference	402
7.11	CressmanWeighting.java File Reference	403
7.12	CrossCorrelation.java File Reference	404
7.13	CumulativeDistributionFunction.java File Reference	405
7.14	DailyRainfall.java File Reference	406

7.15 DataSimulator.java File Reference	407
7.16 DataTransform.java File Reference	408
7.17 Differencer.java File Reference	409
7.18 DilateErodeFilter.java File Reference	410
7.19 DilationFilter.java File Reference	411
7.20 EdgeBased.java File Reference	412
7.21 Ellipsoid.java File Reference	413
7.22 EnhancedWatershedSegmenter.java File Reference	414
7.23 Entropy.java File Reference	415
7.24 ErodeDilateFilter.java File Reference	416
7.25 ErosionFilter.java File Reference	417
7.26 EsriGrid.java File Reference	418
7.27 EuclideanDT.java File Reference	419
7.28 EuclideanDTPropagation.java File Reference	420
7.29 EuclideanDTRecursivePropagation.java File Reference	421
7.30 EuclideanDTSaito.java File Reference	422
7.31 FFT.java File Reference	423
7.32 FFT2D.java File Reference	424
7.33 FFTBandpassFilter.java File Reference	425
7.34 FFTConvolutionFilter.java File Reference	426
7.35 FuzzyCandidateMarket.java File Reference	427
7.36 FuzzyLogic.java File Reference	428
7.37 GaussianComponent.java File Reference	429
7.38 GaussianMixtureModel.java File Reference	430
7.39 GaussWeighting.java File Reference	431
7.40 GdiPattern.java File Reference	432
7.41 GlobalPopulation.java File Reference	433
7.42 GraylevelCooccurrenceMatrix.java File Reference	434
7.43 HilditchSkeletonization.java File Reference	435
7.44 Histogram.java File Reference	436
7.45 HistogramBinSelection.java File Reference	437

7.46	HornSchunk.java File Reference	438
7.47	HoughTransform.java File Reference	439
7.48	HungarianAssigner.java File Reference	440
7.49	HybridTracker.java File Reference	441
7.50	HysteresisSegmenter.java File Reference	442
7.51	Inverter.java File Reference	443
7.52	KalmanFilter.java File Reference	444
7.53	KmlWriter.java File Reference	445
7.54	LabelResult.java File Reference	446
7.55	LambertConformal2SP.java File Reference	447
7.56	LatLon.java File Reference	448
7.57	LatLonGrid.java File Reference	449
7.58	LevelSet.java File Reference	450
7.59	Line.java File Reference	451
7.60	LinearityVerifier.java File Reference	452
7.61	LinearScaling.java File Reference	453
7.62	LocalMeasures.java File Reference	454
7.63	LoGEdgeFilter.java File Reference	455
7.64	LogScaling.java File Reference	456
7.65	MadisTemperature.java File Reference	457
7.66	MatchedFilter.java File Reference	458
7.67	MaxValueFilter.java File Reference	459
7.68	MedialAxisSkeletonization.java File Reference	460
7.69	MedianFilter.java File Reference	461
7.70	MotionEstimator.java File Reference	462
7.71	MultiFilter.java File Reference	463
7.72	MultiscaleKMeansSegmenter.java File Reference	464
7.73	NHighest.java File Reference	465
7.74	NHighestLevelSetImpl.java File Reference	466
7.75	NighttimeLights.java File Reference	467
7.76	ObjectiveAnalysisUtils.java File Reference	468

7.77	ObjectTracker.java File Reference	469
7.78	OrientedEllipseFilter.java File Reference	470
7.79	OtsuThresholdSelector.java File Reference	471
7.80	OutputDirectory.java File Reference	472
7.81	package-info.java File Reference	473
7.82	package-info.java File Reference	474
7.83	package-info.java File Reference	475
7.84	package-info.java File Reference	476
7.85	package-info.java File Reference	477
7.86	package-info.java File Reference	478
7.87	package-info.java File Reference	479
7.88	package-info.java File Reference	480
7.89	package-info.java File Reference	481
7.90	package-info.java File Reference	482
7.91	package-info.java File Reference	483
7.92	package-info.java File Reference	484
7.93	package-info.java File Reference	485
7.94	package-info.java File Reference	486
7.95	package-info.java File Reference	487
7.96	package-info.java File Reference	488
7.97	package-info.java File Reference	489
7.98	package-info.java File Reference	490
7.99	package-info.java File Reference	491
7.100	package-info.java File Reference	492
7.101	Pair.java File Reference	493
7.102	PhaseCorrelation.java File Reference	494
7.103	Pixel.java File Reference	495
7.104	PngWriter.java File Reference	496
7.105	PointObservations.java File Reference	497
7.106	Polygon.java File Reference	498
7.107	PrimaryCities.java File Reference	499

7.108ProjectionPursuit.java File Reference	500
7.109PyramidalCrossCorrelation.java File Reference	501
7.110Quantizer.java File Reference	502
7.111QuickSelect.java File Reference	503
7.112RadialBasisFunction.java File Reference	504
7.113RegionGrowing.java File Reference	505
7.114RegionProperty.java File Reference	506
7.115Remapper.java File Reference	507
7.116SaturateFilter.java File Reference	508
7.117ScalarStatistic.java File Reference	509
7.118Segmenter.java File Reference	510
7.119SeparableConvolutionFilter.java File Reference	511
7.120SeviriInfraredTemperature.java File Reference	512
7.121SimpleThresholder.java File Reference	513
7.122SnakeActiveContour.java File Reference	514
7.123SobelEdgeFilter.java File Reference	515
7.124SpatialFilter.java File Reference	516
7.125SpeckleFilter.java File Reference	517
7.126Sprawl.java File Reference	518
7.127StructuralMeasures.java File Reference	519
7.128SurfaceAlbedo.java File Reference	520
7.129ThresholdSegmenter.java File Reference	521
7.130UsaZipcode.java File Reference	522
7.131VectorQuantizer.java File Reference	523
7.132WatershedSegmenter.java File Reference	524
7.133WeightedAverage.java File Reference	525
7.134WeightedAverageOptimized.java File Reference	526
7.135WeightFunction.java File Reference	527
7.136WorldBankGDI.java File Reference	528

Chapter 1

Automating Analysis of Spatial Grids Namespace Index

1.1 Automating Analysis of Spatial Grids Package List

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

edu	17
edu.ou	18
edu.ou.asgbook	19
edu.ou.asgbook.core (Classes that form the basis of spatial analysis techniques discussed in the book)	21
edu.ou.asgbook.datamining (Classes discussed in Chapter 8 (Data Mining))	22
edu.ou.asgbook.dataset (Classes that provide the ability to read the various datasets that are used to illustrate spatial analysis techniques)	23
edu.ou.asgbook.distance (Distance Transform methods (See Chapter 4))	24
edu.ou.asgbook.filters (Most of these filters are neighborhood/window based and are discussed in Chapter 5)	25
edu.ou.asgbook.geocode (Geocoding is discussed in Chapter 2)	27
edu.ou.asgbook.gmm (A parametric approximation to an image (See Chapter 3))	28
edu.ou.asgbook.histogram (Histograms and histogram-based techniques are discussed in Chapter 4)	29
edu.ou.asgbook.imgstat (Texture, vector quantization and other image statistics techniques discussed in Chapter 4)	30
edu.ou.asgbook.io (Helper classes to read ESRI grids and write out KML/PNG)	31
edu.ou.asgbook.linearity (Linearity verification and data transformation are discussed in Chapter 2)	32

edu.ou.asgbook.motion (Change and motion estimation techniques discussed in Chapter 7)	33
edu.ou.asgbook.oban (Techniques to interpolate point observations to a spatial grid discussed in Chapter 3)	35
edu.ou.asgbook.projections (Map projection techniques discussed in Chapter 2)	36
edu.ou.asgbook.rasterization (Rasterization techniques discussed in Chapter 2)	37
edu.ou.asgbook.rbf (Parametric approximations discussed in Chapter 3) . . .	38
edu.ou.asgbook.segmentation (Object identification techniques discussed in Chapter 6)	39
edu.ou.asgbook.thinning (Skeletonization techniques discussed in Chapter 5)	41
edu.ou.asgbook.transforms (Fourier Transforms are discussed in Chapter 5) .	42
edu.ou.asgbook.usage (A classroom assignment; an example algorithm) . . .	43

Chapter 2

Automating Analysis of Spatial Grids Hierarchical Index

2.1 Automating Analysis of Spatial Grids Class Hierar- chy

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

edu.ou.asgbook.motion.AlignAndDifference	45
edu.ou.asgbook.transforms.AlignmentEstimator	47
edu.ou.asgbook.usage.Assignment4	51
edu.ou.asgbook.rasterization.BoundingBox	53
edu.ou.asgbook.rasterization.CatmullRom	55
edu.ou.asgbook.datamining.CityCategories	59
edu.ou.asgbook.datamining.CityGdiModels	61
edu.ou.asgbook.segmentation.ContiguityEnhancedKMeansSegmenter	64
edu.ou.asgbook.dataset.CountryPolygons	72
edu.ou.asgbook.histogram.CumulativeDistributionFunction	83
edu.ou.asgbook.dataset.DailyRainfall	86
edu.ou.asgbook.rbf.DataSimulator	88
edu.ou.asgbook.linearity.DataTransform	90
edu.ou.asgbook.linearity.LinearScaling	227
edu.ou.asgbook.linearity.LogScaling	235
edu.ou.asgbook.motion.Differencer	92
edu.ou.asgbook.projections.Ellipsoid	102
edu.ou.asgbook.histogram.Entropy	107
edu.ou.asgbook.io.EsriGrid	114
edu.ou.asgbook.distance.EuclideanDT	117
edu.ou.asgbook.distance.EuclideanDTPropagation	118

edu.ou.asgbook.distance.EuclideanDTRRecursivePropagation	120
edu.ou.asgbook.distance.EuclideanDTSaito	122
edu.ou.asgbook.transforms.FFT	124
edu.ou.asgbook.transforms.FFT2D	127
edu.ou.asgbook.transforms.FFTBandpassFilter	131
edu.ou.asgbook.transforms.FFTConvolutionFilter	134
edu.ou.asgbook.datamining.FuzzyCandidateMarket	137
edu.ou.asgbook.datamining.FuzzyLogic	140
edu.ou.asgbook.datamining.FuzzyLogic.Rule	141
edu.ou.asgbook.gmm.GaussianComponent	142
edu.ou.asgbook.gmm.GaussianMixtureModel	147
edu.ou.asgbook.datamining.GdiPattern	152
edu.ou.asgbook.dataset.GlobalPopulation	155
edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix	158
edu.ou.asgbook.thinning.HilditchSkeletonization	162
edu.ou.asgbook.histogram.Histogram	165
edu.ou.asgbook.histogram.HistogramBinSelection	169
edu.ou.asgbook.transforms.HoughTransform	178
edu.ou.asgbook.motion.KalmanFilter	193
edu.ou.asgbook.io.KmlWriter	197
edu.ou.asgbook.segmentation.LabelResult	201
edu.ou.asgbook.projections.LambertConformal2SP	203
edu.ou.asgbook.core.LatLon	206
edu.ou.asgbook.core.LatLonGrid	209
edu.ou.asgbook.core.LevelSet	219
edu.ou.asgbook.rasterization.Line	220
edu.ou.asgbook.linearity.LinearityVerifier	224
edu.ou.asgbook.imgstat.LocalMeasures	229
edu.ou.asgbook.dataset.MadisTemperature	237
edu.ou.asgbook.filters.MaxValueFilter	243
edu.ou.asgbook.thinning.MedialAxisSkeletonization	245
edu.ou.asgbook.motion.MotionEstimator	250
edu.ou.asgbook.motion.CrossCorrelation	79
edu.ou.asgbook.motion.EdgeBased	99
edu.ou.asgbook.motion.HornSchunk	174
edu.ou.asgbook.motion.HybridTracker	184
edu.ou.asgbook.motion.ObjectTracker	266
edu.ou.asgbook.motion.PyramidalCrossCorrelation	307
edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter	253
edu.ou.asgbook.filters.NHighest	257
edu.ou.asgbook.filters.NHighestLevelSetImpl	260
edu.ou.asgbook.dataset.NighttimeLights	262
edu.ou.asgbook.oban.ObjectiveAnalysisUtils	264
edu.ou.asgbook.motion.ObjectTracker.Assigner	270
edu.ou.asgbook.motion.HungarianAssigner	181

edu.ou.asgbook.motion.ObjectTracker.CostEstimator	271
edu.ou.asgbook.histogram.OtsuThresholdSelector	275
edu.ou.asgbook.io.OutputDirectory	278
edu.ou.asgbook.core.Pair< X, Y >	280
edu.ou.asgbook.motion.PhaseCorrelation	281
edu.ou.asgbook.core.Pixel	285
edu.ou.asgbook.io.PngWriter	289
edu.ou.asgbook.core.PointObservations	294
edu.ou.asgbook.rasterization.Polygon	296
edu.ou.asgbook.datamining.PrimaryCities	300
edu.ou.asgbook.rbf.ProjectionPursuit	302
edu.ou.asgbook.rbf.ProjectionPursuit.NextRBF	306
edu.ou.asgbook.imgstat.Quantizer	310
edu.ou.asgbook.filters.QuickSelect	314
edu.ou.asgbook.rbf.RadialBasisFunction	316
edu.ou.asgbook.segmentation.RegionGrowing	318
edu.ou.asgbook.segmentation.RegionProperty	319
edu.ou.asgbook.projections.Remapper	325
edu.ou.asgbook.core.ScalarStatistic	329
edu.ou.asgbook.segmentation.Segmenter	332
edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter	104
edu.ou.asgbook.segmentation.HysteresisSegmenter	188
edu.ou.asgbook.segmentation.ThresholdSegmenter	367
edu.ou.asgbook.segmentation.WatershedSegmenter	377
edu.ou.asgbook.dataset.SeviriInfraredTemperature	338
edu.ou.asgbook.segmentation.SnakeActiveContour	344
edu.ou.asgbook.segmentation.SnakeActiveContour.Snake	348
edu.ou.asgbook.filters.SpatialFilter	353
edu.ou.asgbook.filters.ConvolutionFilter	67
edu.ou.asgbook.filters.DilationFilter	96
edu.ou.asgbook.filters.ErosionFilter	111
edu.ou.asgbook.filters.Inverter	191
edu.ou.asgbook.filters.LoGEdgeFilter	232
edu.ou.asgbook.filters.MatchedFilter	239
edu.ou.asgbook.filters.MedianFilter	247
edu.ou.asgbook.filters.MultiFilter	251
edu.ou.asgbook.filters.DilateErodeFilter	94
edu.ou.asgbook.filters.ErodeDilateFilter	109
edu.ou.asgbook.filters.OrientedEllipseFilter	272
edu.ou.asgbook.filters.SaturateFilter	327
edu.ou.asgbook.filters.SeparableConvolutionFilter	334
edu.ou.asgbook.filters.SimpleThresholder	341
edu.ou.asgbook.filters.SobelEdgeFilter	350
edu.ou.asgbook.filters.SpeckleFilter	355
edu.ou.asgbook.transforms.FFTBandpassFilter	131

edu.ou.asgbook.transforms.FFTConvolutionFilter	134
edu.ou.asgbook.usage.Sprawl	358
edu.ou.asgbook.imgstat.StructuralMeasures	361
edu.ou.asgbook.dataset.SurfaceAlbedo	363
edu.ou.asgbook.geocode.UsaZipcode	370
edu.ou.asgbook.imgstat.VectorQuantizer	372
edu.ou.asgbook.oban.WeightedAverage	380
edu.ou.asgbook.oban.WeightedAverageOptimized	384
edu.ou.asgbook.oban.WeightFunction	387
edu.ou.asgbook.oban.CressmanWeighting	76
edu.ou.asgbook.oban.GaussWeighting	150
edu.ou.asgbook.dataset.WorldBankGDI	389

Chapter 3

Automating Analysis of Spatial Grids Class Index

3.1 Automating Analysis of Spatial Grids Class List

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

edu.ou.asgbook.motion.AlignAndDifference (Aligns two grids and then computes their difference)	45
edu.ou.asgbook.transforms.AlignmentEstimator (Estimate the degree of spatial displacement between two similar grids)	47
edu.ou.asgbook.usage.Assignment4 ((1) Find optimal threshold on log(pop) Find distance of every grid point to a point < thresh Find optimal threshold of distance values Threshold image to keep only values < threshold)	51
edu.ou.asgbook.rasterization.BoundingBox (A rectangular bounding box of a polygon)	53
edu.ou.asgbook.rasterization.CatmullRom (A Catmull-Rom spline, a local spline)	55
edu.ou.asgbook.datamining.CityCategories (Obtains city data for clustering)	59
edu.ou.asgbook.datamining.CityGdiModels (Applies different data mining models to each city)	61
edu.ou.asgbook.segmentation.ContiguityEnhancedKMeansSegmenter (Objects consist of pixels that are grown from initial centers using K-means)	64
edu.ou.asgbook.filters.ConvolutionFilter (Convolve an image by a window) .	67
edu.ou.asgbook.dataset.CountryPolygons (Reads country-by-country coordinates from a KML placemarks file)	72
edu.ou.asgbook.oban.CressmanWeighting (An interpolation method that uses $1/r^2$)	76

edu.ou.asgbook.motion.CrossCorrelation (Estimates motion using cross-correlation)	79
edu.ou.asgbook.histogram.CumulativeDistributionFunction (Forms a CDF from a Histogram)	83
edu.ou.asgbook.dataset.DailyRainfall (Reads the ASCII precipitation data available at http://madis-data.noaa.gov/public/hydrodumpguest.html)	86
edu.ou.asgbook.rbf.DataSimulator (Simulates RBF data to be fit)	88
edu.ou.asgbook.linearity.DataTransform (Transform pixel values, usually to meet linearity requirements)	90
edu.ou.asgbook.motion.Differencer (Just computes a pixel-by-pixel difference)	92
edu.ou.asgbook.filters.DilateErodeFilter (Carries out paired dilation followed by erosion for filling in holes)	94
edu.ou.asgbook.filters.DilationFilter (Expands entities by taking a local maximum)	96
edu.ou.asgbook.motion.EdgeBased (Estimates motion based on the displacement of edges)	99
edu.ou.asgbook.projections.Ellipsoid (An ellipsoidal approximation to the earth)	102
edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter (Enhanced watershed segmentation following Lakshmanan, Hondl and Rabin)	104
edu.ou.asgbook.histogram.Entropy (Compute entropy from a histogram) .	107
edu.ou.asgbook.filters.ErodeDilateFilter (Carries out paired erosion followed by dilation for denoising)	109
edu.ou.asgbook.filters.ErosionFilter (Reduces the size of entities by taking a local minimum)	111
edu.ou.asgbook.io.EsriGrid (Read an ESRI grid)	114
edu.ou.asgbook.distance.EuclideanDT	117
edu.ou.asgbook.distance.EuclideanDTPropagation (Implementation of Euclidean distance that updates the distance instead of computing it afresh each time)	118
edu.ou.asgbook.distance.EuclideanDTRecursivePropagation (Note that this class is only for illustrative purposes)	120
edu.ou.asgbook.distance.EuclideanDTSaito (The Saito technique of computing the distance transform by calculating in the two directions separately)	122
edu.ou.asgbook.transforms.FFT (FFT based on Sedgewick and Wayne) . .	124
edu.ou.asgbook.transforms.FFT2D (Two-dimensional FFT)	127
edu.ou.asgbook.transforms.FFTBandpassFilter (Removes noise (high frequencies) and the gross signal (low frequencies))	131
edu.ou.asgbook.transforms.FFTConvolutionFilter (An optimization for convolution using FFTs)	134
edu.ou.asgbook.datamining.FuzzyCandidateMarket (Uses heuristic rules to choose the next market to enter)	137
edu.ou.asgbook.datamining.FuzzyLogic (A simple fuzzy logic engine) . . .	140

edu.ou.asgbook.datamining.FuzzyLogic.Rule	141
edu.ou.asgbook.gmm.GaussianComponent (Component of a Gaussian Mixture Model)	142
edu.ou.asgbook.gmm.GaussianMixtureModel (A parametric approximation of a spatial grid as a sum of Gaussians)	147
edu.ou.asgbook.oban.GaussWeighting (An interpolation method that uses $\exp(-1/r^2)$)	150
edu.ou.asgbook.datamining.GdiPattern (The training pattern for each city)	152
edu.ou.asgbook.dataset.GlobalPopulation (Reads the ASCII population data available at http://sedac.ciesin.columbia.edu/gpw)	155
edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix (Computes texture properties from a GLCM)	158
edu.ou.asgbook.thinning.HilditchSkeletonization (Hilditch method of skeletonizing a grid)	162
edu.ou.asgbook.histogram.Histogram (A histogram is an empirical probability distribution)	165
edu.ou.asgbook.histogram.HistogramBinSelection (Tries out different values for the number of bins and replaces each pixel value by the center of its bin)	169
edu.ou.asgbook.motion.HornSchunk (Horn-Schunk optical flow method of motion estimation)	174
edu.ou.asgbook.transforms.HoughTransform (Finds lines in image)	178
edu.ou.asgbook.motion.HungarianAssigner (Optimal assignment algorithm)	181
edu.ou.asgbook.motion.HybridTracker (Estimates motion by finding cross-correlation of objects in one frame to the pixels in the previous frame)	184
edu.ou.asgbook.segmentation.HysteresisSegmenter (Objects consist of pixels that are $> \text{thresh2}$ but have at least one pixel $> \text{thresh1}$)	188
edu.ou.asgbook.filters.Inverter (At every pixel, replaces its value (val) by $(A - \text{val})$)	191
edu.ou.asgbook.motion.KalmanFilter (For the time smoothing of motion vectors)	193
edu.ou.asgbook.io.KmlWriter (Writes data out in KML form, for display in Google Earth or similar program)	197
edu.ou.asgbook.segmentation.LabelResult (Result of segmentation)	201
edu.ou.asgbook.projections.LambertConformal2SP (Lambert Conformation 2 Standard Parallels map projection)	203
edu.ou.asgbook.core.LatLon (A point on the earth's surface typically in WGS84)	206
edu.ou.asgbook.core.LatLonGrid (A geospatial grid of data in equilateral coordinates typically in WGS84 ellipsoid)	209
edu.ou.asgbook.core.LevelSet (A representation of a spatial grid as a set of levels)	219
edu.ou.asgbook.rasterization.Line (A line that connects two points on the earth's surface)	220

edu.ou.asgbook.linearity.LinearityVerifier (Given a 2D array of points, reports error measures of assuming linearity)	224
edu.ou.asgbook.linearity.LinearScaling (Scales pixel values as Ax)	227
edu.ou.asgbook.imgstat.LocalMeasures (Statistics computed in the neighborhood of a pixel)	229
edu.ou.asgbook.filters.LoGEdgeFilter (Laplacian of a Gaussian edge filter)	232
edu.ou.asgbook.linearity.LogScaling (Transforms pixel values as $\log(x)$)	235
edu.ou.asgbook.dataset.MadisTemperature (Reads the ASCII temperature data available at http://madis-data.noaa.gov/public/sfcdumpguest.html)	237
edu.ou.asgbook.filters.MatchedFilter (Convolve an image by a window that is akin to the features we want to extract)	239
edu.ou.asgbook.filters.MaxValueFilter (Finds the highest value pixel in the image)	243
edu.ou.asgbook.thinning.MedialAxisSkeletonization (The MAT method of skeletonizing a grid)	245
edu.ou.asgbook.filters.MedianFilter (A smoothing operation that involves replacing a pixel by the local median)	247
edu.ou.asgbook.motion.MotionEstimator	250
edu.ou.asgbook.filters.MultiFilter (Carries out multiple operations)	251
edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter (Quantizes image into K levels, then does multiscale segmentation Does not implement the pruning techniques discussed in the paper)	253
edu.ou.asgbook.filters.NHighest (Finds the N highest valued-pixels in image)	257
edu.ou.asgbook.filters.NHighestLevelSetImpl (Finds the N highest valued-pixels in image using a levelset implementation)	260
edu.ou.asgbook.dataset.NighttimeLights (Reads night-time lights data in ESRI grid format)	262
edu.ou.asgbook.oban.ObjectiveAnalysisUtils (Utility functions for objective analysis)	264
edu.ou.asgbook.motion.ObjectTracker (Estimates motion based on assigning objects in one frame to objects in the previous frame)	266
edu.ou.asgbook.motion.ObjectTracker.Assigner	270
edu.ou.asgbook.motion.ObjectTracker.CostEstimator	271
edu.ou.asgbook.filters.OrientedEllipseFilter (A non-isotropic smoothing filter)	272
edu.ou.asgbook.histogram.OtsuThresholdSelector (Uses Otsu (1979) to select optimal threshold)	275
edu.ou.asgbook.io.OutputDirectory (Change this to change the output directory that is used by all the main())	278
edu.ou.asgbook.core.Pair< X, Y > (An utility class so that methods can return two objects)	280
edu.ou.asgbook.motion.PhaseCorrelation (Estimate motion based on FFT)	281
edu.ou.asgbook.core.Pixel (A grid point in a spatial grid consists of a location and value)	285
edu.ou.asgbook.io.PngWriter (Writes a spatial grid out as PNG file)	289

edu.ou.asgbook.core.PointObservations (A set of observation points)	294
edu.ou.asgbook.rasterization.Polygon (A polygon consisting of straight edges along the earth's surface)	296
edu.ou.asgbook.datamining.PrimaryCities (Identifies the primary cities in each country)	300
edu.ou.asgbook.rbf.ProjectionPursuit (Approximates a spatial grid by a RBF when nothing is known beyond the number of Gaussians desired) .	302
edu.ou.asgbook.rbf.ProjectionPursuit.NextRBF	306
edu.ou.asgbook.motion.PyramidalCrossCorrelation (Cross-correlation at multiple resolutions)	307
edu.ou.asgbook.imgstat.Quantizer (Develops a quantization scheme using histogram equalization)	310
edu.ou.asgbook.filters.QuickSelect (From Numerical Recipes, a fast way to find the kth smallest item in a list Useful to implement rank filters) .	314
edu.ou.asgbook.rbf.RadialBasisFunction (Finds best fit of a spatial grid to a sum of Gaussians when the centers and sigmas of the Gaussians are known)	316
edu.ou.asgbook.segmentation.RegionGrowing (Common object-identification utility)	318
edu.ou.asgbook.segmentation.RegionProperty (Properties of a region such as geometric (centroid, area, etc) and physical (based on other grid values))	319
edu.ou.asgbook.projections.Remapper (Utilities to remap one map projection to another)	325
edu.ou.asgbook.filters.SaturateFilter (Sets all values < MIN to MIN and all values > MAX to MAX)	327
edu.ou.asgbook.core.ScalarStatistic (A utility class to compute mean, variance of a streaming set of inputs)	329
edu.ou.asgbook.segmentation.Segmenter (Object identification technique) .	332
edu.ou.asgbook.filters.SeparableConvolutionFilter (An optimized convolution filter)	334
edu.ou.asgbook.dataset.SeviriInfraredTemperature (To read binary dump output from WDSS-II (http://www.wdssii.org/))	338
edu.ou.asgbook.filters.SimpleThresholder (Replace pixel values with 1 or 0 depending on whether they are above or below a single threshold) .	341
edu.ou.asgbook.segmentation.SnakeActiveContour (Active contour method of identifying objects)	344
edu.ou.asgbook.segmentation.SnakeActiveContour.Snake	348
edu.ou.asgbook.filters.SobelEdgeFilter (Find edges in a grid)	350
edu.ou.asgbook.filters.SpatialFilter	353
edu.ou.asgbook.filters.SpeckleFilter (Denoising filter that removes speckle) .	355
edu.ou.asgbook.usage.Sprawl (Solution to a classroom assignment to identify regions of urban sprawl from the population density data) . .	358
edu.ou.asgbook.imgstat.StructuralMeasures (Statistics computed in the neighborhood of a pixel)	361
edu.ou.asgbook.dataset.SurfaceAlbedo (Reads lambert-conformal ascii grid)	363

edu.ou.asgbook.segmentation.ThresholdSegmenter (Simple object identification based on a single threshold)	367
edu.ou.asgbook.geocode.UsaZipcode (Find the city for each zipcode in the USA)	370
edu.ou.asgbook.imgstat.VectorQuantizer (Develops a quantization scheme using vector quantization)	372
edu.ou.asgbook.segmentation.WatershedSegmenter (Watershed approach of object identification)	377
edu.ou.asgbook.oban.WeightedAverage (Interpolation methods for point observations)	380
edu.ou.asgbook.oban.WeightedAverageOptimized	384
edu.ou.asgbook.oban.WeightFunction (Used by WeightedAverage)	387
edu.ou.asgbook.dataset.WorldBankGDI (Reads country-by-country Global development index from World Bank)	389

Chapter 4

Automating Analysis of Spatial Grids File Index

4.1 Automating Analysis of Spatial Grids File List

Here is a list of all files with brief descriptions:

AlignAndDifference.java	393
AlignmentEstimator.java	394
Assignment4.java	395
BoundingBox.java	396
CatmullRom.java	397
CityCategories.java	398
CityGdiModels.java	399
ContiguityEnhancedKMeansSegmenter.java	400
ConvolutionFilter.java	401
CountryPolygons.java	402
CressmanWeighting.java	403
CrossCorrelation.java	404
CumulativeDistributionFunction.java	405
DailyRainfall.java	406
DataSimulator.java	407
DataTransform.java	408
Differencer.java	409
DilateErodeFilter.java	410
DilationFilter.java	411
EdgeBased.java	412
Ellipsoid.java	413
EnhancedWatershedSegmenter.java	414
Entropy.java	415

ErodeDilateFilter.java	416
ErosionFilter.java	417
EsriGrid.java	418
EuclideanDT.java	419
EuclideanDTPropagation.java	420
EuclideanDTRecursivePropagation.java	421
EuclideanDTSaito.java	422
FFT.java	423
FFT2D.java	424
FFTBandpassFilter.java	425
FFTConvolutionFilter.java	426
FuzzyCandidateMarket.java	427
FuzzyLogic.java	428
GaussianComponent.java	429
GaussianMixtureModel.java	430
GaussWeighting.java	431
GdiPattern.java	432
GlobalPopulation.java	433
GraylevelCooccurrenceMatrix.java	434
HilditchSkeletonization.java	435
Histogram.java	436
HistogramBinSelection.java	437
HornSchunk.java	438
HoughTransform.java	439
HungarianAssigner.java	440
HybridTracker.java	441
HysteresisSegmenter.java	442
Inverter.java	443
KalmanFilter.java	444
KmlWriter.java	445
LabelResult.java	446
LambertConformal2SP.java	447
LatLon.java	448
LatLonGrid.java	449
LevelSet.java	450
Line.java	451
LinearityVerifier.java	452
LinearScaling.java	453
LocalMeasures.java	454
LoGEdgeFilter.java	455
LogScaling.java	456
MadisTemperature.java	457
MatchedFilter.java	458
MaxValueFilter.java	459
MedialAxisSkeletonization.java	460
MedianFilter.java	461

MotionEstimator.java	462
MultiFilter.java	463
MultiscaleKMeansSegmenter.java	464
NHighest.java	465
NHighestLevelSetImpl.java	466
NighttimeLights.java	467
ObjectiveAnalysisUtils.java	468
ObjectTracker.java	469
OrientedEllipseFilter.java	470
OtsuThresholdSelector.java	471
OutputDirectory.java	472
core/package-info.java	473
datamining/package-info.java	474
dataset/package-info.java	475
distance/package-info.java	476
filters/package-info.java	477
geocode/package-info.java	478
gmm/package-info.java	479
histogram/package-info.java	480
imgstat/package-info.java	481
io/package-info.java	482
linearity/package-info.java	483
motion/package-info.java	484
oban/package-info.java	485
projections/package-info.java	486
rasterization/package-info.java	487
rbf/package-info.java	488
segmentation/package-info.java	489
thinning/package-info.java	490
transforms/package-info.java	491
usage/package-info.java	492
Pair.java	493
PhaseCorrelation.java	494
Pixel.java	495
PngWriter.java	496
PointObservations.java	497
Polygon.java	498
PrimaryCities.java	499
ProjectionPursuit.java	500
PyramidalCrossCorrelation.java	501
Quantizer.java	502
QuickSelect.java	503
RadialBasisFunction.java	504
RegionGrowing.java	505
RegionProperty.java	506
Remapper.java	507

SaturateFilter.java	508
ScalarStatistic.java	509
Segmenter.java	510
SeparableConvolutionFilter.java	511
SeviriInfraredTemperature.java	512
SimpleThresholder.java	513
SnakeActiveContour.java	514
SobelEdgeFilter.java	515
SpatialFilter.java	516
SpeckleFilter.java	517
Sprawl.java	518
StructuralMeasures.java	519
SurfaceAlbedo.java	520
ThresholdSegmenter.java	521
UsaZipcode.java	522
VectorQuantizer.java	523
WatershedSegmenter.java	524
WeightedAverage.java	525
WeightedAverageOptimized.java	526
WeightFunction.java	527
WorldBankGDI.java	528

Chapter 5

Automating Analysis of Spatial Grids Namespace Documentation

5.1 Package edu

Packages

- package [ou](#)

5.2 Package edu.ou

Packages

- package [asgbook](#)

5.3 Package edu.ou.asgbook

Packages

- package [core](#)
Classes that form the basis of spatial analysis techniques discussed in the book.
- package [datamining](#)
Classes discussed in Chapter 8 (Data Mining).
- package [dataset](#)
Classes that provide the ability to read the various datasets that are used to illustrate spatial analysis techniques.
- package [distance](#)
Distance Transform methods (See Chapter 4).
- package [filters](#)
Most of these filters are neighborhood/window based and are discussed in Chapter 5.
- package [geocode](#)
Geocoding is discussed in Chapter 2.
- package [gmm](#)
A parametric approximation to an image (See Chapter 3).
- package [histogram](#)
Histograms and histogram-based techniques are discussed in Chapter 4.
- package [imgstat](#)
Texture, vector quantization and other image statistics techniques discussed in Chapter 4.
- package [io](#)
Helper classes to read ESRI grids and write out KML/PNG.
- package [linearity](#)
Linearity verification and data transformation are discussed in Chapter 2.
- package [motion](#)
Change and motion estimation techniques discussed in Chapter 7.
- package [oban](#)

Techniques to interpolate point observations to a spatial grid discussed in Chapter 3.

- package [projections](#)

Map projection techniques discussed in Chapter 2.

- package [rasterization](#)

Rasterization techniques discussed in Chapter 2.

- package [rbf](#)

Parametric approximations discussed in Chapter 3.

- package [segmentation](#)

Object identification techniques discussed in Chapter 6.

- package [thinning](#)

Skeletonization techniques discussed in Chapter 5.

- package [transforms](#)

Fourier Transforms are discussed in Chapter 5.

- package [usage](#)

A classroom assignment; an example algorithm.

5.4 Package edu.ou.asgbook.core

Classes that form the basis of spatial analysis techniques discussed in the book.

Classes

- class [LatLon](#)
A point on the earth's surface typically in WGS84.
- class [LatLonGrid](#)
A geospatial grid of data in equilateral coordinates typically in WGS84 ellipsoid.
- class [LevelSet](#)
A representation of a spatial grid as a set of levels.
- class [Pair< X, Y >](#)
An utility class so that methods can return two objects.
- class [Pixel](#)
A grid point in a spatial grid consists of a location and value.
- class [PointObservations](#)
A set of observation points.
- class [ScalarStatistic](#)
A utility class to compute mean, variance of a streaming set of inputs.

5.4.1 Detailed Description

Classes that form the basis of spatial analysis techniques discussed in the book.

These are discussed in Chapter 3.

5.5 Package edu.ou.asgbook.datamining

Classes discussed in Chapter 8 (Data Mining).

Classes

- class [CityCategories](#)
Obtains city data for clustering.
- class [CityGdiModels](#)
Applies different data mining models to each city.
- class [FuzzyCandidateMarket](#)
Uses heuristic rules to choose the next market to enter.
- class [FuzzyLogic](#)
A simple fuzzy logic engine.
- class [GdiPattern](#)
The training pattern for each city.
- class [PrimaryCities](#)
Identifies the primary cities in each country.

5.5.1 Detailed Description

Classes discussed in Chapter 8 (Data Mining).

5.6 Package edu.ou.asgbook.dataset

Classes that provide the ability to read the various datasets that are used to illustrate spatial analysis techniques.

Classes

- class [CountryPolygons](#)
Reads country-by-country coordinates from a KML placemarks file.
- class [DailyRainfall](#)
Reads the ASCII precipitation data available at <http://madis-data.noaa.gov/public/hydrodumpguest.html>.
- class [GlobalPopulation](#)
Reads the ASCII population data available at <http://sedac.ciesin.columbia.edu/gpw>.
- class [MadisTemperature](#)
Reads the ASCII temperature data available at <http://madis-data.noaa.gov/public/sfcdumpguest.html>.
- class [NighttimeLights](#)
Reads night-time lights data in ESRI grid format.
- class [SeviriInfraredTemperature](#)
To read binary dump output from WDSS-II (<http://www.wdssii.org/>).
- class [SurfaceAlbedo](#)
Reads lambert-conformal ascii grid.
- class [WorldBankGDI](#)
Reads country-by-country Global development index from World Bank.

5.6.1 Detailed Description

Classes that provide the ability to read the various datasets that are used to illustrate spatial analysis techniques.

5.7 Package edu.ou.asgbook.distance

Distance Transform methods (See Chapter 4).

Classes

- interface [EuclideanDT](#)
- class [EuclideanDTPropagation](#)
Implementation of Euclidean distance that updates the distance instead of computing it afresh each time.
- class [EuclideanDTRecursivePropagation](#)
Note that this class is only for illustrative purposes.
- class [EuclideanDTSaito](#)
The Saito technique of computing the distance transform by calculating in the two directions separately.

5.7.1 Detailed Description

Distance Transform methods (See Chapter 4).

5.8 Package edu.ou.asgbook.filters

Most of these filters are neighborhood/window based and are discussed in Chapter 5.

Classes

- class [ConvolutionFilter](#)
Convolve an image by a window.
- class [DilateErodeFilter](#)
Carries out paired dilation followed by erosion for filling in holes.
- class [DilationFilter](#)
Expands entities by taking a local maximum.
- class [ErodeDilateFilter](#)
Carries out paired erosion followed by dilation for denoising.
- class [ErosionFilter](#)
Reduces the size of entities by taking a local minimum.
- class [Inverter](#)
at every pixel, replaces its value (val) by $(A - val)$
- class [LoGEdgeFilter](#)
Laplacian of a Gaussian edge filter.
- class [MatchedFilter](#)
Convolve an image by a window that is akin to the features we want to extract.
- class [MaxValueFilter](#)
Finds the highest value pixel in the image.
- class [MedianFilter](#)
A smoothing operation that involves replacing a pixel by the local median.
- class [MultiFilter](#)
Carries out multiple operations.
- class [NHighest](#)
Finds the N highest valued-pixels in image.

- class [NHighestLevelSetImpl](#)
Finds the N highest valued-pixels in image using a levelset implementation.
- class [OrientedEllipseFilter](#)
A non-isotropic smoothing filter.
- class [QuickSelect](#)
From Numerical Recipes, a fast way to find the k th smallest item in a list Useful to implement rank filters.
- class [SaturateFilter](#)
Sets all values $< MIN$ to MIN and all values $> MAX$ to MAX .
- class [SeparableConvolutionFilter](#)
An optimized convolution filter.
- class [SimpleThresholder](#)
Replace pixel values with 1 or 0 depending on whether they are above or below a single threshold.
- class [SobelEdgeFilter](#)
Find edges in a grid.
- interface [SpatialFilter](#)
- class [SpeckleFilter](#)
Denoising filter that removes speckle.

5.8.1 Detailed Description

Most of these filters are neighborhood/window based and are discussed in Chapter 5.

5.9 Package edu.ou.asgbook.geocode

Geocoding is discussed in Chapter 2.

Classes

- class [UsaZipcode](#)
Find the city for each zipcode in the USA.

5.9.1 Detailed Description

Geocoding is discussed in Chapter 2.

5.10 Package edu.ou.asgbook.gmm

A parametric approximation to an image (See Chapter 3).

Classes

- class [GaussianComponent](#)
Component of a Gaussian Mixture Model.
- class [GaussianMixtureModel](#)
A parametric approximation of a spatial grid as a sum of Gaussians.

5.10.1 Detailed Description

A parametric approximation to an image (See Chapter 3).

5.11 Package edu.ou.asgbook.histogram

Histograms and histogram-based techniques are discussed in Chapter 4.

Classes

- class [CumulativeDistributionFunction](#)
Forms a CDF from a [Histogram](#).
- class [Entropy](#)
Compute entropy from a histogram.
- class [Histogram](#)
A histogram is an empirical probability distribution.
- class [HistogramBinSelection](#)
Tries out different values for the number of bins and replaces each pixel value by the center of its bin.
- class [OtsuThresholdSelector](#)
Uses Otsu (1979) to select optimal threshold.

5.11.1 Detailed Description

Histograms and histogram-based techniques are discussed in Chapter 4.

5.12 Package edu.ou.asgbook.imgstat

Texture, vector quantization and other image statistics techniques discussed in Chapter 4.

Classes

- class [GraylevelCooccurrenceMatrix](#)
Computes texture properties from a GLCM.
- class [LocalMeasures](#)
Statistics computed in the neighborhood of a pixel.
- class [Quantizer](#)
Develops a quantization scheme using histogram equalization.
- class [StructuralMeasures](#)
Statistics computed in the neighborhood of a pixel.
- class [VectorQuantizer](#)
Develops a quantization scheme using vector quantization.

5.12.1 Detailed Description

Texture, vector quantization and other image statistics techniques discussed in Chapter 4.

5.13 Package edu.ou.asgbook.io

Helper classes to read ESRI grids and write out KML/PNG.

Classes

- class [EsriGrid](#)
Read an ESRI grid.
- class [KmlWriter](#)
Writes data out in KML form, for display in Google Earth or similar program.
- class [OutputDirectory](#)
Change this to change the output directory that is used by all the main().
- class [PngWriter](#)
Writes a spatial grid out as PNG file.

5.13.1 Detailed Description

Helper classes to read ESRI grids and write out KML/PNG.

5.14 Package edu.ou.asgbook.linearity

Linearity verification and data transformation are discussed in Chapter 2.

Classes

- class [DataTransform](#)
Transform pixel values, usually to meet linearity requirements.
- class [LinearityVerifier](#)
Given a 2D array of points, reports error measures of assuming linearity.
- class [LinearScaling](#)
Scales pixel values as Ax .
- class [LogScaling](#)
Transforms pixel values as $\log(x)$.

5.14.1 Detailed Description

Linearity verification and data transformation are discussed in Chapter 2.

5.15 Package edu.ou.asgbook.motion

Change and motion estimation techniques discussed in Chapter 7.

Classes

- class [AlignAndDifference](#)
Aligns two grids and then computes their difference.
- class [CrossCorrelation](#)
Estimates motion using cross-correlation.
- class [Differencer](#)
Just computes a pixel-by-pixel difference.
- class [EdgeBased](#)
Estimates motion based on the displacement of edges.
- class [HornSchunk](#)
Horn-Schunk optical flow method of motion estimation.
- class [HungarianAssigner](#)
Optimal assignment algorithm.
- class [HybridTracker](#)
Estimates motion by finding cross-correlation of objects in one frame to the pixels in the previous frame.
- class [KalmanFilter](#)
For the time smoothing of motion vectors.
- interface [MotionEstimator](#)
- class [ObjectTracker](#)
Estimates motion based on assigning objects in one frame to objects in the previous frame.
- class [PhaseCorrelation](#)
Estimate motion based on FFT.
- class [PyramidalCrossCorrelation](#)
Cross-correlation at multiple resolutions.

5.15.1 Detailed Description

Change and motion estimation techniques discussed in Chapter 7.

5.16 Package edu.ou.asgbook.oban

Techniques to interpolate point observations to a spatial grid discussed in Chapter 3.

Classes

- class [CressmanWeighting](#)
An interpolation method that uses $1/r^2$.
- class [GaussWeighting](#)
An interpolation method that uses $\exp(-1/r^2)$.
- class [ObjectiveAnalysisUtils](#)
Utility functions for objective analysis.
- class [WeightedAverage](#)
Interpolation methods for point observations.
- class [WeightedAverageOptimized](#)
- interface [WeightFunction](#)
Used by [WeightedAverage](#).

5.16.1 Detailed Description

Techniques to interpolate point observations to a spatial grid discussed in Chapter 3.

5.17 Package edu.ou.asgbook.projections

Map projection techniques discussed in Chapter 2.

Classes

- class [Ellipsoid](#)
An ellipsoidal approximation to the earth.
- class [LambertConformal2SP](#)
Lambert Conformation 2 Standard Parallels map projection.
- class [Remapper](#)
Utilities to remap one map projection to another.

5.17.1 Detailed Description

Map projection techniques discussed in Chapter 2.

5.18 Package edu.ou.asgbook.rasterization

Rasterization techniques discussed in Chapter 2.

Classes

- class [BoundingBox](#)
A rectangular bounding box of a polygon.
- class [CatmullRom](#)
A Catmull-Rom spline, a local spline.
- class [Line](#)
A line that connects two points on the earth's surface.
- class [Polygon](#)
A polygon consisting of straight edges along the earth's surface.

5.18.1 Detailed Description

Rasterization techniques discussed in Chapter 2.

5.19 Package edu.ou.asgbook.rbf

Parametric approximations discussed in Chapter 3.

Classes

- class [DataSimulator](#)
Simulates RBF data to be fit.
- class [ProjectionPursuit](#)
Approximates a spatial grid by a RBF when nothing is known beyond the number of Gaussians desired.
- class [RadialBasisFunction](#)
Finds best fit of a spatial grid to a sum of Gaussians when the centers and sigmas of the Gaussians are known.

5.19.1 Detailed Description

Parametric approximations discussed in Chapter 3.

5.20 Package edu.ou.asgbook.segmentation

Object identification techniques discussed in Chapter 6.

Classes

- class [ContiguityEnhancedKMeansSegmenter](#)
Objects consist of pixels that are grown from initial centers using K-means.
- class [EnhancedWatershedSegmenter](#)
Enhanced watershed segmentation following Lakshmanan, Hondl and Rabin.
- class [HysteresisSegmenter](#)
Objects consist of pixels that are $> \text{thresh2}$ but have at least one pixel $> \text{thresh1}$.
- class [LabelResult](#)
Result of segmentation.
- class [MultiscaleKMeansSegmenter](#)
Quantizes image into K levels, then does multiscale segmentation Does not implement the pruning techniques discussed in the paper.
- class [RegionGrowing](#)
Common object-identification utility.
- class [RegionProperty](#)
Properties of a region such as geometric (centroid, area, etc) and physical (based on other grid values).
- interface [Segmenter](#)
Object identification technique.
- class [SnakeActiveContour](#)
Active contour method of identifying objects.
- class [ThresholdSegmenter](#)
Simple object identification based on a single threshold.
- class [WatershedSegmenter](#)
Watershed approach of object identification.

5.20.1 Detailed Description

Object identification techniques discussed in Chapter 6.

5.21 Package edu.ou.asgbook.thinning

Skeletonization techniques discussed in Chapter 5.

Classes

- class [HilditchSkeletonization](#)
Hilditch method of skeletonizing a grid.
- class [MedialAxisSkeletonization](#)
The MAT method of skeletonizing a grid.

5.21.1 Detailed Description

Skeletonization techniques discussed in Chapter 5.

5.22 Package edu.ou.asgbook.transforms

Fourier Transforms are discussed in Chapter 5.

Classes

- class [AlignmentEstimator](#)
Estimate the degree of spatial displacement between two similar grids.
- class [FFT](#)
FFT based on Sedgewick and Wayne.
- class [FFT2D](#)
Two-dimensional FFT.
- class [FFTBandpassFilter](#)
Removes noise (high frequencies) and the gross signal (low frequencies).
- class [FFTConvolutionFilter](#)
An optimization for convolution using FFTs.
- class [HoughTransform](#)
Finds lines in image.

5.22.1 Detailed Description

Fourier Transforms are discussed in Chapter 5.

The Hough Transform is discussed in Chapter 6.

5.23 Package edu.ou.asgbook.usage

A classroom assignment; an example algorithm.

Classes

- class [Assignment4](#)
(1) Find optimal threshold on log(pop) Find distance of every grid point to a point < thresh Find optimal threshold of distance values Threshold image to keep only values < threshold
- class [Sprawl](#)
Solution to a classroom assignment to identify regions of urban sprawl from the population density data.

5.23.1 Detailed Description

A classroom assignment; an example algorithm.

Chapter 6

Automating Analysis of Spatial Grids Class Documentation

6.1 edu.ou.asgbook.motion.AlignAndDifference Class Reference

Aligns two grids and then computes their difference.

Public Member Functions

- [LatLonGrid compute](#) ([LatLonGrid](#) data0, [LatLonGrid](#) data1, [Pair](#)< [LatLonGrid](#), [LatLonGrid](#) > uv)
- [LatLonGrid compute](#) ([LatLonGrid](#) data0, [LatLonGrid](#) data1, [Pair](#)< [LatLonGrid](#), [LatLonGrid](#) > uv, int MOT_SCALE)

6.1.1 Detailed Description

Aligns two grids and then computes their difference.

Author:

v.lakshmanan

6.1.2 Member Function Documentation

6.1.2.1 [LatLonGrid](#) edu.ou.asgbook.motion.AlignAndDifference.compute ([LatLonGrid](#) data0, [LatLonGrid](#) data1, [Pair](#)< [LatLonGrid](#), [LatLonGrid](#) > uv, int MOT_SCALE)

```

20
21     LatLonGrid result = LatLonGrid.copyOf(data1);
22
23     final float mot_scale = MOT_SCALE; // integer division truncates
24     for (int i=0; i < result.getNumLat(); ++i){
25         for (int j=0; j < result.getNumLon(); ++j){
26             // align by moving data0 to match up with data1
27             // then compute difference
28             int aligned0 = data0.getValue(i,j);
29             // find motion at this point
30             int motx = Math.round(uv.first.getValue(i,j) / mot_scale);
31             int moty = Math.round(uv.second.getValue(i,j) / mot_scale);
32             // grab pixel from old location
33             int oldx = i - motx;
34             int oldy = j - moty;
35             if (data0.isValid(oldx, oldy)){
36                 aligned0 = data0.getValue(oldx, oldy);
37             }
38             int diff = data1.getValue(i,j) - aligned0;
39             result.setValue(i,j, diff);
40         }
41     }
42
43     return result;
44 }
```

6.1.2.2 [LatLonGrid](#) edu.ou.asgbook.motion.AlignAndDifference.compute ([LatLonGrid](#) data0, [LatLonGrid](#) data1, [Pair](#)< [LatLonGrid](#), [LatLonGrid](#) > uv)

```

16
17     return compute(data0, data1, uv, 1);
18 }
```

6.2 edu.ou.asgbook.transforms.AlignmentEstimator Class Reference

Estimate the degree of spatial displacement between two similar grids.

Public Member Functions

- [AlignmentEstimator](#) (int maxu, int maxv, [LatLonGrid](#) a, [LatLonGrid](#) b)
convolve a with b and find where the maximum correlation lies

Static Public Member Functions

- static [Pixel](#) [computeCentroid](#) ([LatLonGrid](#) a)
- static void [main](#) (String[] args) throws Exception

Package Attributes

- final int [MAXU](#)
- final int [MAXV](#)
- int [motNS](#)
- int [motEW](#)

6.2.1 Detailed Description

Estimate the degree of spatial displacement between two similar grids.

Author:

v.lakshmanan

6.2.2 Constructor & Destructor Documentation

6.2.2.1 edu.ou.asgbook.transforms.AlignmentEstimator.AlignmentEstimator (int maxu, int maxv, [LatLonGrid](#) a, [LatLonGrid](#) b)

convolve a with b and find where the maximum correlation lies

```
25                                     {
26         this.MAXU = maxu;
27         this.MAXV = maxv;
```

```

28
29 // a
30 Complex[][] in1 = FFT2D.fft(FFT2D.zeropad(a));
31
32 // zero-out an area of thickness MAXU/MAXV around the boundary to avoid boundary is
33 LatLonGrid centerb = LatLonGrid.copyOf(b);
34 int minx = MAXU;
35 int miny = MAXV;
36 int maxx = centerb.getNumLat() - minx;
37 int maxy = centerb.getNumLon() - miny;
38 for (int i=0; i < b.getNumLat(); ++i){
39     for (int j=0; j < b.getNumLon(); ++j){
40         if (i < minx || j < miny || i > maxx || j > maxy){
41             centerb.setValue(i, j, 0);
42         }
43     }
44 }
45 Complex[][] in2 = FFT2D.fft(FFT2D.zeropad(centerb));
46
47 // find phase shift at this point
48 for (int i=0; i < in1.length; ++i) for (int j=0; j < in1[0].length; ++j){
49     in1[i][j] = in1[i][j].multiply(in2[i][j].conjugate());
50     in1[i][j] = in1[i][j].multiply( 1.0 / in1[i][j].norm() );
51 }
52 // take ifft
53 Complex[][] result = FFT2D.ifft(in1);
54
55 // find location at which the convolved result is maximum
56 double bestValue = Integer.MIN_VALUE;
57 int startx = 0; // result.length/2 - MAXU;
58 int starty = 0; // result[0].length/2 - MAXV;
59 int endx = result.length; // /2 + MAXU;
60 int endy = result[0].length; // /2 + MAXV;
61 for (int i=startx; i < endx; ++i) for (int j=starty; j < endy; ++j){
62     if ( result[i][j].normsq() > bestValue ){
63         bestValue = result[i][j].real;
64         motNS = -i;
65         motEW = -j;
66     }
67 }
68
69 // we don't want a 345-degree phase shift; we want it to be 15-degrees
70 if ( Math.abs(motNS) > result.length/2 ){
71     if (motNS < 0) motNS += result.length;
72     else motNS -= result.length;
73 }
74 if ( Math.abs(motEW) > result[0].length/2 ){
75     if (motEW < 0) motEW += result[0].length;
76     else motEW -= result[0].length;
77 }
78 }

```

6.2.3 Member Function Documentation

6.2.3.1 static Pixel edu.ou.asgbook.transforms.AlignmentEstimator.computeCentroid (LatLonGrid a) [static]

```

80                                     {
81         double sumx = 0;
82         double sumy = 0;
83         double sumwt = 0;
84         int N = 0;
85         for (int i=0; i < a.getNumLat(); ++i) for (int j=0; j < a.getNumLon(); ++j){
86             double wt = a.getValue(i,j);
87             sumx += i * wt;
88             sumy += j * wt;
89             sumwt += wt;
90             ++N;
91         }
92         return new Pixel((int)Math.round(sumx/sumwt), (int)Math.round(sumy/sumwt), (int)Math.round(sumwt));
93     }

```

6.2.3.2 static void edu.ou.asgbook.transforms.AlignmentEstimator.main (String[] args) throws Exception [static]

```

95                                     {
96         // because the alignment doesn't really check lat-lon extents,
97         // cropping from offset corners will look like translation ...
98         LatLonGrid conus = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
99         LatLonGrid[] grids = new LatLonGrid[2];
100        grids[0] = conus.crop(900, 2500, 256, 256);
101        int motx = 5; int moty = 9;
102        grids[1] = conus.crop(900-motx, 2500-moty, 256, 256);
103
104        // do alg
105        AlignmentEstimator alg = new AlignmentEstimator(30,30,grids[0], grids[1]);
106        System.out.println("Motion N/S = " + alg.motNS + " true N/S=" + motx);
107        System.out.println("Motion E/W = " + alg.motEW + " true E/W=" + moty);
108
109        System.out.println("Centroid of first = " + computeCentroid(grids[0]));
110        System.out.println("Centroid of second = " + computeCentroid(grids[1]));
111
112        // based on edges alone
113        SobelEdgeFilter edgeFilter = new SobelEdgeFilter();
114        LatLonGrid edge1 = edgeFilter.edgeFilter(grids[0]);
115        LatLonGrid edge2 = edgeFilter.edgeFilter(grids[1]);
116        AlignmentEstimator alg2 = new AlignmentEstimator(30,30,edge1, edge2);
117        System.out.println("Edge Motion N/S = " + alg2.motNS );
118        System.out.println("Edge Motion E/W = " + alg2.motEW );
119    }

```

6.2.4 Member Data Documentation

6.2.4.1 **final int** [edu.ou.asgbook.transforms.AlignmentEstimator.MAXU](#)
[package]

6.2.4.2 **final int** [edu.ou.asgbook.transforms.AlignmentEstimator.MAXV](#)
[package]

6.2.4.3 **int** [edu.ou.asgbook.transforms.AlignmentEstimator.motEW](#)
[package]

6.2.4.4 **int** [edu.ou.asgbook.transforms.AlignmentEstimator.motNS](#)
[package]

6.3 edu.ou.asgbook.usage.Assignment4 Class Reference

(1) Find optimal threshold on log(pop) Find distance of every grid point to a point < thresh Find optimal threshold of distance values Threshold image to keep only values < threshold

Static Public Member Functions

- static void `main` (String[] args) throws Exception

6.3.1 Detailed Description

(1) Find optimal threshold on log(pop) Find distance of every grid point to a point < thresh Find optimal threshold of distance values Threshold image to keep only values < threshold

Author:

v.lakshmanan

6.3.2 Member Function Documentation

6.3.2.1 static void edu.ou.asgbook.usage.Assignment4.main (String[] args) throws Exception [static]

```

30                                     {
31         File outdir = OutputDirectory.getDefault("assignment4");
32
33         // read input
34         LatLon nwCorner = new LatLon(60, -130);
35         LatLon seCorner = new LatLon(7, -52);
36         // LatLonGrid conus = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulation
37         LatLonGrid conus = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
38         conus = conus.crop(conus.getRow(nwCorner),
39                             conus.getCol(nwCorner),
40                             conus.getRow(seCorner) - conus.getRow(nwCorner),
41                             conus.getCol(seCorner) - conus.getCol(nwCorner));
42         KmlWriter.write(conus, outdir, "orig", PngWriter.createCoolToWarmColormap());
43
44         // find threshold
45         int popthresh = -1;
46         {
47             final int MIN = 0;
48             final int MAX = 500;
49             final int incr = 10;
50             Histogram hist = new Histogram(MIN, incr, (MAX-MIN)/incr );

```

```
51         hist.update(conus);
52         popthresh = new OtsuThresholdSelector(hist).getOptimalThreshold();
53         System.out.println("Optimal population threshold=" + popthresh);
54     }
55
56     // threshold
57     //LatLonGrid threshed = new SimpleThresholder(popthresh).filter(conus);
58     //KmlWriter.write(threshed, outdir, "thresh", PngWriter.createCoolToWarmColormap());
59
60     // distance to points > thresh
61     LatLonGrid distToCity = new EuclideanDTSaito().getDistanceTransform(conus, popthresh);
62     KmlWriter.write(distToCity, outdir, "distToCity", PngWriter.createCoolToWarmColormap());
63
64     // optimal threshold on distance
65     int distthresh = -1;
66     {
67         final int MIN = 0;
68         final int MAX = 10000;
69         final int incr = 10;
70         Histogram hist = new Histogram(MIN, incr, (MAX-MIN)/incr );
71         hist.update(distToCity);
72         distthresh = new OtsuThresholdSelector(hist).getOptimalThreshold();
73         System.out.println("Optimal distance threshold=" + distthresh);
74     }
75
76     // threshold by distance to find metropolitan areas
77     LatLonGrid boondocks = new SimpleThresholder(distthresh/2).filter(distToCity);
78     LatLonGrid metros = new Inverter(1).filter(boondocks);
79     KmlWriter.write(metros, outdir, "metros", PngWriter.createCoolToWarmColormap());
80
81
82 }
```


6.4 edu.ou.asgbook.rasterization.BoundingBox Class Reference

A rectangular bounding box of a polygon.

Public Member Functions

- [BoundingBox](#) ([LatLon](#)[] vertices)
- boolean [contains](#) (double x, double y)
- void [update](#) ([BoundingBox](#) a)

Static Public Member Functions

- static [BoundingBox copyOf](#) ([BoundingBox](#) a)

6.4.1 Detailed Description

A rectangular bounding box of a polygon.

It can sometimes be cheaper to use a bounding box instead of the accurate locations and do the real calculation only if the bounding box passes.

Author:

valliappa.lakshmanan

6.4.2 Constructor & Destructor Documentation

6.4.2.1 edu.ou.asgbook.rasterization.BoundingBox.BoundingBox ([LatLon](#)[] vertices)

```
22                                     {
23         ScalarStatistic lat = new ScalarStatistic();
24         ScalarStatistic lon = new ScalarStatistic();
25         for (int i=0; i < vertices.length; ++i){
26             lat.update(vertices[i].getLat());
27             lon.update(vertices[i].getLon());
28         }
29         maxx = lat.getMax();
30         maxy = lon.getMax();
31         minx = lat.getMin();
32         miny = lon.getMin();
33     }
```

6.4.3 Member Function Documentation

6.4.3.1 boolean edu.ou.asgbook.rasterization.BoundingBox.contains (double x, double y)

```
35                                     {
36         return (x >= minx && x <= maxx && y >= miny && y <= maxy);
37     }
```

6.4.3.2 static [BoundingBox](#) edu.ou.asgbook.rasterization.BoundingBox.copyOf ([BoundingBox a](#)) [static]

```
43                                     {
44         BoundingBox b = new BoundingBox();
45         b.minx = a.minx;
46         b.maxx = a.maxx;
47         b.miny = a.miny;
48         b.maxy = a.maxy;
49         return b;
50     }
```

6.4.3.3 void edu.ou.asgbook.rasterization.BoundingBox.update ([BoundingBox a](#))

```
52                                     {
53         minx = Math.min(minx, a.minx);
54         miny = Math.min(miny, a.miny);
55         maxx = Math.max(maxx, a.maxx);
56         maxy = Math.max(maxy, a.maxy);
57     }
```

6.5 edu.ou.asgbook.rasterization.CatmullRom Class Reference

A Catmull-Rom spline, a local spline.

Static Public Member Functions

- static double[] [interpolate](#) (double[] x1, double[] y1, double[] x2)
Determines the y coordinates for the given x2 by interpolating the spline control points (x1,y1).
- static double[] [sort_and_interpolate](#) (double[] x1, double[] y1, double[] x2)
- static List<[Pixel](#)> [getPositionIn](#) (double[] controllat, double[] controllon, [LatLonGrid](#) grid)
- static void [main](#) (String args[]) throws Exception

Classes

- class [XtoY](#)

6.5.1 Detailed Description

A Catmull-Rom spline, a local spline.

Author:

valliappa.lakshmanan

6.5.2 Member Function Documentation

6.5.2.1 static List<[Pixel](#)> edu.ou.asgbook.rasterization.CatmullRom.getPositionIn (double[] controllat, double[] controllon, [LatLonGrid](#) grid) [static]

```

130
131     List<Pixel> result = new ArrayList<Pixel>();
132     // we want to find the intersection at all the lat of the grid
133     double[] lat2 = new double[grid.getNumLat()];
134     for (int i=0; i < lat2.length; ++i){
135         lat2[i] = grid.getLocation(i, 0).getLat(); // lat of row
136     }
137     double[] lon2 = sort_and_interpolate(controllat, controllon, lat2);
138     for (int i=0; i < lon2.length; ++i){
139         int col = grid.getCol(new LatLon(lat2[i],lon2[i])); // col to fill in

```

```

140         result.add(new Pixel(i,col,grid.getValue(i,col)));
141     }
142     return result;
143 }

```

6.5.2.2 static double [] edu.ou.asgbook.rasterization.CatmullRom.interpolate (double[] x1, double[] y1, double[] x2) [static]

Determines the y coordinates for the given x2 by interpolating the spline control points (x1,y1).

The control points need to be sorted in x.

```

33                                                     {
34         // result: initialize at lower-bound value
35         if ( x1.length == 0 ) return new double[x2.length];
36         double[] y2 = new double[x2.length];
37         for (int i=0; i < y2.length; ++i){
38             y2[i] = y1[0];
39         }
40
41         // every interval is p1 <= p2 <= p3 where p2 is resampling position
42         double p3 = x2[0] - 1; // below first value
43         for (int i=0; i < x2.length; ++i){
44             // find interval which contains p2
45             double p2 = x2[i];
46             if ( p2 <= x1[0] ) { y2[i] = y1[0]; continue; }
47             if ( p2 >= back(x1) ) { y2[i] = back(y1); continue; }
48             int j = 0;
49             while (j < (int)x1.length && p2 > x1[j]){ ++j; }
50             --j; //if ( p2 < x1[j] ) --j;
51
52             double p1 = x1[j];
53             p3 = x1[j+1];
54
55             // j and j+1 will be in bounds but j-1 and j+2 may not be
56             int j1 = j-1; if (j1 < 0) j1 = 0;
57             int j2 = j+2; if (j2 > (x1.length-1)) j2 = x1.length-1;
58
59             // spline
60             double dx  = 1.0f / (p3 - p1);
61             double dx1 = 1.0f / (p3 - x1[j1]);
62             double dx2 = 1.0f / (x1[j2] - p1);
63             double dy  = (y1[j+1] - y1[j]) * dx;
64             double yd1 = (y1[j+1] - y1[j1]) * dx1;
65             double yd2 = (y1[j2] - y1[j]) * dx2;
66             double a0y = y1[j];
67             double a1y = yd1;
68             double a2y = dx * ( 3*dy - 2*yd1 - yd2);
69             double a3y = dx*dx*(-2*dy + yd1 + yd2);
70
71             // cubic polynomial
72             double x = p2 - p1;
73             y2[i] = ((a3y*x + a2y)*x + a1y)*x + a0y;

```

```

74         }
75         return y2;
76     }

```

6.5.2.3 static void edu.ou.asgbook.rasterization.CatmullRom.main (String args[]) throws Exception [static]

```

145         {
146             LatLonGrid grid = new LatLonGrid(100,100,0,new LatLon(100,-90),0.01,0.01);
147             double[] controlx = new double[]{99.4,99.3,99.5,99.7};
148             double[] controly = new double[]{-89.5,-89.3,-89.6,-89.4};
149
150             List<Pixel> pixels = getPositionIn(controlx, controly, grid);
151             for (Pixel p : pixels){
152                 grid.setValue(p.getRow(), p.getCol(), 10);
153             }
154
155             File out = OutputDirectory.getDefault("raster");
156             KmlWriter.write(grid, out, "drawspline", PngWriter.createCoolToWarmColormap());
157         }

```

6.5.2.4 static double [] edu.ou.asgbook.rasterization.CatmullRom.sort_ and interpolate (double[] x1, double[] y1, double[] x2) [static]

```

90         {
91
92             int N = x1.length;
93
94             // create structure for sorting
95             XtoY[] data = new XtoY[N];
96             for (int i=0; i < N; ++i){
97                 data[i] = new XtoY();
98                 data[i].orig_index = i;
99                 data[i].scaledx = (int)Math.round(x1[i] * 1000 + 0.5);
100                 data[i].scaledy = (int)Math.round(y1[i] * 1000 + 0.5);
101             }
102             Arrays.sort(data);
103
104             // create input data
105             double[] x= new double[N];
106             double[] y= new double[N];
107             int curr = 0;
108             for (int i=0; i < N; ++i){
109                 // if you have two or more y values for same x, then use avg
110                 int start_i = i;
111                 while ( (i+1) < N && data[i].scaledx == data[i+1].scaledx ){
112                     ++i;
113                 }
114                 x[curr] = x1[data[i].orig_index];
115                 double sumy = 0;

```

```
116         for (int k=start_i; k <= i; ++k){
117             sumy += y1[data[k].orig_index];
118         }
119         y[curr] = sumy/(i-start_i+1);
120         ++curr;
121     }
122
123     x = Arrays.copyOf(x, curr);
124     y = Arrays.copyOf(y, curr);
125
126     // call interpolate
127     return interpolate( x, y, x2 );
128 }
```

6.6 edu.ou.asgbook.datamining.CityCategories Class Reference

Obtains city data for clustering.

Static Public Member Functions

- static void `main` (String[] args) throws Exception

6.6.1 Detailed Description

Obtains city data for clustering.

Author:

valliappa.lakshmanan

6.6.2 Member Function Documentation

6.6.2.1 static void edu.ou.asgbook.datamining.CityCategories.main (String[] args) throws Exception [static]

```

29                                     {
30         // create output directory
31         File out = OutputDirectory.getDefault("citycategories");
32         final boolean SMALL = true;
33
34         // read input (crop to cover China)
35         LatLonGrid pop = GlobalPopulation.read(GlobalPopulation.WORLD);
36         if (SMALL){
37             pop = pop.crop(900, 6000, 800, 1600); // China mainly
38         }
39         KmlWriter.write(pop, out, "modelpop", PngWriter.createRandomColormap());
40
41         LatLonGrid nightTimeLights = NighttimeLights.read(NighttimeLights.WORLD).remapTo(pop);
42         KmlWriter.write(nightTimeLights, out, "modellights", PngWriter.createCoolToWarmColormap());
43
44         EnhancedWatershedSegmenter seg = new EnhancedWatershedSegmenter(10, 1, 600, 10, 5);
45         LabelResult allcities = seg.label(pop);
46         KmlWriter.write(allcities.label, out, "modelcities", PngWriter.createRandomColormap());
47
48         // write out cluster file
49         String filename = out.getAbsolutePath()+"/citydata.txt";
50         PrintWriter writer = new PrintWriter(new FileWriter(filename));
51         writer.println("Pop light");
52         RegionProperty[] population = RegionProperty.compute(allcities, pop);
53         RegionProperty[] lighting = RegionProperty.compute(allcities, nightTimeLights);
54         for (int i=1; i < population.length; ++i){

```

```
55         writer.println(population[i].getCval() + " " + lighting[i].getCval());
56     }
57     writer.close();
58     System.out.println("Wrote " + filename);
59
60     // compute the category of each (based on clustering result)
61     int[] categories = new int[population.length];
62     for (int i=1; i < categories.length; ++i){
63         categories[i] = computeCategory( population[i].getCval(), lighting[i].getCval()
64     }
65     LatLonGrid result = LatLonGrid.copyOf(allcities.label);
66     result.setMissing(0);
67     result.fill(result.getMissing());
68     for (int i=0; i < result.getNumLat(); ++i){
69         for (int j=0; j < result.getNumLon(); ++j){
70             int cityno = allcities.label.getValue(i,j);
71             if ( cityno > 0 ){
72                 result.setValue(i,j, categories[cityno]);
73             }
74         }
75     }
76     KmlWriter.write(result, out, "citycategories", PngWriter.createCoolToWarmColormap());
77 }
```


6.7 edu.ou.asgbook.datamining.CityGdiModels Class Reference

Applies different data mining models to each city.

Static Public Member Functions

- static double[][] [findPatterns](#) ([LabelResult](#) cities, [LatLonGrid](#) population, [LatLonGrid](#) nightTimeLights)
- static int[] [applyLinearModel](#) (double[][] pattern)
- static int[] [applyDecisionTree](#) (double[][] pattern)
- static int[] [applyNeuralNetwork](#) (double[][] pattern)
- static void [main](#) (String[] args) throws Exception

6.7.1 Detailed Description

Applies different data mining models to each city.

Author:

valliappa.lakshmanan

6.7.2 Member Function Documentation

6.7.2.1 static int [] edu.ou.asgbook.datamining.CityGdi-Models.applyDecisionTree (double *pattern*[][]) [static]

```

50                                                                 {
51         int[] result = new int[ pattern.length ];
52         for (int i=0; i < pattern.length; ++i){
53             double pop = pattern[i][0];
54             double light = pattern[i][1];
55             if (light < 48.91){
56                 if (light < 17.61){
57                     result[i] = 0;
58                 } else {
59                     result[i] = 1;
60                 }
61             } else {
62                 if ( light < 81.25 ){
63                     if ( pop >= 31.77 ){
64                         result[i] = 1;
65                     } else {
66                         result[i] = 2;
67                     }

```

```

68         } else {
69             if ( pop >= 105.7 ){
70                 result[i] = 2;
71             } else {
72                 result[i] = 4;
73             }
74         }
75     }
76 }
77 return result;
78 }

```

6.7.2.2 static int [] edu.ou.asgbook.datamining.CityGdiModels.applyLinearModel (double *pattern*[][]) [static]

```

42                                     {
43         int[] result = new int[ pattern.length ];
44         for (int i=0; i < pattern.length; ++i){
45             result[i] = (int) Math.round(0.003494 + 0.034444 * pattern[i][1] - 0.005992 * p
46         }
47         return result;
48     }

```

6.7.2.3 static int [] edu.ou.asgbook.datamining.CityGdiModels.applyNeuralNetwork (double *pattern*[][]) [static]

```

94                                     {
95         int[] result = new int[ pattern.length ];
96         for (int i=0; i < pattern.length; ++i){
97             result[i] = (int) Math.round(100 * probOfRichNN(pattern[i][0], pattern[i][1]));
98         }
99         return result;
100     }

```

6.7.2.4 static double [][] edu.ou.asgbook.datamining.CityGdiModels.findPatterns ([LabelResult](#) *cities*, [LatLonGrid](#) *population*, [LatLonGrid](#) *nightTimeLights*) [static]

```

29
30         // make sure that this is identical to GdiPattern.java
31         RegionProperty[] pop = RegionProperty.compute(cities, population);
32         RegionProperty[] lights = RegionProperty.compute(cities, nightTimeLights);
33
34         double[][] patterns = new double[pop.length][2];
35         for (int i=1; i < patterns.length; ++i){
36             patterns[i][0] = pop[i].getCval();

```

```

37         patterns[i][1] = lights[i].getCval();
38     }
39     return patterns;
40 }

```

6.7.2.5 static void edu.ou.asgbook.datamining.CityGdiModels.main (String[] args) throws Exception [static]

```

102                                     {
103     // create output directory
104     File out = OutputDirectory.getDefault("gdimodels");
105     final boolean SMALL = true;
106
107     // read input (crop to cover China)
108     LatLonGrid pop = GlobalPopulation.read(GlobalPopulation.WORLD);
109     if (SMALL){
110         pop = pop.crop(900, 6000, 800, 1600); // China mainly
111     }
112     KmlWriter.write(pop, out, "modelpop", PngWriter.createRandomColormap());
113
114     LatLonGrid nightTimeLights = NighttimeLights.read(NighttimeLights.WORLD).remapTo(pop);
115     KmlWriter.write(nightTimeLights, out, "modellights", PngWriter.createCoolToWarmColormap());
116
117     EnhancedWatershedSegmenter seg = new EnhancedWatershedSegmenter(10, 1, 600, 10, 5);
118     LabelResult allcities = seg.label(pop);
119     KmlWriter.write(allcities.label, out, "modelcities", PngWriter.createRandomColormap());
120
121     // compute gdi for each city
122     double[][] patterns = findPatterns(allcities, pop, nightTimeLights);
123     String[] models = {"linear", "tree", "nn" };
124     for (String model : models){
125         int[] modelresult = null;
126         if (model.equals("linear")){
127             modelresult = applyLinearModel(patterns);
128         } else if (model.equals("tree")){
129             modelresult = applyDecisionTree(patterns);
130         } else if (model.equals("nn")){
131             modelresult = applyNeuralNetwork(patterns);
132         }
133         LatLonGrid result = LatLonGrid.copyOf(allcities.label);
134         result.setMissing(WorldBankGDI.DevelopmentCategory.Unknown.ordinal());
135         result.fill( result.getMissing() );
136         for (int i=0; i < result.getNumLat(); ++i){
137             for (int j=0; j < result.getNumLon(); ++j){
138                 int cityno = allcities.label.getValue(i, j);
139                 if (cityno > 0 ){
140                     result.setValue(i, j, modelresult[cityno]);
141                 }
142             }
143         }
144         KmlWriter.write(result, out, model+"gdi", PngWriter.createCoolToWarmColormap());
145     }
146 }

```

6.8 edu.ou.asgbook.segmentation.Contiguity-EnhancedKMeansSegmenter Class Reference

Objects consist of pixels that are grown from initial centers using K-means.

Public Member Functions

- [ContiguityEnhancedKMeansSegmenter](#) (int min_thresh, int seed_value, int max_data_difference, int max_cluster_size)

KMeans is seeded from points > seed_value, so pass in a high enough value here Only pixels > min_thresh are eligible to be part of an object.

- List< [LabelResult](#) > [label](#) (LatLonGrid data)

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

Classes

- class [Cluster](#)

6.8.1 Detailed Description

Objects consist of pixels that are grown from initial centers using K-means.

Author:

v.lakshmanan

6.8.2 Constructor & Destructor Documentation

- 6.8.2.1 edu.ou.asgbook.segmentation.ContiguityEnhancedKMeans-Segmenter.ContiguityEnhancedKMeansSegmenter** (int *min_thresh*, int *seed_value*, int *max_data_difference*, int *max_cluster_size*)

KMeans is seeded from points > seed_value, so pass in a high enough value here Only pixels > min_thresh are eligible to be part of an object.

```
37
38     this.START_THRESH = seed_value;
39     this.MIN_THRESH = min_thresh;
40     this.MAX_DATA_DIFFERENCE = max_data_difference;
41     this.MAX_CLUSTER_SIZE = max_cluster_size;
42 }
```

6.8.3 Member Function Documentation

6.8.3.1 List<[LabelResult](#)> edu.ou.asgbook.segmentation.Contiguity-EnhancedKMeansSegmenter.label ([LatLonGrid](#) data)

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

.. All pixels > thresh are part of an object. Returns a [LabelResult](#) for each iteration, with the last one being the final result.

```
132                                     {
133     List<LabelResult> result = new ArrayList<LabelResult>();
134     final int nrows = data.getNumLat();
135     final int ncols = data.getNumLon();
136
137     // initialize based on simple thresholding at a high value
138     final ThresholdSegmenter seeder = new ThresholdSegmenter(START_THRESH);
139     LabelResult seed = seeder.label(data);
140     result.add(seed); // first one
141
142     // Start K-means
143     int iter = 1;
144     int n_changed = 0;
145     do {
146         // compute means
147         Cluster[] clusters = findClusters(data, seed);
148         // move pixels
149         LabelResult next = new LabelResult(LatLonGrid.copyOf(seed.label), seed.maxlabel);
150         n_changed = 0;
151         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
152             if ( data.getValue(i,j) != data.getMissing() && data.getValue(i,j) > MIN_THRESH ){
153                 int closest = findClosestCluster(data.getValue(i,j),i,j,seed.label, clusters);
154                 if (closest != seed.label.getValue(i,j)){
155                     // change the label to closest
156                     next.label.setValue(i, j, closest);
157                     ++n_changed;
158                 }
159             }
160         }
161         System.out.println("Changing " + n_changed + " at " + iter + " th iteration");
162         // for next step
163         seed = next;
164         result.add(seed);
165         ++iter;
166     }
```

```
166         } while (iter < MAX_ITER && n_changed > 0);
167         return result;
168     }
```

6.8.3.2 `static void edu.ou.asgbook.segmentation.ContiguityEnhanced-KMeansSegmenter.main (String[] args) throws Exception`

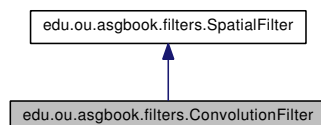
[static]

```
170                                     {
171         File out = OutputDirectory.getDefault("contigkmeans");
172
173         LatLonGrid grid = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalP
174         KmlWriter.write(grid, out, "orig", PngWriter.createCoolToWarmColormap());
175
176         // label image based on threshold
177         List<LabelResult> labels = new ContiguityEnhancedKMeansSegmenter(10,20,100,10).la
178         for (int i=0; i < labels.size(); ++i){
179             LatLonGrid label = labels.get(i).label;
180             KmlWriter.write(label, out, "label_" + i, PngWriter.createCoolToWarmColormap())
181         }
182     }
```

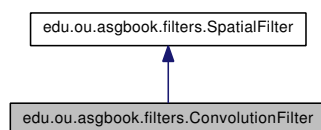
6.9 edu.ou.asgbook.filters.ConvolutionFilter Class Reference

Convolve an image by a window.

Inheritance diagram for edu.ou.asgbook.filters.ConvolutionFilter:



Collaboration diagram for edu.ou.asgbook.filters.ConvolutionFilter:



Public Member Functions

- [ConvolutionFilter](#) (double[][] coeffs)
- int [getFilterNumRows](#) ()
- int [getFilterNumCols](#) ()
- [LatLonGrid smooth](#) (final [LatLonGrid](#) input)
Uses weights, but only at non-missing pixels, and divides by the total weight.
- [LatLonGrid convolve](#) (final [LatLonGrid](#) input)
Uses the coefficients and returns the convolved value without dividing by sum of weights Use this for non-smoothing coefficients.
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)

Static Public Member Functions

- static double[][] [boxcar](#) (int numx, int numy)
- static double[][] [gauss](#) (int numx, int numy)
- static double[][] [gauss](#) (int numx, int numy, double sigmax, double sigmay)
- static void [main](#) (String[] args) throws Exception

6.9.1 Detailed Description

Convolve an image by a window.

Author:

Valliappa.Lakshmanan

6.9.2 Constructor & Destructor Documentation

6.9.2.1 `edu.ou.asgbook.filters.ConvolutionFilter.ConvolutionFilter (double coeffs[][])`

```

23                                     {
24         this.coeffs = coeffs;
25         if ( coeffs.length % 2 == 0 || coeffs[0].length % 2 == 0 ){
26             throw new IllegalArgumentException("Dimensions of coefficients array needs to be
27         }
28     }
```

6.9.3 Member Function Documentation

6.9.3.1 `static double [][] edu.ou.asgbook.filters.ConvolutionFilter.boxcar (int numx, int numy) [static]`

```

104                                     {
105         double[ ][ ] coeffs = new double[numx][numy];
106
107         double tot = numx * numy;
108         for (int i=0; i < coeffs.length; ++i){
109             for (int j=0; j < coeffs[0].length; ++j){
110                 coeffs[i][j] = 1 / tot;
111             }
112         }
113
114         return coeffs;
115     }
```

6.9.3.2 [LatLonGrid](#) `edu.ou.asgbook.filters.ConvolutionFilter.convolve (final LatLonGrid input)`

Uses the coefficients and returns the convolved value without dividing by sum of weights Use this for non-smoothing coefficients.

```

77                                     {
78         LatLonGrid output = LatLonGrid.copyOf(input);
79         output.fill(output.getMissing());
77                                     }
```



```

80         int[][] outData = output.getData();
81         int[][] inData = input.getData();
82         final int hx = coeffs.length / 2;
83         final int hy = coeffs[0].length / 2;
84         final int nx = output.getNumLat();
85         final int ny = output.getNumLon();
86         for (int i=hx; i < (nx-hx); ++i){
87             for (int j=hy; j < (ny-hy); ++j){
88                 double tot = 0;
89                 for (int m=-hx; m <= hx; ++m){
90                     for (int n=-hy; n <= hy; ++n){
91                         double coeff = coeffs[m+hx][n+hy];
92                         int inval = inData[i+m][j+n];
93                         if (inval != input.getMissing()){
94                             tot += inval*coeff;
95                         }
96                     }
97                 }
98                 outData[i][j] = (int) Math.round(tot);
99             }
100         }
101         return output;
102     }

```

6.9.3.3 Override [LatLonGrid](#) edu.ou.asgbook.filters.ConvolutionFilter.filter ([LatLonGrid](#) input)

```

166                                     {
167         return convolve(input);
168     }

```

6.9.3.4 static double [][] edu.ou.asgbook.filters.ConvolutionFilter.gauss (int numx, int numy, double sigmax, double sigmay) [static]

```

121                                     {
122         double[][] coeffs = new double[numx][numy];
123
124         for (int i=0; i < coeffs.length; ++i){
125             for (int j=0; j < coeffs[0].length; ++j){
126                 double x = (i - coeffs.length/2.0)/sigmax;
127                 double y = (j - coeffs[0].length/2.0)/sigmay;
128                 coeffs[i][j] = Math.exp(-(x*x + y*y));
129             }
130         }
131
132         return coeffs;
133     }

```

6.9.3.5 static double [][] edu.ou.asgbook.filters.ConvolutionFilter.gauss (int numx, int numy) [static]

```
117                                     {
118     return gauss(numx, numy, numx/6.0, numy/6.0); // 3-sigma on either side
119 }
```

6.9.3.6 int edu.ou.asgbook.filters.ConvolutionFilter.getFilterNumCols ()

```
34                                     {
35     return coeffs[0].length;
36 }
```

6.9.3.7 int edu.ou.asgbook.filters.ConvolutionFilter.getFilterNumRows ()

```
30                                     {
31     return coeffs.length;
32 }
```

6.9.3.8 static void edu.ou.asgbook.filters.ConvolutionFilter.main (String[] args) throws Exception [static]

```
135                                     {
136     // create output directory
137     File out = OutputDirectory.getDefault("convolve");
138
139     // read input
140     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new G
141     KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
142
143     // average
144     {
145         ConvolutionFilter filter = new ConvolutionFilter(ConvolutionFilter.boxcar(3, 3));
146         LatLonGrid sm = filter.smooth(popdensity);
147         KmlWriter.write(sm, out, "boxcar1", PngWriter.createCoolToWarmColormap());
148     }
149
150     // boxcar
151     {
152         ConvolutionFilter filter = new ConvolutionFilter(ConvolutionFilter.boxcar(5, 5));
153         LatLonGrid sm = filter.smooth(popdensity);
154         KmlWriter.write(sm, out, "boxcar", PngWriter.createCoolToWarmColormap());
155     }
156
157     // gauss
158     {
159         ConvolutionFilter filter = new ConvolutionFilter(ConvolutionFilter.gauss(11, 11));
160         LatLonGrid sm = filter.smooth(popdensity);
161         KmlWriter.write(sm, out, "gauss", PngWriter.createCoolToWarmColormap());
162     }
163 }
```

```

162         }
163     }

```

6.9.3.9 [LatLonGrid](#) edu.ou.asgbook.filters.ConvolutionFilter.smooth (final [LatLonGrid](#) input)

Uses weights, but only at non-missing pixels, and divides by the total weight.

Use this for smoothing

```

42         {
43         LatLonGrid output = LatLonGrid.copyOf(input);
44         output.fill(output.getMissing());
45         int[][] outData = output.getData();
46         int[][] inData = input.getData();
47         final int hx = coeffs.length / 2;
48         final int hy = coeffs[0].length / 2;
49         final int nx = output.getNumLat();
50         final int ny = output.getNumLon();
51         for (int i=hx; i < (nx-hx); ++i){
52             for (int j=hy; j < (ny-hy); ++j){
53                 double tot = 0;
54                 double wt = 0;
55                 for (int m=-hx; m <= hx; ++m){
56                     for (int n=-hy; n <= hy; ++n){
57                         double coeff = coeffs[m+hx][n+hy];
58                         int inval = inData[i+m][j+n];
59                         if (inval != input.getMissing()){
60                             tot += inval*coeff;
61                             wt += coeff;
62                         }
63                     }
64                 }
65                 if ( wt > 0 ){
66                     outData[i][j] = (int)( Math.round(tot / wt) );
67                 }
68             }
69         }
70         return output;
71     }

```

6.10 edu.ou.asgbook.dataset.CountryPolygons Class Reference

Reads country-by-country coordinates from a KML placemarks file.

Static Public Member Functions

- static `Country[] readKml` (File file) throws Exception
reads data from a File.
- static `LatLonGrid readGrid` (File file) throws Exception
- static `LatLonGrid asLatLonGrid` (Country[] countries, double latres, double lonres)
- static void `main` (String[] args) throws Exception

Static Public Attributes

- static File `WORLD_KML` = new File("data/countries/countries_world.kml")
- static File `WORLD_GRID` = new File("data/countries/countries_world.txt.gz")

Classes

- class `Country`

6.10.1 Detailed Description

Reads country-by-country coordinates from a KML placemarks file.

Author:

v.lakshmanan

6.10.2 Member Function Documentation

6.10.2.1 static `LatLonGrid` edu.ou.asgbook.dataset.CountryPolygons.asLatLonGrid (Country[] countries, double latres, double lonres) [static]

```

121
122         int nrows = (int) Math.round(180 / latres);
123         int ncols = (int) Math.round(360 / lonres);

```

```

124         LatLon nwCorner = new LatLon(90,-180);
125         LatLonGrid result = new LatLonGrid(nrows, ncols, -1, nwCorner, latres, lonres);
126         for (int i=0; i < nrows; ++i){
127             for (int j=0; j < ncols; ++j){
128                 LatLon pt = result.getLocation(i, j);
129                 result.setValue(i,j, result.getMissing());
130                 for (int c = 0; c < countries.length; ++c){
131                     if (countries[c].contains(pt)){
132                         result.setValue(i, j, c);
133                         break;
134                     }
135                 }
136             }
137             System.out.println("row " + i + " computed.");
138         }
139         return result;
140     }

```

6.10.2.2 static void edu.ou.asgbook.dataset.CountryPolygons.main (String[] args) throws Exception [static]

```

142         {
143             CountryPolygons.Country[] countries = CountryPolygons.readKml(CountryPolygons.WORLD_KML);
144             for (CountryPolygons.Country c : countries){
145                 System.out.println(c);
146             }
147
148             List<LatLon> cities = new ArrayList<LatLon>();
149             cities.add(new LatLon(35, -97.1)); // Norman, Oklahoma
150             cities.add(new LatLon(40,33)); // Istanbul, Turkey
151             cities.add(new LatLon(-34,151)); // Sydney, Australia
152             cities.add(new LatLon(-23.5,-46.5)); // Rio, Brazil
153             for (LatLon city : cities){
154                 System.out.println("Looking for " + city);
155                 for (CountryPolygons.Country c : countries){
156                     if (c.contains(city)){
157                         System.out.println(city + " is in " + c);
158                         break;
159                     }
160                 }
161             }
162             System.out.println("Finished place search using country list");
163
164             File out = OutputDirectory.getDefault("countries");
165             LatLonGrid grid = asLatLonGrid(countries, 0.1, 0.1);
166             for (LatLon city : cities){
167                 int country = grid.getValue(city);
168                 if (country >= 0){
169                     System.out.println("Location " + city + " is in " + countries[country]);
170                 } else {
171                     System.out.println("Location " + city + " is unclaimed");
172                 }
173             }
174
175             KmlWriter.write(grid, out, "countries", PngWriter.createRandomColormap());

```

```

176         EsriGrid.write(grid, out, "countries.txt.gz");
177         EsriGrid.write(grid, CountryPolygons.WORLD_GRID);
178
179
180     /*      // combine with GDI ...
181     List<CountryPolygons.Country> notfound = new ArrayList<CountryPolygons.Country>();
182     WorldBankGDI.Lookup gdicountries = WorldBankGDI.readAsMap(WorldBankGDI.WORLD);
183     for (CountryPolygons.Country c : countries){
184         WorldBankGDI.Country match = gdicountries.get(c.name);
185         System.out.println(c + " " + match);
186         if ( match == null ){
187             notfound.add(c);
188         }
189     }
190     for (CountryPolygons.Country c : notfound){
191         System.out.println("Not found: " + c);
192     }*/
193 }

```

6.10.2.3 static [LatLonGrid](#) edu.ou.asgbook.dataset.CountryPolygons.readGrid (File *file*) throws Exception [static]

```

84                                     {
85         return EsriGrid.read(file, new LinearScaling(1));
86     }

```

6.10.2.4 static Country [] edu.ou.asgbook.dataset.CountryPolygons.readKml (File *file*) throws Exception [static]

reads data from a File.

The File can be gzipped or uncompressed.

```

74                                     {
75         InputStream f = null;
76         System.out.println("Reading " + file.getAbsolutePath());
77         f = new FileInputStream(file);
78         if (file.getAbsolutePath().endsWith(".gz")) {
79             f = new GZIPInputStream(f);
80         }
81         return readKml(f);
82     }

```

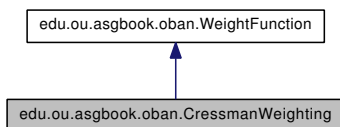
6.10.3 Member Data Documentation

- 6.10.3.1 File `edu.ou.asgbook.dataset.CountryPolygons.WORLD_GRID` = new
File("data/countries/countries_world.txt.gz") [static]
- 6.10.3.2 File `edu.ou.asgbook.dataset.CountryPolygons.WORLD_KML` = new
File("data/countries/countries_world.kml") [static]

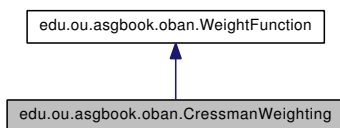
6.11 edu.ou.asgbook.oban.CressmanWeighting Class Reference

An interpolation method that uses $1/r^2$.

Inheritance diagram for edu.ou.asgbook.oban.CressmanWeighting:



Collaboration diagram for edu.ou.asgbook.oban.CressmanWeighting:



Public Member Functions

- [CressmanWeighting](#) (double radiusOfInfluence)
- Override double [computeWt](#) (double latdist, double londist)

Subclasses implement a weighting function.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.11.1 Detailed Description

An interpolation method that uses $1/r^2$.

Author:

Valliappa.Lakshmanan

6.11.2 Constructor & Destructor Documentation

6.11.2.1 edu.ou.asgbook.oban.CressmanWeighting.CressmanWeighting (double *radiusOfInfluence*)

Parameters:

radiusOfInfluence Set extent of influence in degrees

```

27                                     {
28         this.R2 = radiusOfInfluence * radiusOfInfluence;
29     }
```

6.11.3 Member Function Documentation

6.11.3.1 Override double edu.ou.asgbook.oban.CressmanWeighting.computeWt (double *latdist*, double *londist*) [virtual]

Subclasses implement a weighting function.

If -ve value is returned, then the point will be considered too far away and not used in weighting.

Implements [edu.ou.asgbook.oban.WeightFunction](#).

```

32                                     {
33         double r2 = latdist * latdist + londist * londist;
34         if ( r2 > R2 ){
35             return INVALID_WEIGHT;
36         }
37         double factor = r2/R2;
38         return (1 - factor)/(1 + factor);
39     }
```

6.11.3.2 static void edu.ou.asgbook.oban.CressmanWeighting.main (String[] *args*) throws Exception [static]

```

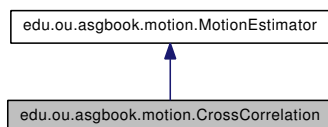
41                                     {
42         PointObservations data = DailyRainfall.read(DailyRainfall.TN_Oct2010);
43
44         double meansep = ObjectiveAnalysisUtils.computeMeanDistance(data);
45         System.out.println("Objectively analyzing " + data.getPoints().length + " pts with a mean separ
46         WeightFunction wtFunc = new CressmanWeighting(3*meansep);
47         WeightedAverage analyzer = new WeightedAverage(wtFunc, 0.01, 0.01, 1);
48
49         long startTime = System.nanoTime();
50         final int numPasses = 2;
51         LatLonGrid grid = analyzer.analyze(data, numPasses, 0, data.getMaxValue());
```

```
52         System.out.println("Took " + (System.nanoTime() - startTime)/(1000*1000.0*1000) + "  
53  
54         // write output  
55         File out = OutputDirectory.getDefault("cressman");  
56         KmlWriter.write(grid, out, "Precip24H", PngWriter.createCoolToWarmColormap());  
57     }
```

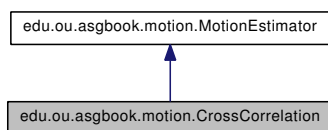
6.12 edu.ou.asgbook.motion.CrossCorrelation Class Reference

Estimates motion using cross-correlation.

Inheritance diagram for edu.ou.asgbook.motion.CrossCorrelation:



Collaboration diagram for edu.ou.asgbook.motion.CrossCorrelation:



Public Member Functions

- [CrossCorrelation](#) (int est_halfsize_x, int est_halfsize_y, int maxmotion_x, int maxmotion_y)
Pass size of window to estimate motion of, and the maximum movement in the two directions.
- Override Pair< [LatLonGrid](#), [LatLonGrid](#) > [compute](#) ([LatLonGrid](#) data0, [LatLonGrid](#) data1, File outdir)
returns motion in the two directions.

Static Public Member Functions

- static void [test](#) () throws Exception
- static void [main](#) (String[] args) throws Exception

6.12.1 Detailed Description

Estimates motion using cross-correlation.

Author:

v.lakshmanan

6.12.2 Constructor & Destructor Documentation**6.12.2.1 edu.ou.asgbook.motion.CrossCorrelation.CrossCorrelation (int *est_halfsize_x*, int *est_halfsize_y*, int *maxmotion_x*, int *maxmotion_y*)**

Pass size of window to estimate motion of, and the maximum movement in the two directions.

Parameters:*est_halfsize_x**est_halfsize_y*

```

40                                     {
41         super();
42         EST_HALFSIZE_NS = est_halfsize_x;
43         EST_HALFSIZE_EW = est_halfsize_y;
44         MAX_U = maxmotion_x;
45         MAX_V = maxmotion_y;
46         MIN_FILL_PIXELS = (int) Math.round(MIN_FILL_RATIO * (2*EST_HALFSIZE_NS+1) * (2*EST_
47     }
```

6.12.3 Member Function Documentation**6.12.3.1 Override Pair<LatLonGrid, LatLonGrid>
edu.ou.asgbook.motion.CrossCorrelation.compute (LatLonGrid *data0*,
LatLonGrid *data1*, File *outdir*)**

returns motion in the two directions.

The first one is north to south and the second one is east to west. The data is aligned to second time frame. The output dir is used for intermediate products and may be null.

Implements [edu.ou.asgbook.motion.MotionEstimator](#).

```

50
51         // Grids we need. initialize all of them at zero
52         final int nrows = data1.getNumLat();
53         final int ncols = data1.getNumLon();
54         LatLonGrid u = new LatLonGrid(nrows, ncols, 0, data1.getNwCorner(), data1.getLatRes
55         LatLonGrid v = LatLonGrid.copyOf(u);
56
57         System.out.println("Computing u,v using xcorr on " + nrows + "x" + ncols + " image;
58
59         // compute u,v for every pixel
```

```

60     double meanu = 0;
61     double meanv = 0;
62     int nestimates = 0;
63     for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
64         // at pixel, search best match for entire neighborhood
65         // best_m, best_n are not changed from default unless < error_ratio
66         double lse = MAX_ERROR_RATIO;
67         int best_m = 0;
68         int best_n = 0;
69         for (int m=-MAX_U; m <= MAX_U; ++m){
70             for (int n=-MAX_V; n <= MAX_V; ++n){
71                 double error = compute_error(data0, data1, i, j, m, n);
72                 if ( error < lse ){
73                     lse = error;
74                     best_m = m;
75                     best_n = n;
76                 }
77             }
78         }
79         u.setValue(i, j, best_m);
80         v.setValue(i, j, best_n);
81
82         if ( lse != MAX_ERROR_RATIO ){
83             meanu += best_m;
84             meanv += best_n;
85             ++nestimates;
86         }
87
88         if ( i%10 == 0 && j == 0){
89             System.out.println( (100*i)/nrows + "% of pixels complete.");
90         }
91     }
92
93     System.out.println("Mean motion vector: u=" + meanu/(nestimates) + " v=" + meanv/(nestimates));
94     return new Pair<LatLonGrid, LatLonGrid>(u, v);
95 }

```

6.12.3.2 static void edu.ou.asgbook.motion.CrossCorrelation.main (String[] args) throws Exception [static]

```

141                                     {
142         // test();
143         // create output directory
144         File out = OutputDirectory.getDefault("xcorr");
145
146         // read
147         File f = new File("data/seviri");
148         Pair<LatLonGrid, Date>[] grids = SeviriInfraredTemperature.readAll(f);
149
150         // do alg
151         CrossCorrelation alg = new CrossCorrelation(3, 3, 5, 5);
152         Pair<LatLonGrid, LatLonGrid> motion = alg.compute(grids[0].first, grids[1].first, out);
153
154         // write
155         KmlWriter.write(motion.first, out, "xcorr_u", PngWriter.createCoolToWarmColormap());

```

```
156         KmlWriter.write(motion.second, out, "xcorr_v", PngWriter.createCoolToWarmColormap());
157
158         // align and compute difference
159         LatLonGrid diff = new AlignAndDifference().compute(grid[0].first, grid[1].first,
160         KmlWriter.write(diff, out, "xcorr_diff", PngWriter.createCoolToWarmColormap());
161     }
```

6.12.3.3 **static void edu.ou.asgbook.motion.CrossCorrelation.test () throws Exception** [static]

```
125         {
126             // because the alignment doesn't really check lat-lon extents,
127             // cropping from offset corners will look like translation ...
128             LatLonGrid conus = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
129             LatLonGrid[] grids = new LatLonGrid[2];
130             grids[0] = conus.crop(900, 2500, 256, 256);
131             grids[0].setMissing(0);
132             int motx = -2; int moty = -3;
133             grids[1] = conus.crop(900-motx, 2500-moty, 256, 256);
134             grids[1].setMissing(0);
135             CrossCorrelation alg = new CrossCorrelation(5,5,Math.abs(2*motx),Math.abs(2*moty));
136             alg.compute(grids[0], grids[1], null);
137
138             System.exit(0);
139     }
```

6.13 edu.ou.asgbook.histogram.Cumulative-DistributionFunction Class Reference

Forms a CDF from a [Histogram](#).

Public Member Functions

- [CumulativeDistributionFunction](#) ([Histogram](#) hist)
- int [getMin](#) ()
- int [getIncr](#) ()
- float[] [getProb](#) ()
- Override String [toString](#) ()

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.13.1 Detailed Description

Forms a CDF from a [Histogram](#).

Author:

v.lakshmanan

6.13.2 Constructor & Destructor Documentation

6.13.2.1 edu.ou.asgbook.histogram.CumulativeDistribution-Function.CumulativeDistributionFunction ([Histogram](#) *hist*)

```

24                                     {
25         this.min = hist.getMin();
26         this.incr = hist.getIncr();
27         prob = new float[hist.getHist().length];
28         int tot = 0;
29         for (int i=0; i < hist.getHist().length; ++i){
30             tot += hist.getHist()[i];
31         }
32         if ( tot == 0 ) return;
33
34         int sofar = 0;
35         for (int i=0; i < hist.getHist().length; ++i){
36             sofar += hist.getHist()[i];

```

```

37         prob[i] = sofar / (float) tot;
38     }
39 }

```

6.13.3 Member Function Documentation

6.13.3.1 `int edu.ou.asgbook.histogram.CumulativeDistributionFunction.getIncr()`

```

45     {
46         return incr;
47     }

```

6.13.3.2 `int edu.ou.asgbook.histogram.CumulativeDistributionFunction.getMin()`

```

41     {
42         return min;
43     }

```

6.13.3.3 `float [] edu.ou.asgbook.histogram.CumulativeDistributionFunction.getProb()`

```

49     {
50         return prob;
51     }

```

6.13.3.4 `static void edu.ou.asgbook.histogram.CumulativeDistributionFunction.main (String[] args) throws Exception` [static]

```

69     {
70         // create output directory
71         File outdir = OutputDirectory.getDefault("cdf");
72
73         // read input
74         LatLonGrid conus = SurfaceAlbedo.read(SurfaceAlbedo.CONUS, 100);
75
76         // find histogram
77         final int MIN = 0;
78         final int MAX = 30;
79         for (int incr=1; incr < 10; incr += 2){
80             Histogram hist = new Histogram(MIN, incr, (MAX-MIN)/incr );
81             hist.update(conus);
82             CumulativeDistributionFunction cdf = new CumulativeDistributionFunction(hist);
83             System.out.println("INCR=" + incr + " nbins=" + cdf.prob.length);

```


6.13 edu.ou.asgbook.histogram.CumulativeDistributionFunction Class Reference

```
84         System.out.println(cdf);
85         String filename = outdir.getAbsolutePath() + "/cdf_" + incr + ".txt";
86         PrintWriter writer = new PrintWriter(new FileWriter(filename));
87         writer.println(cdf);
88         writer.close();
89         System.out.println("Wrote to " + filename);
90     }
91 }
```

6.13.3.5 Override String edu.ou.asgbook.histogram.CumulativeDistributionFunction.toString ()

```
54     {
55         StringBuilder sb = new StringBuilder();
56         for (int i=0; i < prob.length; ++i){
57             int sval = min + i * incr;
58             int eval = sval + incr;
59             sb.append(sval);
60             sb.append(" ");
61             sb.append(eval);
62             sb.append(" ");
63             sb.append(prob[i]);
64             sb.append("\n");
65         }
66         return sb.toString();
67     }
```

6.14 edu.ou.asgbook.dataset.DailyRainfall Class Reference

Reads the ASCII precipitation data available at <http://madis-data.noaa.gov/public/hydrodumpguest.html>.

Static Public Member Functions

- static [PointObservations read](#) (File file) throws IOException
- static void [main](#) (String[] args) throws Exception

Static Public Attributes

- static final File [TN_Oct2010](#) = new File("data/madishydro/tn_oct2010_-24hr.txt")

Package Functions

- [SuppressWarnings](#) ("unused") public static [PointObservations read](#)(Reader r) throws IOException

6.14.1 Detailed Description

Reads the ASCII precipitation data available at <http://madis-data.noaa.gov/public/hydrodumpguest.html>.

Author:

Valliappa.Lakshmanan

6.14.2 Member Function Documentation

6.14.2.1 static void edu.ou.asgbook.dataset.DailyRainfall.main (String[] args) throws Exception [static]

```

62                                     {
63         PointObservations data = DailyRainfall.read(DailyRainfall.TN_Oct2010);
64         for (int i=0; i < data.getPoints().length; ++i){
65             System.out.println(data.getPoints()[i]);
66         }
67     }
```

6.14.2.2 static [PointObservations](#) edu.ou.asgbook.dataset.DailyRainfall.read (File *file*) throws IOException [static]

```

29                                     {
30         Reader f = null;
31         if (file.getAbsolutePath().endsWith(".gz")) {
32             f = new InputStreamReader(new GZIPInputStream(new FileInputStream(
33                 file)));
34         } else {
35             f = new FileReader(file);
36         }
37         return read(f);
38     }

```

6.14.2.3 edu.ou.asgbook.dataset.DailyRainfall.SuppressWarnings ("unused") throws IOException [package]

```

41                                     {
42         Scanner s = new Scanner(r);
43         List<PointObservations.ObservationPoint> result = new ArrayList<PointObservations.ObservationPoint>();
44
45         final int FACTOR = 1000;
46         final int MISSING = -9999 * FACTOR;
47         s.nextLine(); // header
48         while (s.hasNext()){
49             String station = s.next();
50             String date = s.next();
51             String time = s.next();
52             int precip = (int) Math.round( s.nextDouble() * FACTOR );
53             double lat = s.nextDouble();
54             double lon = s.nextDouble();
55             result.add(new PointObservations.ObservationPoint(lat,lon,precip));
56         }
57
58         PointObservations.ObservationPoint[] pts = result.toArray(new PointObservations.ObservationPoint[result.size()]);
59         return new PointObservations(pts, MISSING);
60     }

```

6.14.3 Member Data Documentation

6.14.3.1 final File [edu.ou.asgbook.dataset.DailyRainfall.TN_Oct2010](#) = new File("data/madishydro/tn_oct2010_24hr.txt") [static]

6.15 edu.ou.asgbook.rbf.DataSimulator Class Reference

Simulates RBF data to be fit.

Static Public Member Functions

- static [LatLonGrid](#) [simulateData](#) ([Pixel](#)[] centers, double[] sigmax, double[] sigmay, int nrows, int ncols)
- static void [simulateData](#) ([LatLonGrid](#) result, [Pixel](#)[] centers, double[] sigmax, double[] sigmay)
- static void [main](#) (String[] args) throws Exception

6.15.1 Detailed Description

Simulates RBF data to be fit.

Author:

v.lakshmanan

6.15.2 Member Function Documentation

6.15.2.1 static void edu.ou.asgbook.rbf.DataSimulator.main (String[] args) throws Exception [static]

```

50                                     {
51         int nrows = 100;
52         int ncols = 100;
53         Pixel[] centers = new Pixel[]{ new Pixel(nrows/4,ncols/3,20), new Pixel(nrows/3,ncols/4,10) };
54         double[] sigmax = new double[] { nrows/12, ncols/8 };
55         double[] sigmay = new double[] { nrows/8, ncols/12 };
56         LatLonGrid m = DataSimulator.simulateData(centers, sigmax, sigmay, nrows, ncols);
57
58         // write out as image, for viewing
59         File out = OutputDirectory.getDefault("rbf");
60         KmlWriter.write(m, out, "simulated", PngWriter.createCoolToWarmColormap());
61
62         System.out.println("Done");
63     }
```

6.15.2.2 static void edu.ou.asgbook.rbf.DataSimulator.simulateData ([LatLonGrid](#) result, [Pixel](#)[] centers, double[] sigmax, double[] sigmay) [static]

```

31
32     for (int i=0; i < result.getNumLat(); ++i) for (int j=0; j < result.getNumLon(); ++j){
33         double tot = 0;
34         for (int k=0; k < centers.length; ++k){
35             double xdist = i - centers[k].getX();
36             double ydist = j - centers[k].getY();
37             double xnorm = (xdist*xdist) / (sigmax[k] * sigmax[k]);
38             double ynorm = (ydist*ydist) / (sigmay[k] * sigmay[k]);
39             double wt = Math.exp(-(xnorm + ynorm));
40             tot += wt * centers[k].getValue();
41         }
42         if ( tot > 0 ){
43             result.setValue(i, j, (int) Math.round(tot));
44         } else {
45             result.setValue(i, j, 0);
46         }
47     }
48 }
```

6.15.2.3 static [LatLonGrid](#) edu.ou.asgbook.rbf.DataSimulator.simulateData ([Pixel](#)[] centers, double[] sigmax, double[] sigmay, int nrows, int ncols) [static]

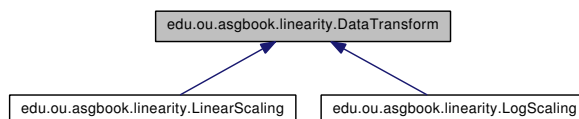
```

22
23     LatLon nwCorner = new LatLon(38, -100);
24     double latres = 0.01;
25     double lonres = 0.01;
26     LatLonGrid result = new LatLonGrid(nrows, ncols, -999, nwCorner, latres, lonres );
27     simulateData(result, centers, sigmax, sigmay);
28     return result;
29 }
```

6.16 edu.ou.asgbook.linearity.DataTransform Class Reference

Transform pixel values, usually to meet linearity requirements.

Inheritance diagram for edu.ou.asgbook.linearity.DataTransform:



Public Member Functions

- int [transformAndRoundoff](#) (double value)
- abstract double [transform](#) (double value)
- abstract double [inverse](#) (double value)

6.16.1 Detailed Description

Transform pixel values, usually to meet linearity requirements.

Author:

valliappa.lakshmanan

6.16.2 Member Function Documentation

6.16.2.1 abstract double edu.ou.asgbook.linearity.DataTransform.inverse (double value) [pure virtual]

Implemented in [edu.ou.asgbook.linearity.LinearScaling](#), and [edu.ou.asgbook.linearity.LogScaling](#).

6.16.2.2 abstract double edu.ou.asgbook.linearity.DataTransform.transform (double value) [pure virtual]

Implemented in [edu.ou.asgbook.linearity.LinearScaling](#), and [edu.ou.asgbook.linearity.LogScaling](#).

**6.16.2.3 int edu.ou.asgbook.linearity.DataTransform.transformAndRoundoff
(double *value*)**

```
14      {  
15      return (int) Math.round(transform(value));  
16      }
```

6.17 edu.ou.asgbook.motion.Differencer Class Reference

Just computes a pixel-by-pixel difference.

Public Member Functions

- [LatLonGrid](#) `compute` ([LatLonGrid](#) data0, [LatLonGrid](#) data1)

Static Public Member Functions

- static void `main` (String[] args) throws Exception

6.17.1 Detailed Description

Just computes a pixel-by-pixel difference.

This is really not a motion estimation, but is there just to show what happens when you do so.

Author:

v.lakshmanan

6.17.2 Member Function Documentation

6.17.2.1 [LatLonGrid](#) `edu.ou.asgbook.motion.Differencer.compute` ([LatLonGrid](#) data0, [LatLonGrid](#) data1)

```

26                                     {
27         LatLonGrid result = LatLonGrid.copyOf(data1);
28         for (int i=0; i < result.getNumLat(); ++i){
29             for (int j=0; j < result.getNumLon(); ++j){
30                 int diff = data1.getValue(i, j) - data0.getValue(i, j);
31                 result.setValue(i, j, diff);
32             }
33         }
34         return result;
35     }
```

6.17.2.2 static void `edu.ou.asgbook.motion.Differencer.main` (String[] args) throws Exception [static]

```

37                                     {
```

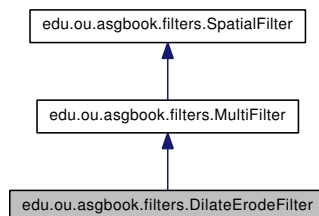


```
38     File out = OutputDirectory.getDefault("difference");
39
40     // seviri
41     File f = new File("data/seviri");
42     Pair<LatLonGrid,Date>[] grids = SeviriInfraredTemperature.readAll(f);
43     KmlWriter.write(grids[0].first, out, "ir0", PngWriter.createCoolToWarmColormap());
44     KmlWriter.write(grids[1].first, out, "ir1", PngWriter.createCoolToWarmColormap());
45     LatLonGrid diff = new Differencer().compute(grids[0].first, grids[1].first);
46     KmlWriter.write(diff, out, "irdiff", PngWriter.createCoolToWarmColormap());
47
48     // popdensity
49     DataTransform[] transforms = {new GlobalPopulation.LinearScaling(), new GlobalPopulation.LogScaling()};
50     String[] prefix = {"popdensity", "logpopdensity"};
51     for (int i=0; i < transforms.length; ++i){
52         LatLonGrid popdensity0 = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA1990, transforms[i]);
53         LatLonGrid popdensity1 = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, transforms[i]);
54         KmlWriter.write(popdensity0, out, prefix[i]+"0", PngWriter.createCoolToWarmColormap());
55         KmlWriter.write(popdensity1, out, prefix[i]+"1", PngWriter.createCoolToWarmColormap());
56         diff = new Differencer().compute(popdensity0, popdensity1);
57         KmlWriter.write(diff, out, prefix[i]+"diff", PngWriter.createCoolToWarmColormap());
58     }
59 }
```

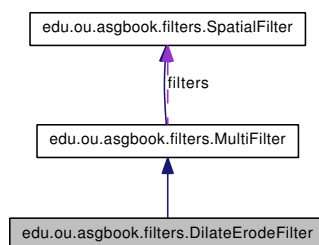
6.18 edu.ou.asgbook.filters.DilateErodeFilter Class Reference

Carries out paired dilation followed by erosion for filling in holes.

Inheritance diagram for edu.ou.asgbook.filters.DilateErodeFilter:



Collaboration diagram for edu.ou.asgbook.filters.DilateErodeFilter:



Public Member Functions

- [DilateErodeFilter](#) (int halfSize, int numTimes)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.18.1 Detailed Description

Carries out paired dilation followed by erosion for filling in holes.

Author:

Valliappa.Lakshmanan

6.18.2 Constructor & Destructor Documentation

6.18.2.1 edu.ou.asgbook.filters.DilateErodeFilter.DilateErodeFilter (int *halfSize*, int *numTimes*)

```
21                                     {
22     super(new SpatialFilter[]{
23         new DilationFilter(halfSize), new ErosionFilter(halfSize)
24     }, numTimes);
25 }
```

6.18.3 Member Function Documentation

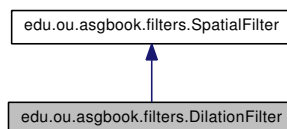
6.18.3.1 static void edu.ou.asgbook.filters.DilateErodeFilter.main (String[] *args*) throws Exception [static]

```
27                                     {
28     // create output directory
29     File out = OutputDirectory.getDefault("dilateerode");
30
31     // read input
32     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulati
33     KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
34
35     // erode
36     LatLonGrid erode1 = new DilateErodeFilter(1,1).filter(popdensity);
37     KmlWriter.write(erode1, out, "dilateerode_3_1", PngWriter.createCoolToWarmColormap());
38     LatLonGrid erode3 = new DilateErodeFilter(1,3).filter(popdensity);
39     KmlWriter.write(erode3, out, "dilateerode_3_3", PngWriter.createCoolToWarmColormap());
40     LatLonGrid erode5 = new DilateErodeFilter(2,3).filter(popdensity);
41     KmlWriter.write(erode5, out, "dilateerode_5_3", PngWriter.createCoolToWarmColormap());
42 }
```

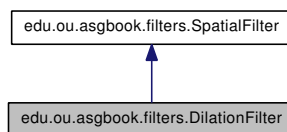
6.19 edu.ou.asgbook.filters.DilationFilter Class Reference

Expands entities by taking a local maximum.

Inheritance diagram for edu.ou.asgbook.filters.DilationFilter:



Collaboration diagram for edu.ou.asgbook.filters.DilationFilter:



Public Member Functions

- [DilationFilter](#) (int halfSize)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)
- [LatLonGrid dilate](#) (final [LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.19.1 Detailed Description

Expands entities by taking a local maximum.

Author:

Valliappa.Lakshmanan

6.19.2 Constructor & Destructor Documentation

6.19.2.1 edu.ou.asgbook.filters.DilationFilter.DilationFilter (int *halfSize*)

```

23                                     {
24         this.halfSize = halfSize;
25     }
```

6.19.3 Member Function Documentation

6.19.3.1 [LatLonGrid](#) edu.ou.asgbook.filters.DilationFilter.dilate (final [LatLonGrid](#) *input*)

```

32                                     {
33         LatLonGrid output = LatLonGrid.copyOf(input);
34         output.fill(output.getMissing());
35         int[][] outData = output.getData();
36         int[][] inData = input.getData();
37         int hx = halfSize;
38         int hy = halfSize;
39         int nx = inData.length;
40         int ny = inData[0].length;
41         for (int i=hx; i < (nx-hx); ++i){
42             for (int j=hy; j < (ny-hy); ++j){
43                 int max = input.getMissing();
44                 boolean set = false;
45                 for (int m=-hx; m <= hx; ++m){
46                     for (int n=-hy; n <= hy; ++n){
47                         int inval = inData[i+m][j+n];
48                         if (inval != input.getMissing()){
49                             if ( !set || inval > max ){
50                                 max = inval;
51                                 set = true;
52                             }
53                         }
54                     }
55                 }
56                 if ( set ){
57                     outData[i][j] = max;
58                 }
59             }
60         }
61         return output;
62     }
```

6.19.3.2 Override [LatLonGrid](#) edu.ou.asgbook.filters.DilationFilter.filter ([LatLonGrid](#) *input*)

```

28                                     {
29         return dilate(input);
30     }
```

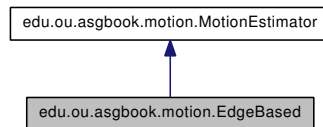
**6.19.3.3 static void edu.ou.asgbook.filters.DilationFilter.main (String[] args)
throws Exception [static]**

```
64                                     {
65     // create output directory
66     File out = OutputDirectory.getDefault("dilate");
67
68     // read input
69     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GL
70     KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
71
72     // dilate
73     LatLonGrid dilate1 = new DilationFilter(1).dilate(popdensity);
74     KmlWriter.write(dilate1, out, "dilate_3", PngWriter.createCoolToWarmColormap());
75     LatLonGrid dilate3 = new DilationFilter(3).dilate(popdensity);
76     KmlWriter.write(dilate3, out, "dilate_7", PngWriter.createCoolToWarmColormap());
77     LatLonGrid dilate5 = new DilationFilter(5).dilate(popdensity);
78     KmlWriter.write(dilate5, out, "dilate_11", PngWriter.createCoolToWarmColormap());
79 }
```

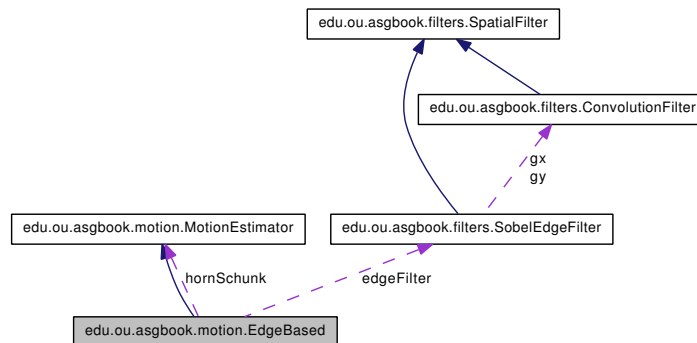
6.20 edu.ou.asgbook.motion.EdgeBased Class Reference

Estimates motion based on the displacement of edges.

Inheritance diagram for edu.ou.asgbook.motion.EdgeBased:



Collaboration diagram for edu.ou.asgbook.motion.EdgeBased:



Public Member Functions

- Override `Pair< LatLonGrid, LatLonGrid > compute (LatLonGrid data0, LatLonGrid data1, File outdir)`
returns motion in the two directions.

Static Public Member Functions

- static void `main (String[] args)` throws Exception

6.20.1 Detailed Description

Estimates motion based on the displacement of edges.

Author:

v.lakshmanan

6.20.2 Member Function Documentation**6.20.2.1 Override Pair<LatLonGrid, LatLonGrid>
edu.ou.asgbook.motion.EdgeBased.compute (LatLonGrid data0,
LatLonGrid data1, File outdir)**

returns motion in the two directions.

The first one is north to south and the second one is east to west. The data is aligned to second time frame. The output dir is used for intermediate products and may be null.

Implements [edu.ou.asgbook.motion.MotionEstimator](#).

```

30
31     // do an edge filter on the pair of images
32     LatLonGrid edge0 = edgeFilter.edgeFilter(data0);
33     LatLonGrid edge1 = edgeFilter.edgeFilter(data1);
34
35     if (outdir != null){
36         try {
37             KmlWriter.write(edge0, outdir, "edge0", PngWriter.createCoolToWarmColormap());
38             KmlWriter.write(edge1, outdir, "edge1", PngWriter.createCoolToWarmColormap());
39         } catch (Exception e) {
40             e.printStackTrace();
41         }
42     }
43
44     return hornSchunk.compute(edge0, edge1, outdir);
45 }
```

**6.20.2.2 static void edu.ou.asgbook.motion.EdgeBased.main (String[] args)
throws Exception [static]**

```

47                                     {
48     // create output directory
49     File out = OutputDirectory.getDefault("edgemotion");
50
51     // read
52     File f = new File("data/seviri");
53     Pair<LatLonGrid,Date>[] grids = SeviriInfraredTemperature.readAll(f);
54
55     // do alg
56     MotionEstimator alg = new EdgeBased();
57     Pair<LatLonGrid,LatLonGrid> motion = alg.compute(grids[0].first, grids[1].first, out);
58
59     // write
60     SaturateFilter filter = new SaturateFilter(-150, 150);
```



```
61         LatLonGrid u = filter.filter(motion.first);
62         LatLonGrid v = filter.filter(motion.second);
63         KmlWriter.write(u, out, "opticflow_u", PngWriter.createCoolToWarmColormap());
64         KmlWriter.write(v, out, "opticflow_v", PngWriter.createCoolToWarmColormap());
65     }
```

6.21 edu.ou.asgbook.projections.Ellipsoid Class Reference

An ellipsoidal approximation to the earth.

Public Member Functions

- [Ellipsoid](#) (double [eqr](#), double [eccsq](#))

Static Public Member Functions

- static [Ellipsoid WGS84](#) ()
- static [Ellipsoid NAD27](#) ()

Public Attributes

- final double [eqr](#)
Equatorial radius (the semi-major axis) in meters.
- final double [eccsq](#)
Square of the eccentricity.

6.21.1 Detailed Description

An ellipsoidal approximation to the earth.

Author:

v.lakshmanan

6.21.2 Constructor & Destructor Documentation

6.21.2.1 edu.ou.asgbook.projections.Ellipsoid.Ellipsoid (double *eqr*, double *eccsq*)

```
18                                     {
19         super();
20         this.eqr = eqr;
21         this.eccsq = eccsq;
22     }
```

6.21.3 Member Function Documentation

6.21.3.1 static [Ellipsoid](#) edu.ou.asgbook.projections.Ellipsoid.NAD27 () [static]

```
28         {  
29     return new Ellipsoid(6378206, 0.006768658);  
30     }
```

6.21.3.2 static [Ellipsoid](#) edu.ou.asgbook.projections.Ellipsoid.WGS84 () [static]

```
24         {  
25     return new Ellipsoid(6378137, 0.00669438);  
26     }
```

6.21.4 Member Data Documentation

6.21.4.1 final double [edu.ou.asgbook.projections.Ellipsoid.eccsq](#)

Square of the eccentricity.

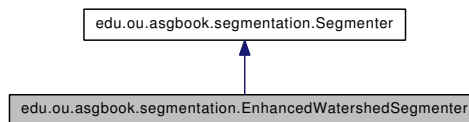
6.21.4.2 final double [edu.ou.asgbook.projections.Ellipsoid.eqr](#)

Equatorial radius (the semi-major axis) in meters.

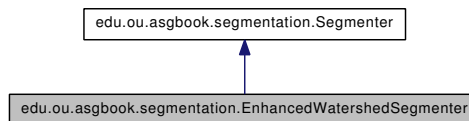
6.22 edu.ou.asgbook.segmentation.Enhanced-WatershedSegmenter Class Reference

Enhanced watershed segmentation following Lakshmanan, Hondl and Rabin.

Inheritance diagram for edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter:



Collaboration diagram for edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter:



Public Member Functions

- [EnhancedWatershedSegmenter](#) (int minThresh, int dataIncr, int maxThresh, int sizeThresholdInPixels, int deltaForCluster)
- [LabelResult label](#) (LatLonGrid dataval)

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

Package Functions

- [SuppressWarnings](#) ("serial") private static class Pixels extends ArrayList< [Pixel](#) >

Classes

- class **Glob**

6.22.1 Detailed Description

Enhanced watershed segmentation following Lakshmanan, Hondl and Rabin.

Author:

valliappa.lakshmanan

6.22.2 Constructor & Destructor Documentation

6.22.2.1 edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter.EnhancedWatershedSegmenter (int *minThresh*, int *dataIncr*, int *maxThresh*, int *sizeThresholdInPixels*, int *deltaForCluster*)

Parameters:

***minThresh*,:** minimum pixel value for a pixel to be part of a region

***dataIncr*,:** quantization interval. Use 1 if you don't want to quantize

***maxThresh*,:** values > maxThresh are treated as maxThresh

***sizeThresholdInPixels*,:** Blobs smaller than the specified size will get ignored.

***deltaForCluster*,:** Specify how many data-increments a cluster is allowed to range over. For example, if you specify 0, then a cluster will contain only values that fall in the same interval as the maximum. Larger values of D also yield clusters at larger scales.

```
43
44     this.myDelta = deltaForCluster;
45     this.myMinSize = sizeThresholdInPixels;
46     this.minThresh = minThresh;
47     this.maxThresh = maxThresh;
48     this.dataIncr = dataIncr;
49 }
```

6.22.3 Member Function Documentation

6.22.3.1 [LabelResult](#) edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter.label ([LatLonGrid](#) *dataval*) [virtual]

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

.. All pixels > thresh are part of an object.

Implements [edu.ou.asgbook.segmentation.Segmenter](#).

```

267                                     {
268         LatLonGrid marked = findLocalMaxima(dataaval);
269         LabelResult initial = new ThresholdSegmenter(0).label(marked);
270         LabelResult pruned = RegionProperty.pruneBySize(initial, dataaval, myMinSize);
271         return pruned;
272     }

```

6.22.3.2 static void edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter.main (String[] args) throws Exception [static]

```

274                                     {
275         File out = OutputDirectory.getDefault("ewshed");
276
277         // data
278         LatLonGrid grid = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulation());
279         KmlWriter.write(grid, out, "orig", PngWriter.createCoolToWarmColormap());
280
281         // int MIN = 200; int MAX = 500; int INCR = 10; // log scaling
282         int MIN = 1; int MAX = 100; int INCR = 1; // linear scaling
283
284         for (int sizethresh=5; sizethresh <= 20; sizethresh += 5){
285             EnhancedWatershedSegmenter seg = new EnhancedWatershedSegmenter(MIN, INCR, MAX);
286             LatLonGrid label = seg.label(grid).label;
287             KmlWriter.write(label, out, "ewsheds_"+sizethresh, PngWriter.createRandomColorMap());
288         }
289
290         grid = new ConvolutionFilter(ConvolutionFilter.gauss(9, 9)).smooth(grid);
291         for (int sizethresh=5; sizethresh <= 20; sizethresh += 5){
292             EnhancedWatershedSegmenter seg = new EnhancedWatershedSegmenter(MIN, INCR, MAX);
293             LatLonGrid label = seg.label(grid).label;
294             KmlWriter.write(label, out, "smewsheds_"+sizethresh, PngWriter.createRandomColorMap());
295         }
296     }

```

6.22.3.3 edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter.SuppressWarnings ("serial") [package]

```

52                                     {
53     }

```

6.23 edu.ou.asgbook.histogram.Entropy Class Reference

Compute entropy from a histogram.

Static Public Member Functions

- static double [computeEntropy](#) ([Histogram](#) hist)
- static void [main](#) (String[] args) throws Exception

6.23.1 Detailed Description

Compute entropy from a histogram.

Author:

v.lakshmanan

6.23.2 Member Function Documentation

6.23.2.1 static double edu.ou.asgbook.histogram.Entropy.computeEntropy ([Histogram](#) hist) [static]

```
17                                     {
18     float[] prob = hist.calcProb();
19     double entropy = 0;
20     for (int i=0; i < prob.length; ++i){
21         if ( prob[i] > 0 ){
22             double plogp = prob[i] * Math.log(prob[i]);
23             entropy -= plogp;
24         }
25     }
26     // to base 2
27     return (entropy / Math.log(2.0));
28 }
```

6.23.2.2 static void edu.ou.asgbook.histogram.Entropy.main (String[] args) throws Exception [static]

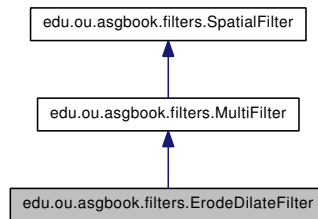
```
30                                     {
31     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
32     Histogram hist = new Histogram(0, 100, 100 );
33     hist.update(popdensity);
34     double e1 = Entropy.computeEntropy(hist);
35     System.out.println("Population density entropy = " + e1);
36 }
```

```
36
37     LatLonGrid albedo = SurfaceAlbedo.read(SurfaceAlbedo.CONUS, 100);
38     hist = new Histogram(0, 30, 30 );
39     hist.update(albedo);
40     double e2 = Entropy.computeEntropy(hist);
41     System.out.println("surface albedo entropy = " + e2);
42 }
```

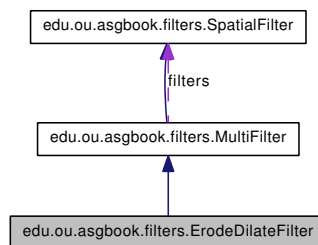

6.24 edu.ou.asgbook.filters.ErodeDilateFilter Class Reference

Carries out paired erosion followed by dilation for denoising.

Inheritance diagram for edu.ou.asgbook.filters.ErodeDilateFilter:



Collaboration diagram for edu.ou.asgbook.filters.ErodeDilateFilter:



Public Member Functions

- [ErodeDilateFilter](#) (int halfSize, int numTimes)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.24.1 Detailed Description

Carries out paired erosion followed by dilation for denoising.

Author:

Valliappa.Lakshmanan

6.24.2 Constructor & Destructor Documentation

6.24.2.1 `edu.ou.asgbook.filters.ErodeDilateFilter.ErodeDilateFilter (int halfSize, int numTimes)`

```
21                                     {
22     super(new SpatialFilter[]{
23         new ErosionFilter(halfSize), new DilationFilter(halfSize)
24     }, numTimes);
25 }
```

6.24.3 Member Function Documentation

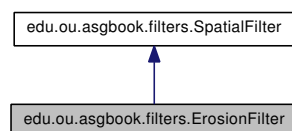
6.24.3.1 `static void edu.ou.asgbook.filters.ErodeDilateFilter.main (String[] args) throws Exception` [static]

```
27                                     {
28     // create output directory
29     File out = OutputDirectory.getDefault("dilateerode");
30
31     // read input
32     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new Gl
33     KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
34
35     // erode
36     LatLonGrid erode1 = new ErodeDilateFilter(1,1).filter(popdensity);
37     KmlWriter.write(erode1, out, "erodedilate_3_1", PngWriter.createCoolToWarmColormap());
38     LatLonGrid erode3 = new ErodeDilateFilter(1,3).filter(popdensity);
39     KmlWriter.write(erode3, out, "erodedilate_3_3", PngWriter.createCoolToWarmColormap());
40     LatLonGrid erode5 = new ErodeDilateFilter(2,3).filter(popdensity);
41     KmlWriter.write(erode5, out, "erodedilate_5_3", PngWriter.createCoolToWarmColormap());
42 }
```

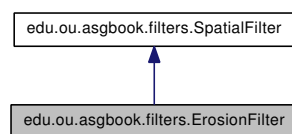
6.25 edu.ou.asgbook.filters.ErosionFilter Class Reference

Reduces the size of entities by taking a local minimum.

Inheritance diagram for edu.ou.asgbook.filters.ErosionFilter:



Collaboration diagram for edu.ou.asgbook.filters.ErosionFilter:



Public Member Functions

- [ErosionFilter](#) (int halfSize)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)
- [LatLonGrid erode](#) (final [LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.25.1 Detailed Description

Reduces the size of entities by taking a local minimum.

Author:

Valliappa.Lakshmanan

6.25.2 Constructor & Destructor Documentation

6.25.2.1 `edu.ou.asgbook.filters.ErosionFilter.ErosionFilter (int halfSize)`

```

23                                     {
24         this.halfSize = halfSize;
25     }
```

6.25.3 Member Function Documentation

6.25.3.1 [LatLonGrid](#) `edu.ou.asgbook.filters.ErosionFilter.erode (final LatLonGrid input)`

```

32                                     {
33         LatLonGrid output = LatLonGrid.copyOf(input);
34         output.fill(output.getMissing());
35         int[][] outData = output.getData();
36         int[][] inData = input.getData();
37         int hx = halfSize;
38         int hy = halfSize;
39         int nx = inData.length;
40         int ny = inData[0].length;
41         for (int i=hx; i < (nx-hx); ++i){
42             for (int j=hy; j < (ny-hy); ++j){
43                 int min = input.getMissing();
44                 boolean set = false;
45                 for (int m=-hx; m <= hx; ++m){
46                     for (int n=-hy; n <= hy; ++n){
47                         int inval = inData[i+m][j+n];
48                         if (inval != input.getMissing()){
49                             if ( !set || inval < min ){
50                                 min = inval;
51                                 set = true;
52                             }
53                         }
54                     }
55                 }
56                 if ( set ){
57                     outData[i][j] = min;
58                 }
59             }
60         }
61         return output;
62     }
```

6.25.3.2 Override [LatLonGrid](#) `edu.ou.asgbook.filters.ErosionFilter.filter (LatLonGrid input)`

```

28                                     {
29         return erode(input);
30     }
```

**6.25.3.3 static void edu.ou.asgbook.filters.ErosionFilter.main (String[] args)
throws Exception [static]**

```
64                                     {
65     // create output directory
66     File out = OutputDirectory.getDefault("erode");
67
68     // read input
69     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulati
70     KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
71
72     // erode
73     LatLonGrid erode1 = new ErosionFilter(1).erode(popdensity);
74     KmlWriter.write(erode1, out, "erode_3", PngWriter.createCoolToWarmColormap());
75     LatLonGrid erode3 = new ErosionFilter(3).erode(popdensity);
76     KmlWriter.write(erode3, out, "erode_7", PngWriter.createCoolToWarmColormap());
77     LatLonGrid erode5 = new ErosionFilter(5).erode(popdensity);
78     KmlWriter.write(erode5, out, "erode_11", PngWriter.createCoolToWarmColormap());
79 }
```

6.26 edu.ou.asgbook.io.EsriGrid Class Reference

Read an ESRI grid.

Static Public Member Functions

- static [LatLonGrid](#) [read](#) (File file, [DataTransform](#) t) throws IOException, FileNotFoundException
- static [LatLonGrid](#) [read](#) (Reader inputFile, [DataTransform](#) t)
- static void [write](#) ([LatLonGrid](#) data, File outdir, String fname) throws IOException
- static void [write](#) ([LatLonGrid](#) data, File out) throws IOException

6.26.1 Detailed Description

Read an ESRI grid.

Author:

valliappa.lakshmanan

6.26.2 Member Function Documentation

6.26.2.1 static [LatLonGrid](#) edu.ou.asgbook.io.EsriGrid.read (Reader *inputFile*, [DataTransform](#) t) [static]

```

41                                                                 {
42         BufferedReader reader = null;
43         try {
44             reader = new BufferedReader(inputFile);
45             // fields separated by spaces. This is the regular expression for spaces
46             final String sep = " +";
47             // read header
48             int ncols = Integer.parseInt( reader.readLine().split(sep)[1] );
49             int nrows = Integer.parseInt( reader.readLine().split(sep)[1] );
50             double cornerlon = Double.parseDouble( reader.readLine().split(sep)[1] );
51             double cornerlat = Double.parseDouble( reader.readLine().split(sep)[1] );
52             double latres = Double.parseDouble( reader.readLine().split(sep)[1] );
53             double lonres = latres;
54             String missingValue = reader.readLine().split(sep)[1];
55             int missing = Integer.parseInt(missingValue);
56
57             // read in data
58             int[][] data = new int[nrows][ncols];
59             int numvalid = 0;
60             int nummissing = 0;
61             int numzero = 0;

```

```

62         int minval = Integer.MAX_VALUE;
63         int maxval = 0;
64         int i = 0;
65         int j = 0;
66         String line = null;
67         while ( (line = reader.readLine()) != null ){
68             for (String field : line.split(sep)){
69                 if (field.equals(missingValue)){
70                     data[i][j] = missing;
71                     ++nummissing;
72                 } else {
73                     double value = Double.parseDouble(field);
74                     data[i][j] = t.transformAndRoundoff(value);
75                     if ( data[i][j] != 0 ){
76                         ++numvalid;
77                         minval = Math.min(minval, data[i][j]);
78                         maxval = Math.max(maxval, data[i][j]);
79                     } else {
80                         ++numzero;
81                     }
82                 }
83                 ++j; // next column
84                 if ( j == ncols ){
85                     j = 0; // next row
86                     ++i;
87                 }
88             }
89         }
90         System.out.println(numvalid + " valid pixels; " + numzero + " zero; " + nummissing + " " +
91         LatLon nwCorner = new LatLon(cornerlat + latres*nrows, cornerlon);
92         return new LatLonGrid(data, missing, nwCorner, latres, lonres);
93     } catch (Exception e){
94         System.err.println("Error reading file: " + e);
95         throw new IllegalArgumentException(e);
96     } finally {
97         if (reader != null) {
98             try{
99                 reader.close();
100             } catch (Exception e){
101                 // okay
102             }
103         }
104     }
105 }

```

6.26.2.2 static [LatLonGrid](#) edu.ou.asgbook.io.EsriGrid.read (File *file*, [DataTransform](#) *t*) throws IOException, FileNotFoundException

[static]

```

30
31     Reader f = null;
32     if (file.getAbsolutePath().endsWith(".gz")) {
33         f = new InputStreamReader(new GZIPInputStream(new FileInputStream(
34             file)));

```

```

35         } else {
36             f = new FileReader(file);
37         }
38         return read(f, t);
39     }

```

6.26.2.3 static void edu.ou.asgbook.io.EsriGrid.write ([LatLonGrid](#) data, File out) throws IOException [static]

```

112                                                     {
113         PrintWriter writer = null;
114         try {
115             writer = new PrintWriter(new GZIPOutputStream(new FileOutputStream(out)));
116             int nrows = data.getNumLat();
117             int ncols = data.getNumLon();
118             writer.println("ncols " + ncols);
119             writer.println("nrows " + nrows);
120             writer.println("xllcorner " + data.getNwCorner().getLon());
121             writer.println("yllcorner " + (data.getNwCorner().getLat() - data.getLatRes()) *
122             writer.println("cellsize " + data.getLatRes());
123             writer.println("NODATA_value " + data.getMissing());
124
125             final String sep = " ";
126             for (int i=0; i < nrows; ++i){
127                 for (int j=0; j < ncols; ++j){
128                     writer.print(data.getValue(i, j));
129                     if (j != (ncols-1)){
130                         writer.print(sep);
131                     }
132                 }
133                 writer.println();
134             }
135         } finally {
136             if (writer != null){
137                 System.out.println("Successfully wrote " + out);
138                 writer.close();
139             }
140         }
141     }

```

6.26.2.4 static void edu.ou.asgbook.io.EsriGrid.write ([LatLonGrid](#) data, File outdir, String fname) throws IOException [static]

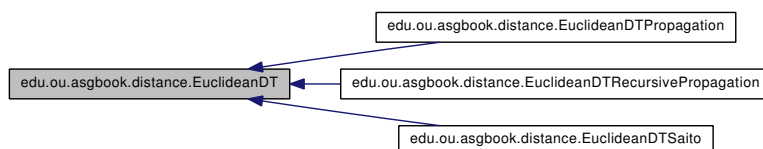
```

107
108         File f = new File(outdir.getAbsolutePath() + "/" + fname);
109         write(data, f);
110     }

```


6.27 edu.ou.asgbook.distance.EuclideanDT Interface Reference

Inheritance diagram for edu.ou.asgbook.distance.EuclideanDT:



Public Member Functions

- abstract [LatLonGrid](#) `getDistanceTransform` ([LatLonGrid](#) data, int thresh)
At every pixel, finds the square of the Euclidean distance to the nearest pixel > thresh.

6.27.1 Member Function Documentation

6.27.1.1 abstract [LatLonGrid](#) `edu.ou.asgbook.distance.EuclideanDT.getDistanceTransform` ([LatLonGrid](#) data, int thresh) [pure virtual]

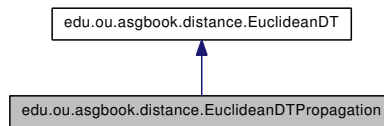
At every pixel, finds the square of the Euclidean distance to the nearest pixel > thresh.

Implemented in [edu.ou.asgbook.distance.EuclideanDTPropagation](#),
[edu.ou.asgbook.distance.EuclideanDTRecursivePropagation](#), and
[edu.ou.asgbook.distance.EuclideanDTSaito](#).

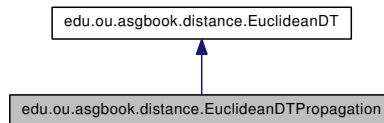
6.28 edu.ou.asgbook.distance.Euclidean-DTPropagation Class Reference

Implementation of Euclidean distance that updates the distance instead of computing it afresh each time.

Inheritance diagram for edu.ou.asgbook.distance.EuclideanDTPropagation:



Collaboration diagram for edu.ou.asgbook.distance.EuclideanDTPropagation:



Public Member Functions

- Override [LatLonGrid](#) `getDistanceTransform` ([LatLonGrid](#) data, int thresh)
At every pixel, finds the square of the Euclidean distance to the nearest pixel > thresh.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.28.1 Detailed Description

Implementation of Euclidean distance that updates the distance instead of computing it afresh each time.

Author:

v.lakshmanan

6.28.2 Member Function Documentation

6.28.2.1 Override [LatLonGrid](#) edu.ou.asgbook.distance.Euclidean-DTPropagation.getDistanceTransform ([LatLonGrid](#) data, int thresh) [virtual]

At every pixel, finds the square of the Euclidean distance to the nearest pixel > thresh.

Implements [edu.ou.asgbook.distance.EuclideanDT](#).

```

26                                     {
27         int nrows = data.getNumLat();
28         int ncols = data.getNumLon();
29         final int MAXDIST = nrows * nrows + ncols * ncols;
30         LatLonGrid dist = new LatLonGrid(nrows, ncols, MAXDIST,
31             data.getNwCorner(), data.getLatRes(), data.getLonRes());
32         dist.fill(dist.getMissing());
33         for (int i = 0; i < nrows; ++i)
34             for (int j = 0; j < ncols; ++j) {
35                 if (data.getValue(i, j) > thresh) {
36                     dist.setValue(i, j, 0);
37                     propagate(dist, i, j, 250*250);
38                 }
39             }
40         return dist;
41     }

```

6.28.2.2 static void edu.ou.asgbook.distance.EuclideanDTPropagation.main (String[] args) throws Exception [static]

```

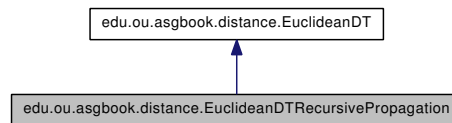
70                                     {
71         File out = OutputDirectory.getDefault("distance");
72         LatLonGrid popdensity = GlobalPopulation
73             .read(GlobalPopulation.NORTHAMERICA);
74
75         EuclideanDT transform = new EuclideanDTPropagation();
76         LatLonGrid edt = transform.getDistanceTransform(popdensity, 50);
77
78         // write it clamped out at a reasonable distance
79         final int maxdist = 250 * 250;
80         for (int i=0; i < edt.getNumLat(); ++i){
81             for (int j=0; j < edt.getNumLon(); ++j){
82                 if ( edt.getValue(i,j) > maxdist){
83                     edt.setValue(i,j, edt.getMissing() );
84                 }
85             }
86         }
87         KmlWriter.write(edt, out, "edtupdate", PngWriter.createCoolToWarmColormap());
88     }

```

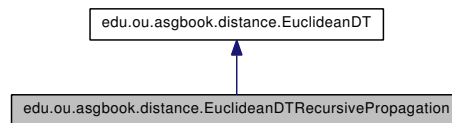
6.29 edu.ou.asgbook.distance.EuclideanDTRecursive-Propagation Class Reference

Note that this class is only for illustrative purposes.

Inheritance diagram for edu.ou.asgbook.distance.EuclideanDTRecursivePropagation:



Collaboration diagram for edu.ou.asgbook.distance.EuclideanDTRecursivePropagation:



Public Member Functions

- Override [LatLonGrid](#) `getDistanceTransform` ([LatLonGrid](#) data, int thresh)
At every pixel, finds the square of the Euclidean distance to the nearest pixel > thresh.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.29.1 Detailed Description

Note that this class is only for illustrative purposes.

It will not work because of stack overflow. Use the [EuclideanDTPropagation](#) implementation that replaces the recursion by a list.

Author:

v.lakshmanan

6.29.2 Member Function Documentation

6.29.2.1 Override [LatLonGrid](#) edu.ou.asgbook.distance.Euclidean-DTRecursivePropagation.getDistanceTransform ([LatLonGrid](#) *data*, int *thresh*) [virtual]

At every pixel, finds the square of the Euclidean distance to the nearest pixel > thresh.

Implements [edu.ou.asgbook.distance.EuclideanDT](#).

```
24                                     {
25         int nrows = data.getNumLat();
26         int ncols = data.getNumLon();
27         final int MAXDIST = nrows * nrows + ncols * ncols;
28         LatLonGrid dist = new LatLonGrid(nrows, ncols, MAXDIST,
29             data.getNwCorner(), data.getLatRes(), data.getLonRes());
30         dist.fill(dist.getMissing());
31         for (int i = 0; i < nrows; ++i)
32             for (int j = 0; j < ncols; ++j) {
33                 if (data.getValue(i, j) > thresh) {
34                     dist.setValue(i, j, 0);
35                     propagate(dist, i, j, i, j);
36                 }
37             }
38         return dist;
39     }
```

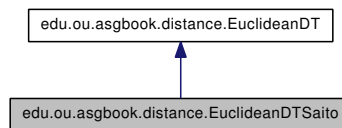
6.29.2.2 static void edu.ou.asgbook.distance.EuclideanDTRecursive-Propagation.main (String[] *args*) throws Exception [static]

```
55                                     {
56         File out = OutputDirectory.getDefault("distance");
57         LatLonGrid popdensity = GlobalPopulation
58             .read(GlobalPopulation.NORTHAMERICA);
59
60         EuclideanDT transform = new EuclideanDTRecursivePropagation();
61         LatLonGrid edt = transform.getDistanceTransform(popdensity, 50);
62
63         // write it clamped out at a reasonable distance
64         final int maxdist = 250 * 250;
65         for (int i=0; i < edt.getNumLat(); ++i){
66             for (int j=0; j < edt.getNumLon(); ++j){
67                 if ( edt.getValue(i,j) > maxdist){
68                     edt.setValue(i,j, edt.getMissing() );
69                 }
70             }
71         }
72         KmlWriter.write(edt, out, "edt", PngWriter.createCoolToWarmColormap());
73     }
```

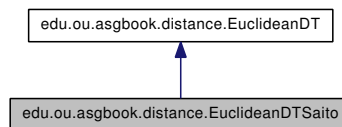
6.30 edu.ou.asgbook.distance.EuclideanDTSaito Class Reference

The Saito technique of computing the distance transform by calculating in the two directions separately.

Inheritance diagram for edu.ou.asgbook.distance.EuclideanDTSaito:



Collaboration diagram for edu.ou.asgbook.distance.EuclideanDTSaito:



Public Member Functions

- Override [LatLonGrid](#) `getDistanceTransform` ([LatLonGrid](#) data, int thresh)
At every pixel, finds the square of the Euclidean distance to the nearest pixel > thresh.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.30.1 Detailed Description

The Saito technique of computing the distance transform by calculating in the two directions separately.

Author:

v.lakshmanan

6.30.2 Member Function Documentation

6.30.2.1 Override [LatLonGrid](#) edu.ou.asgbook.distance.EuclideanDTSaito.getDistanceTransform ([LatLonGrid](#) *data*, int *thresh*) [virtual]

At every pixel, finds the square of the Euclidean distance to the nearest pixel > thresh.

Implements [edu.ou.asgbook.distance.EuclideanDT](#).

```
24                                     {
25         try {
26             return getDistanceTransform(data, thresh, null);
27         } catch (Exception e) {
28             throw new IllegalStateException();
29         }
30     }
```

6.30.2.2 static void edu.ou.asgbook.distance.EuclideanDTSaito.main (String[] *args*) throws Exception [static]

```
124                                     {
125         File out = OutputDirectory.getDefault("euclideanDt");
126         LatLonGrid popdensity = GlobalPopulation
127             .read(GlobalPopulation.NORTHAMERICA);
128
129         EuclideanDTSaito transform = new EuclideanDTSaito();
130         LatLonGrid edt = transform.getDistanceTransform(popdensity, 50, out);
131         writeClamped(edt, out, "edt");
132     }
```

6.31 edu.ou.asgbook.transforms.FFT Class Reference

[FFT](#) based on Sedgewick and Wayne.

Static Public Member Functions

- static `Complex[]` [fft](#) (`Complex[]` x)
Computes [FFT](#) of array whose length is a power of 2.
- static `Complex[]` [ifft](#) (`Complex[]` x)
compute inverse [FFT](#) of array whose length is a power of 2
- static void [main](#) (`String[]` args)

Classes

- class `Complex`

6.31.1 Detailed Description

[FFT](#) based on Sedgewick and Wayne.

Author:

v.lakshmanan

6.31.2 Member Function Documentation

6.31.2.1 static `Complex[]` edu.ou.asgbook.transforms.FFT.fft (`Complex[]` x) [static]

Computes [FFT](#) of array whose length is a power of 2.

```

54                                     {
55         int N = x.length;
56
57         if (N == 1){
58             return new Complex[] { x[0] };
59         } else if (N % 2 != 0) {
60             throw new IllegalArgumentException("N is not a power of 2");
61         }
62
63         // Break the array down into two parts and perform FFT of each piece
64         Complex[] part = new Complex[N / 2];

```



```

65         for (int k = 0; k < N / 2; k++) {
66             part[k] = x[2 * k]; // even terms
67         }
68         Complex[] evenfft = fft(part);
69         for (int k = 0; k < N / 2; k++) {
70             part[k] = x[2 * k + 1]; // odd terms
71         }
72         Complex[] oddfft = fft(part);
73
74         // combine
75         Complex[] y = new Complex[N];
76         for (int k = 0; k < N / 2; k++) {
77             double kth = -2 * k * Math.PI / N;
78             Complex wk = new Complex(Math.cos(kth), Math.sin(kth));
79             y[k] = evenfft[k].add(wk.multiply(oddfft[k]));
80             y[k + N / 2] = evenfft[k].subtract(wk.multiply(oddfft[k]));
81         }
82         return y;
83     }

```

6.31.2.2 static Complex [] edu.ou.asgbook.transforms.FFT.iff (Complex[] x) [static]

compute inverse [FFT](#) of array whose length is a power of 2

```

86                                     {
87         int N = x.length;
88
89         // Conjugate x
90         Complex[] y = new Complex[N];
91         for (int i = 0; i < N; i++) {
92             y[i] = x[i].conjugate();
93         }
94
95         // compute forward FFT
96         y = fft(y);
97
98         // Conjugate result and divide by N
99         for (int i = 0; i < N; i++) {
100             y[i] = y[i].conjugate().divide(N);
101         }
102
103         return y;
104
105     }

```

6.31.2.3 static void edu.ou.asgbook.transforms.FFT.main (String[] args) [static]

```

107                                     {
108         DecimalFormat df = new DecimalFormat("0.0");

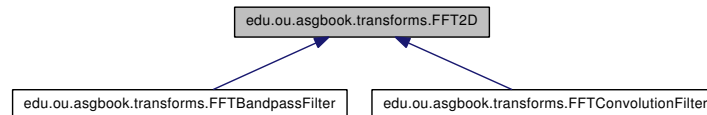
```

```
109         // FFT( rect ) should be a sinc function
110         FFT.Complex[] input = new FFT.Complex[32];
111         for (int i=0; i < input.length; ++i){
112             input[i] = new FFT.Complex(0,0);
113         }
114         for (int i=input.length/3; i < 2*input.length/3; ++i){
115             input[i] = new FFT.Complex(1, 0);
116         }
117         FFT.Complex[] output = fft(input);
118         for (int i=0; i < output.length; ++i){
119             System.out.print(df.format(output[i].norm()) + " ");
120         }
121         System.out.println();
122         FFT.Complex[] reverse = ifft(output);
123         for (int i=0; i < reverse.length; ++i){
124             System.out.print(df.format(reverse[i].norm()) + " ");
125         }
126         System.out.println();
127     }
```

6.32 edu.ou.asgbook.transforms.FFT2D Class Reference

Two-dimensional [FFT](#).

Inheritance diagram for edu.ou.asgbook.transforms.FFT2D:



Static Public Member Functions

- static `Complex[][]` [ifft](#) (`Complex[][]` input)
- static `Complex[][]` [zeropad](#) (`LatLonGrid` data)
- static `Complex[][]` [zeropad](#) (`double[][]` data)
- static `Complex[][]` [zeropad](#) (`double[][]` data, `int` nrows, `int` ncols)
- static `Complex[][]` [fft](#) (`Complex[][]` data)
- static void [main](#) (`String[]` args) throws Exception

6.32.1 Detailed Description

Two-dimensional [FFT](#).

Author:

valliappa.lakshmanan

6.32.2 Member Function Documentation

6.32.2.1 static `Complex[][]` edu.ou.asgbook.transforms.FFT2D.fft (`Complex data[][]`) `[static]`

```

74                                     {
75         int nrows = data.length;
76         int ncols = data[0].length;
77
78         // compute fft row-by-row
79         Complex[ ][ ] rowwise = new Complex[nrows][ ];
80         for (int i=0; i < nrows; ++i){
81             rowwise[i] = FFT.fft(data[i]);
82         }
83     }
  
```

```

84         // on the result, compute fft column by column
85         Complex[][] result = new Complex[nrows][ncols];
86         {
87             Complex[] tmp = new Complex[nrows];
88             for (int j=0; j < ncols; ++j){
89                 for (int i=0; i < nrows; ++i){
90                     tmp[i] = rowwise[i][j];
91                 }
92                 Complex[] tmp2 = FFT.fft(tmp);
93                 for (int i=0; i < nrows; ++i){
94                     result[i][j] = tmp2[i];
95                 }
96             }
97         }
98
99         return result;
100     }

```

6.32.2.2 static Complex [][] edu.ou.asgbook.transforms.FFT2D.ifft (Complex input[][]) [static]

```

16
17         // compute ifft row-wise, then column-wise
18         int nrows = input.length;
19         int ncols = input[0].length;
20         Complex[][] rowwise = new Complex[nrows][];
21         for (int i=0; i < nrows; ++i){
22             rowwise[i] = FFT.ifft(input[i]);
23         }
24         Complex[][] result = new Complex[nrows][ncols];
25         Complex[] tmp = new Complex[nrows];
26         for (int j=0; j < ncols; ++j){
27             for (int i=0; i < nrows; ++i){
28                 tmp[i] = rowwise[i][j];
29             }
30             Complex[] tmp2 = FFT.ifft(tmp);
31             for (int i=0; i < nrows; ++i){
32                 result[i][j] = tmp2[i];
33             }
34         }
35         return result;
36     }

```

6.32.2.3 static void edu.ou.asgbook.transforms.FFT2D.main (String[] args) throws Exception [static]

Reimplemented in [edu.ou.asgbook.transforms.FFTBandpassFilter](#), and [edu.ou.asgbook.transforms.FFTConvolutionFilter](#).

```

106
107         DecimalFormat df = new DecimalFormat("0.0");

```

```

108         // FFT( rect ) should be a sinc function
109         FFT.Complex[][] input = new FFT.Complex[8][8];
110         for (int i=0; i < input.length; ++i){
111             for (int j=0; j < input[i].length; ++j){
112                 input[i][j] = new FFT.Complex(0,0);
113             }
114         }
115         for (int i=input.length/3; i < 2*input.length/3; ++i){
116             for (int j=input[i].length/3; j < 2*input[i].length/3; ++j){
117                 input[i][j] = new FFT.Complex(1, 0);
118             }
119         }
120         FFT.Complex[][] output = fft(input);
121         for (int i=0; i < output.length; ++i){
122             for (int j=0; j < input[i].length; ++j){
123                 System.out.print(df.format(output[i][j].norm()) + " ");
124             }
125             System.out.println();
126         }
127         System.out.println();
128         FFT.Complex[][] reverse = ifft(output);
129         for (int i=0; i < reverse.length; ++i){
130             for (int j=0; j < reverse[i].length; ++j){
131                 System.out.print(df.format(reverse[i][j].norm()) + " ");
132             }
133             System.out.println();
134         }
135         System.out.println();
136     }

```

6.32.2.4 static Complex [][] edu.ou.asgbook.transforms.FFT2D.zeropad (double data[][], int nrows, int ncols) [static]

```

60                                                                 {
61         Complex[][] result = new Complex[nrows][ncols];
62         Complex ZERO = new Complex(0,0);
63         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
64             result[i][j] = ZERO;
65         }
66         for (int i=0; i < data.length; ++i){
67             for (int j=0; j < data[0].length; ++j){
68                 result[i][j] = new Complex(data[i][j], 0);
69             }
70         }
71         return result;
72     }

```

6.32.2.5 static Complex [][] edu.ou.asgbook.transforms.FFT2D.zeropad (double data[][]) [static]

```

54                                                                 {
55         int nrows = getNextPowerOf2(data.length);

```

```
56         int ncols = getNextPowerOf2(data[0].length);
57         return zeropad(data, nrows, ncols);
58     }
```

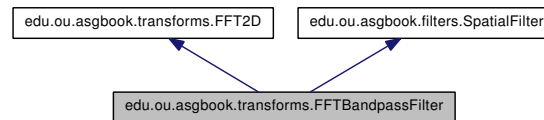
6.32.2.6 static Complex [][] edu.ou.asgbook.transforms.FFT2D.zeropad ([LatLonGrid](#) data) [static]

```
38         {
39         int nrows = getNextPowerOf2(data.getNumLat());
40         int ncols = getNextPowerOf2(data.getNumLon());
41         Complex[][] result = new Complex[nrows][ncols];
42         Complex ZERO = new Complex(0,0);
43         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
44             result[i][j] = ZERO;
45         }
46         for (int i=0; i < data.getNumLat(); ++i){
47             for (int j=0; j < data.getNumLon(); ++j){
48                 result[i][j] = new Complex(data.getValue(i,j), 0);
49             }
50         }
51         return result;
52     }
```

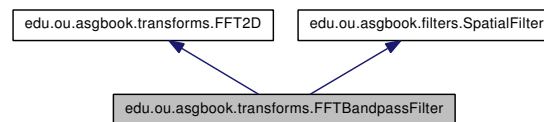
6.33 edu.ou.asgbook.transforms.FFTBandpassFilter Class Reference

Removes noise (high frequencies) and the gross signal (low frequencies).

Inheritance diagram for edu.ou.asgbook.transforms.FFTBandpassFilter:



Collaboration diagram for edu.ou.asgbook.transforms.FFTBandpassFilter:



Public Member Functions

- [FFTBandpassFilter](#) (double minr, double maxr)
Supply numbers in the range (0,1) where 1 is the full dynamic range.
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)
- [LatLonGrid convolve](#) ([LatLonGrid](#) data)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.33.1 Detailed Description

Removes noise (high frequencies) and the gross signal (low frequencies).

Author:

valliappa.lakshmanan

6.33.2 Constructor & Destructor Documentation

6.33.2.1 `edu.ou.asgbook.transforms.FFTBandpassFilter.FFTBandpassFilter` (double *minr*, double *maxr*)

Supply numbers in the range (0,1) where 1 is the full dynamic range.

```

25                                     {
26         this.minr = minr;
27         this.maxr = maxr;
28     }
```

6.33.3 Member Function Documentation

6.33.3.1 [LatLonGrid](#) `edu.ou.asgbook.transforms.FFTBandpassFilter.convolve` ([LatLonGrid](#) *data*)

```

35                                     {
36         Complex[][] in1 = zeropad(data);
37         final int nrows = in1.length;
38         final int ncols = in1[0].length;
39
40         // compute the fft
41         in1 = fft(in1);
42
43         // the fft is arranged in quadrants, so we need to be careful
44         // to remove the corresponding data in all the quadrants
45         Complex zero = new Complex(0,0);
46         double diag = Math.sqrt(nrows*nrows+ncols*ncols)/4;
47         for (int i=0; i < nrows/2; ++i){
48             for (int j=0; j < ncols/2; ++j){
49                 double r = Math.sqrt(i*i + j*j)/diag;
50                 if (r < minr || r > maxr){
51                     in1[i][j] = zero; // 1st quadrant
52                     in1[nrows-i-1][j] = zero; // 3rd quadrant
53                     in1[i][ncols-j-1] = zero; // 2nd quadrant
54                     in1[nrows-i-1][ncols-i-1] = zero; // 4th quadrant
55                 }
56             }
57         }
58
59         // take ifft
60         Complex[][] result = ifft(in1);
61
62         // return real part, rounded off
63         LatLonGrid out = LatLonGrid.copyOf(data);
64         for (int i=0; i < out.getNumLat(); ++i) for (int j=0; j < out.getNumLon(); ++j){
65             out.setValue(i, j, (int)Math.round(result[i][j].real));
66         }
67         return out;
68     }
```


6.33.3.2 Override [LatLonGrid](#) edu.ou.asgbook.transforms.FFTBandpassFilter.filter ([LatLonGrid](#) *input*)

```
31                                     {
32     return convolve(input);
33 }
```

6.33.3.3 static void edu.ou.asgbook.transforms.FFTBandpassFilter.main (String[] *args*) throws Exception [static]

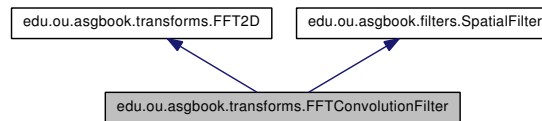
Reimplemented from [edu.ou.asgbook.transforms.FFT2D](#).

```
70                                     {
71     // create output directory
72     File out = OutputDirectory.getDefault("fftbandpass");
73
74     // read input
75     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulati
76     KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
77     LatLonGrid sm = new FFTBandpassFilter(0,0.1).convolve(popdensity);
78     KmlWriter.write(sm, out, "bp0_10", PngWriter.createCoolToWarmColormap());
79     sm = new FFTBandpassFilter(0,0.2).convolve(popdensity);
80     KmlWriter.write(sm, out, "bp0_20", PngWriter.createCoolToWarmColormap());
81     sm = new FFTBandpassFilter(0.2,0.8).convolve(popdensity);
82     KmlWriter.write(sm, out, "bp20_80", PngWriter.createCoolToWarmColormap());
83     sm = new FFTBandpassFilter(0.8,1.0).convolve(popdensity);
84     KmlWriter.write(sm, out, "bp80_100", PngWriter.createCoolToWarmColormap());
85 }
```

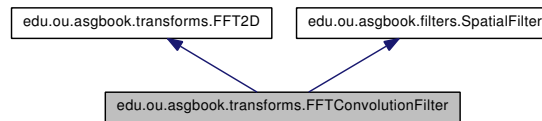
6.34 edu.ou.asgbook.transforms.FFTConvolutionFilter Class Reference

An optimization for convolution using FFTs.

Inheritance diagram for edu.ou.asgbook.transforms.FFTConvolutionFilter:



Collaboration diagram for edu.ou.asgbook.transforms.FFTConvolutionFilter:



Public Member Functions

- [FFTConvolutionFilter](#) (double[][] coeffs)
- [LatLonGrid convolve](#) ([LatLonGrid](#) data)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.34.1 Detailed Description

An optimization for convolution using FFTs.

Author:

valliappa.lakshmanan

6.34.2 Constructor & Destructor Documentation

6.34.2.1 edu.ou.asgbook.transforms.FFTConvolutionFilter.FFTConvolutionFilter (double *coeffs*[][])

```

23                                     {
24         this.coeffs = coeffs;
25     }
```

6.34.3 Member Function Documentation

6.34.3.1 [LatLonGrid](#) edu.ou.asgbook.transforms.FFTConvolutionFilter.convolve ([LatLonGrid](#) *data*)

```

27                                     {
28         Complex[][] in1 = zeropad(data);
29         int nrows = in1.length;
30         int ncols = in1[0].length;
31         Complex[][] in2 = zeropad(coeffs, nrows, ncols );
32
33         // compute their ffts
34         in1 = fft(in1);
35         in2 = fft(in2);
36
37         // multiply point by point (this by the conjugate of other)
38         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
39             in1[i][j] = in1[i][j].multiply(in2[i][j].conjugate());
40         }
41         // take ifft
42         Complex[][] result = ifft(in1);
43
44         // return real part, rounded off
45         LatLonGrid out = LatLonGrid.copyOf(data);
46         for (int i=0; i < out.getNumLat(); ++i) for (int j=0; j < out.getNumLon(); ++j){
47             out.setValue(i, j, (int)Math.round(result[i][j].real));
48         }
49         return out;
50     }
```

6.34.3.2 Override [LatLonGrid](#) edu.ou.asgbook.transforms.FFTConvolutionFilter.filter ([LatLonGrid](#) *input*)

```

53                                     {
54         return convolve(input);
55     }
```

6.34.3.3 static void edu.ou.asgbook.transforms.FFTConvolutionFilter.main (String[] args) throws Exception [static]

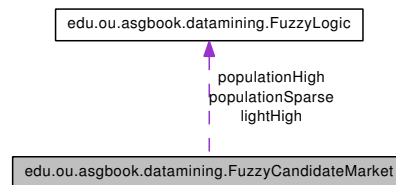
Reimplemented from [edu.ou.asgbook.transforms.FFT2D](#).

```
57                                     {
58         // create output directory
59         File out = OutputDirectory.getDefault("fftconv");
60
61         // read input
62         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GL
63         System.out.println("Grid size=" + popdensity.getNumLat() + "x" + popdensity.getNumL
64         KmlWriter.write(popdensity, out, "fullgrid", PngWriter.createCoolToWarmColormap());
65         double[][] coeffs = ConvolutionFilter.gauss(301, 301);
66         long timer = System.currentTimeMillis();
67         LatLonGrid sm = new FFTConvolutionFilter(coeffs).convolve(popdensity);
68         long ffttime = System.currentTimeMillis() - timer;
69         KmlWriter.write(sm, out, "fftgauss", PngWriter.createCoolToWarmColormap());
70
71         // do it in spatial domain
72         timer = System.currentTimeMillis();
73         LatLonGrid sm2 = new ConvolutionFilter(coeffs).convolve(popdensity);
74         long spatialtime = System.currentTimeMillis() - timer;
75         KmlWriter.write(sm2, out, "spgauss", PngWriter.createCoolToWarmColormap());
76
77         double improvement = 100*((double)(spatialtime - ffttime))/spatialtime;
78         System.out.println("The FFT technique took " + ffttime + "ms whereas the spatial te
79     }
```

6.35 edu.ou.asgbook.datamining.FuzzyCandidateMarket Class Reference

Uses heuristic rules to choose the next market to enter.

Collaboration diagram for edu.ou.asgbook.datamining.FuzzyCandidateMarket:



Public Member Functions

- [FuzzyCandidateMarket](#) ()
- [int isGoodCandidate](#) (double population, double lightIntensity)

Static Public Member Functions

- [static void main](#) (String[] args) throws Exception

6.35.1 Detailed Description

Uses heuristic rules to choose the next market to enter.

Author:

valliappa.lakshmanan

6.35.2 Constructor & Destructor Documentation

6.35.2.1 edu.ou.asgbook.datamining.FuzzyCandidateMarket.FuzzyCandidateMarket ()

```

34         {
35         lightHigh = new FuzzyLogic.IsHigh(30, 70);
36         populationSparse = new FuzzyLogic.IsLow(5,10);
37         populationHigh = new FuzzyLogic.IsHigh(30,80);
38     }
  
```

6.35.3 Member Function Documentation

6.35.3.1 `int edu.ou.asgbook.datamining.FuzzyCandidateMarket.isGoodCandidate (double population, double lightIntensity)`

```

41                                     {
42         // apply the basic rules
43         FuzzyLogic.Fuzzy highlight = lightHigh.apply(lightIntensity);
44         FuzzyLogic.Fuzzy popSparse = populationSparse.apply(population);
45         FuzzyLogic.Fuzzy popHigh = populationHigh.apply(population);
46
47         // if high light and moderate population density ...
48         FuzzyLogic.Fuzzy popModerate = popSparse.not().and( popHigh.not() );
49         FuzzyLogic.Fuzzy result = popModerate.and(highlight);
50
51         return (int) Math.round(result.getValue()*10);
52     }

```

6.35.3.2 `static void edu.ou.asgbook.datamining.FuzzyCandidateMarket.main (String[] args) throws Exception` [static]

```

54                                     {
55         // create output directory
56         File out = OutputDirectory.getDefault("fuzzy");
57
58         // read input (crop to cover Spain)
59         LatLonGrid lights = NighttimeLights.read(NighttimeLights.WORLD).crop(980, 4080, 220, 130);
60         LatLonGrid pop     = GlobalPopulation.read(GlobalPopulation.WORLD).crop(980, 4080, 220, 130);
61
62         // sanity check: are both grids correctly geolocated?
63         System.out.println("Lights nwcorner: " + lights.getNwCorner());
64         System.out.println("Population nwcorner: " + pop.getNwCorner());
65
66         // apply fuzzy logic
67         FuzzyCandidateMarket rules = new FuzzyCandidateMarket();
68         LatLonGrid result = LatLonGrid.copyOf(lights);
69         result.fill(0);
70         result.setMissing(0);
71         for (int i=0; i < result.getNumLat(); ++i){
72             for (int j=0; j < result.getNumLon(); ++j){
73                 result.setValue(i, j, rules.isGoodCandidate(pop.getValue(i, j), lights.getValue(i, j)));
74             }
75         }
76
77         // write out as image, for viewing
78         KmlWriter.write(lights, out, "fzlights", PngWriter.createCoolToWarmColormap());
79         KmlWriter.write(pop, out, "fzpop", PngWriter.createCoolToWarmColormap());
80         KmlWriter.write(result, out, "candidatepixels", PngWriter.createCoolToWarmColormap());
81
82         // find cities from population data using watershed
83         EnhancedWatershedSegmenter seg = new EnhancedWatershedSegmenter(10, 1, 130, 10, 5);
84         LabelResult label = seg.label(pop);

```

```

85     RegionProperty[] popProps = RegionProperty.compute(label, pop);
86     RegionProperty[] lightProps = RegionProperty.compute(label, lights);
87
88     List<LatLon> points = new ArrayList<LatLon>();
89     List<String> names = new ArrayList<String>();
90     int[] howgood = new int[popProps.length];
91     for (int i=1; i < howgood.length; ++i){
92         howgood[i] = rules.isGoodCandidate(popProps[i].getCval(), lightProps[i].getCval());
93         if (howgood[i] > 5){
94             points.add( result.getLocation(popProps[i].getCx(), popProps[i].getCy()) );
95             names.add( " " + howgood[i]);
96             System.out.println( points.get(points.size()-1) + " " + howgood[i]);
97         }
98     }
99     KmlWriter.write(points, names, out, "candidates");
100
101     LatLonGrid candidateCities = LatLonGrid.copyOf(result);
102     for (int i=0; i < candidateCities.getNumLat(); ++i){
103         for (int j=0; j < candidateCities.getNumLon(); ++j){
104             int regno = label.label.getValue(i, j);
105             if (regno > 0){
106                 candidateCities.setValue(i, j, howgood[regno]);
107             } else {
108                 candidateCities.setValue(i, j, 0);
109             }
110         }
111     }
112     KmlWriter.write(candidateCities, out, "candidateCities", PngWriter.createCoolToWarmColormap());
113 }

```

6.36 edu.ou.asgbook.datamining.FuzzyLogic Class Reference

A simple fuzzy logic engine.

Classes

- class **Aggregate**
Simply applies an equal weighting to all of the values.
- class **Fuzzy**
- class **IsAbout**
- class **IsHigh**
- class **IsLow**
- interface [Rule](#)

6.36.1 Detailed Description

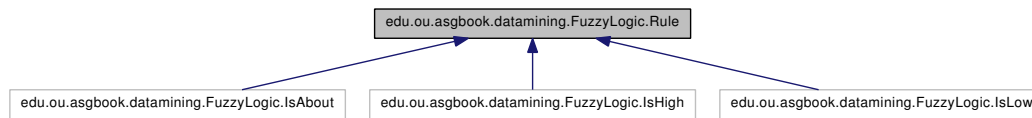
A simple fuzzy logic engine.

Author:

valliappa.lakshmanan

6.37 edu.ou.asgbook.datamining.FuzzyLogic.Rule Interface Reference

Inheritance diagram for edu.ou.asgbook.datamining.FuzzyLogic.Rule:



Public Member Functions

- Fuzzy [apply](#) (double value)

6.37.1 Member Function Documentation

6.37.1.1 Fuzzy edu.ou.asgbook.datamining.FuzzyLogic.Rule.apply (double value)

6.38 edu.ou.asgbook.gmm.GaussianComponent Class Reference

Component of a Gaussian Mixture Model.

Public Member Functions

- [GaussianComponent](#) (double cx, double cy, double varx, double vary, double sigmaxy, double inwt)

Initialize with known values.

- [GaussianComponent](#) (double cx, double cy, double varx, double vary)
- double [getWeight](#) ()
- double [getCx](#) ()
- double [getCy](#) ()
- double [getSigmax](#) ()
- double [getSigmay](#) ()
- double [getSigmaxy](#) ()
- [GaussianComponent](#) ([Pixel](#)[] pixels, double[] wts)

Finds best fit (the M-step in E-M).

- double [computeProbabilityDensityAt](#) ([Pixel](#) p)
- double [computeProbabilityDensityAt](#) (double x, double y)

Value of Normal function at x,y given these parameters.

- boolean [isValid](#) ()
- Override String [toString](#) ()

Classes

- class [Expectation](#)

6.38.1 Detailed Description

Component of a Gaussian Mixture Model.

Author:

valliappa.lakshmanan

6.38.2 Constructor & Destructor Documentation

6.38.2.1 edu.ou.asgbook.gmm.GaussianComponent.GaussianComponent (double *cx*, double *cy*, double *varx*, double *vary*, double *sigmaxy*, double *inwt*)

Initialize with known values.

```

21                                     {
22         mux = cx;
23         muy = cy;
24         sxx = varx;
25         syy = vary;
26         sxy = sigmaxy;
27         wt = inwt;
28         det = Math.abs(sxx * syy - sxy * sxy);
29         denom = 2 * Math.PI * Math.sqrt(det);
30     }
```

6.38.2.2 edu.ou.asgbook.gmm.GaussianComponent.GaussianComponent (double *cx*, double *cy*, double *varx*, double *vary*)

```

32                                     {
33         this(cx, cy, varx, vary, 0, 1);
34     }
```

6.38.2.3 edu.ou.asgbook.gmm.GaussianComponent.GaussianComponent (Pixel[] *pixels*, double[] *wt*)

Finds best fit (the M-step in E-M).

```

78                                     {
79         int n_pts = pixels.length;
80         if (wt.length != pixels.length){
81             throw new IllegalArgumentException("Array lengths have to match");
82         }
83         // compute pi_k (wt)
84         wt = 0;
85         for (int i = 0; i < n_pts; ++i) {
86             wt += wt[i];
87         }
88         wt /= n_pts;
89
90         // mean
91         Expectation wm_x = new Expectation(), wm_y = new Expectation();
92         for (int i = 0; i < n_pts; ++i) {
93             wm_x.update(pixels[i].getX(), wt[i]);
94             wm_y.update(pixels[i].getY(), wt[i]);
95         }
```

```

96         mux = wm_x.result();
97         muy = wm_y.result();
98
99         // covariance matrix
100         Expectation wv_x = new Expectation();
101         Expectation wv_y = new Expectation();
102         Expectation cv_xy = new Expectation();
103         for (int i = 0; i < n_pts; ++i) {
104             double dx = pixels[i].getX() - mux;
105             double dy = pixels[i].getY() - muy;
106             wv_x.update(dx * dx, wts[i]);
107             wv_y.update(dy * dy, wts[i]);
108             cv_xy.update(dx * dy, wts[i]);
109         }
110         sxx = wv_x.result();
111         syy = wv_y.result();
112         sxy = cv_xy.result();
113
114         final double EPSILON = 0.01; // at-least 1/10 pixel of variance ...
115         if (sxx < EPSILON || syy < EPSILON) {
116             det = denom = 0;
117             return;
118         }
119
120         // normalizing constant
121         det = (sxx * syy - sxy * sxy);
122         denom = 2 * Math.PI * Math.sqrt(Math.abs(det)); // always positive
123
124     }

```

6.38.3 Member Function Documentation

6.38.3.1 double edu.ou.asgbook.gmm.GaussianComponent.computeProbabilityDensityAt (double x, double y)

Value of Normal function at x,y given these parameters.

You typically want to weight this contribution by [getWeight\(\)](#) This goes into the E-step in E-M.

```

135                                     {
136         if (denom < 0.00001) {
137             return 0;
138         } // singular
139         double dx = x - mux;
140         double dy = y - muy;
141         double term = -(syy * dx * dx - 2 * sxy * dx * dy + sxx * dy * dy);
142         double num = Math.exp(term / (2 * det));
143         double result = num / denom;
144         if (result > 1) {
145             // usually because of numerical instability
146             return 0;
147         }
148         return result;

```

```
149
150 }
```

6.38.3.2 double edu.ou.asgbook.gmm.GaussianComponent.computeProbabilityDensityAt (Pixel *p*)

```
126
127     return computeProbabilityDensityAt(p.getX(), p.getY());
128 }
```

6.38.3.3 double edu.ou.asgbook.gmm.GaussianComponent.getCx ()

```
40
41     return mux;
42 }
```

6.38.3.4 double edu.ou.asgbook.gmm.GaussianComponent.getCy ()

```
44
45     return muy;
46 }
```

6.38.3.5 double edu.ou.asgbook.gmm.GaussianComponent.getSigmax ()

```
48
49     return Math.sqrt(sxx);
50 }
```

6.38.3.6 double edu.ou.asgbook.gmm.GaussianComponent.getSigmaxy ()

```
56
57     return sxy;
58 }
```

6.38.3.7 double edu.ou.asgbook.gmm.GaussianComponent.getSigmay ()

```
52
53     return Math.sqrt(syy);
54 }
```

6.38.3.8 double edu.ou.asgbook.gmm.GaussianComponent.getWeight ()

```
36         {
37         return wt;
38     }
```

6.38.3.9 boolean edu.ou.asgbook.gmm.GaussianComponent.isValid ()

```
152         {
153         return denom > 0;
154     }
```

6.38.3.10 Override String edu.ou.asgbook.gmm.GaussianComponent.toString ()

```
157         {
158         return "center=(" + mux + "," + muy + ") covar=(" + sxx + "," + sxy
159             + "," + syx + ") wt=" + wt;
160     }
```

6.39 edu.ou.asgbook.gmm.GaussianMixtureModel Class Reference

A parametric approximation of a spatial grid as a sum of Gaussians.

Public Member Functions

- [GaussianMixtureModel](#) ([LatLonGrid](#) input, int numModels)
- [GaussianComponent\[\]](#) [getMixture](#) ()
- void [setValueInGrid](#) ([LatLonGrid](#) data, double scale)

Estimates the value at each pixel in the grid based on GMM The values will be 0-1, so provide a scale in order to make them integers.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.39.1 Detailed Description

A parametric approximation of a spatial grid as a sum of Gaussians.

Author:

valliappa.lakshmanan

6.39.2 Constructor & Destructor Documentation

6.39.2.1 edu.ou.asgbook.gmm.GaussianMixtureModel.GaussianMixtureModel ([LatLonGrid](#) input, int numModels)

```
29                                     {
30         final int MIN_DISTSQ = 100; // distance between initial centers in px
31         initGMM(LevelSet.newInstance(input), numModels, MIN_DISTSQ);
32         final int MAX_ITER = 10;
33         final double MIN_IMPROVEMENT = 0.01; // 1 percent
34         tuneGMM(input.asPixels(), MAX_ITER, MIN_IMPROVEMENT);
35     }
```

6.39.3 Member Function Documentation

6.39.3.1 [GaussianComponent](#) [] edu.ou.asgbook.gmm.GaussianMixtureModel.getMixture ()

```

37         {
38         return mixture.toArray(new GaussianComponent[0]);
39     }

```

6.39.3.2 static void edu.ou.asgbook.gmm.GaussianMixtureModel.main (String[] args) throws Exception [static]

```

155         {
156         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new G
157
158         File out = OutputDirectory.getDefault("gmmpopdensity");
159         KmlWriter.write(popdensity, out, "original", PngWriter.createCoolToWarmColormap());
160
161         GaussianMixtureModel gmm = new GaussianMixtureModel(popdensity, 10);
162         GaussianComponent[] fit = gmm.getMixture();
163         List<LatLon> locs = new ArrayList<LatLon>();
164         List<String> names = new ArrayList<String>();
165         for (int i=0; i < fit.length; ++i){
166             LatLon loc = popdensity.getLocation( fit[i].getCx(), fit[i].getCy() );
167             String name = ("GMM#" + i + " ampl=" + fit[i].getWeight() + " sigmax=" + fit[i]
168             System.out.println(" loc: " + loc + name);
169             locs.add(loc);
170             names.add(name);
171         }
172         KmlWriter.write(locs, names, out, "gmmcities");
173
174         // write out the approximation
175         gmm.setValueInGrid(popdensity, 500);
176         KmlWriter.write(popdensity, out, "gmmapprox", PngWriter.createCoolToWarmColormap());
177     }

```

6.39.3.3 void edu.ou.asgbook.gmm.GaussianMixtureModel.setValueInGrid ([LatLonGrid](#) data, double scale)

Estimates the value at each pixel in the grid based on GMM The values will be 0-1, so provide a scale in order to make them integers.

Parameters:

data

```

46         {
47         data.setMissing(0);
48         double peakval = 0;

```

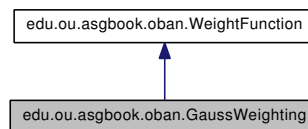


```
49         for (int m=0; m < mixture.size(); ++m){
50             peakval = Math.max( peakval, mixture.get(m).getWeight() );
51         }
52         for (int i=0; i < data.getNumLat(); ++i){
53             for (int j=0; j < data.getNumLon(); ++j){
54                 double raw = 0;
55                 for (int m=0; m < mixture.size(); ++m){
56                     raw += mixture.get(m).computeProbabilityDensityAt(i,j);
57                 }
58                 data.setValue(i,j, (int) Math.round(raw*scale/peakval));
59             }
60         }
61     }
```

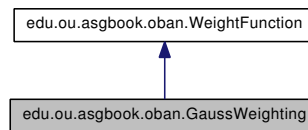
6.40 edu.ou.asgbook.oban.GaussWeighting Class Reference

An interpolation method that uses $\exp(-1/r^2)$.

Inheritance diagram for edu.ou.asgbook.oban.GaussWeighting:



Collaboration diagram for edu.ou.asgbook.oban.GaussWeighting:



Public Member Functions

- [GaussWeighting](#) (double sigma)
- Override double [computeWt](#) (double latdist, double londist)

Subclasses implement a weighting function.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.40.1 Detailed Description

An interpolation method that uses $\exp(-1/r^2)$.

Author:

Valliappa.Lakshmanan

6.40.2 Constructor & Destructor Documentation

6.40.2.1 edu.ou.asgbook.oban.GaussWeighting.GaussWeighting (double *sigma*)

Parameters:

sigma Set extent of gaussian in degrees

```

28         {
29             this.sigmasq = sigma * sigma;
30             this.epsilonDistSq = (3*3)*(2*sigmasq); // at distance of 3*sigma in both directions
31         }

```

6.40.3 Member Function Documentation

6.40.3.1 Override double edu.ou.asgbook.oban.GaussWeighting.computeWt (double *latdist*, double *londist*) [virtual]

Subclasses implement a weighting function.

If -ve value is returned, then the point will be considered too far away and not used in weighting.

Implements [edu.ou.asgbook.oban.WeightFunction](#).

```

34         {
35             double r2 = latdist * latdist + londist * londist;
36             if ( r2 < epsilonDistSq ){
37                 return Math.exp(-r2 / (2*sigmasq));
38             } else {
39                 return INVALID_WEIGHT;
40             }
41         }

```

6.40.3.2 static void edu.ou.asgbook.oban.GaussWeighting.main (String[] *args*) throws Exception [static]

```

43         {
44             PointObservations data = DailyRainfall.read(DailyRainfall.TN_Oct2010);
45
46             double sigma = ObjectiveAnalysisUtils.computeMeanDistance(data);
47             System.out.println("Objectively analyzing " + data.getPoints().length + " pts with a mean separa
48             WeightFunction wtFunc = new GaussWeighting(sigma);
49             WeightedAverage analyzer = new WeightedAverage(wtFunc, 0.01, 0.01, 1);
50             LatLonGrid grid = analyzer.analyze(data);
51
52             // write output
53             File out = OutputDirectory.getDefault("gaussoban");
54             KmlWriter.write(grid, out, "Precip24H", PngWriter.createCoolToWarmColormap());
55         }

```

6.41 edu.ou.asgbook.datamining.GdiPattern Class Reference

The training pattern for each city.

Public Member Functions

- Override String [toString](#) ()
- String [toString](#) (String colsep, String linesep)

Static Public Member Functions

- static [GdiPattern](#)[] [findTrainingPattern](#) ([LabelResult](#) cities, [LatLonGrid](#) population, [LatLonGrid](#) nightTimeLights, [LatLonGrid](#) gdiGrid)
- static void [write](#) ([GdiPattern](#)[] patterns, File outdir) throws IOException
- static void [main](#) (String[] args) throws Exception

6.41.1 Detailed Description

The training pattern for each city.

Author:

valliappa.lakshmanan

6.41.2 Member Function Documentation

6.41.2.1 static [GdiPattern](#) [] edu.ou.asgbook.datamining.GdiPattern.findTrainingPattern ([LabelResult](#) cities, [LatLonGrid](#) population, [LatLonGrid](#) nightTimeLights, [LatLonGrid](#) gdiGrid) [static]

```

52
53         // for each city, compute the other properties
54         RegionProperty[] pop = RegionProperty.compute(cities, population);
55         RegionProperty[] lights = RegionProperty.compute(cities, nightTimeLights);
56         RegionProperty[] gdi = RegionProperty.compute(cities, gdiGrid);
57         GdiPattern[] patterns = new GdiPattern[pop.length];
58         for (int i=1; i < patterns.length; ++i){
59             patterns[i] = new GdiPattern();
60             patterns[i].data[0] = pop[i].getCval();
61             patterns[i].data[1] = lights[i].getCval();
62             patterns[i].data[2] = gdi[i].getCval();
63         }
64         return patterns;
65     }
```

6.41.2.2 static void edu.ou.asgbook.datamining.GdiPattern.main (String[] args) throws Exception [static]

```

82                                     {
83     // create output directory
84     File out = OutputDirectory.getDefault("gdipattern");
85     final boolean SMALL = true;
86
87     // read input (crop to cover Spain)
88     LatLonGrid pop = GlobalPopulation.read(GlobalPopulation.WORLD);
89     if (SMALL){
90         pop = pop.crop(980, 4080, 220, 350);
91     }
92     KmlWriter.write(pop, out, "pop", PngWriter.createRandomColormap());
93
94     LatLonGrid nightTimeLights = NightTimeLights.read(NightTimeLights.WORLD).remapTo(pop);
95     KmlWriter.write(nightTimeLights, out, "nighttimelights", PngWriter.createCoolToWarmColormap());
96
97     LatLonGrid countries = CountryPolygons.readGrid(CountryPolygons.WORLD_GRID).remapTo(pop);
98     KmlWriter.write(countries, out, "countries", PngWriter.createRandomColormap());
99
100    LabelResult primary = PrimaryCities.findPrimaryCities(pop, countries, out);
101    KmlWriter.write(primary.label, out, "primarycities", PngWriter.createRandomColormap());
102
103    LatLonGrid gdiGrid = WorldBankGDI.readGrid(WorldBankGDI.WORLD_GRID).remapTo(pop);
104    KmlWriter.write(gdiGrid, out, "gdism", PngWriter.createCoolToWarmColormap());
105
106    // obtain pattern
107    GdiPattern[] patterns = GdiPattern.findTrainingPattern(primary, pop, nightTimeLights, gdiGrid);
108    System.out.println("Population & Lighting & GDI \\\");
109    for (int i=1; i < patterns.length; ++i){
110        System.out.println(patterns[i]);
111    }
112
113    if (!SMALL){
114        write(patterns, out); // for R
115    }
116 }

```

6.41.2.3 String edu.ou.asgbook.datamining.GdiPattern.toString (String colsep, String linesep)

```

43                                     {
44     String result = "";
45     for (int i=0; i < data.length; ++i){
46         String sep = (i == data.length-1)? linesep : colsep;
47         result += format(data[i]) + sep;
48     }
49     return result;
50 }

```

6.41.2.4 Override String edu.ou.asgbook.datamining.GdiPattern.toString ()

```
39         {
40         return toString(" & ", "\\"); // for LaTeX
41     }
```

6.41.2.5 static void edu.ou.asgbook.datamining.GdiPattern.write ([GdiPattern\[\]](#) *patterns*, File *outdir*) throws IOException [static]

```
67                                                     {
68         PrintWriter writer = null;
69         try {
70             String name = outdir.getAbsolutePath() + "/gdipatterns.txt";
71             writer = new PrintWriter( new FileWriter(name) );
72             for (int i=1; i < patterns.length; ++i){
73                 writer.println(patterns[i].toString(" ", "")); // for R
74             }
75         } finally {
76             if (writer != null){
77                 writer.close();
78             }
79         }
80     }
```

6.42 edu.ou.asgbook.dataset.GlobalPopulation Class Reference

Reads the ASCII population data available at <http://sedac.ciesin.columbia.edu/gpw>.

Static Public Member Functions

- static [LatLonGrid read](#) (Reader inputFile, [DataTransform](#) t)
- static [LatLonGrid read](#) (File file, [DataTransform](#) t) throws IOException
reads data from a File.
- static [LatLonGrid read](#) (File file) throws IOException
- static void [main](#) (String[] args) throws Exception

Static Public Attributes

- static File [WORLD](#) = new File("data/popdensity/glp10ag.asc.gz")
- static File [NORTHAMERICA](#) = new File("data/popdensity/nap10ag.asc.gz")
- static File [NORTHAMERICA1990](#) = new File("data/popdensity/nap90ag.asc.gz")

Classes

- class [LinearScaling](#)
- class [LogScaling](#)

6.42.1 Detailed Description

Reads the ASCII population data available at <http://sedac.ciesin.columbia.edu/gpw>.

Author:

Valliappa.Lakshmanan

6.42.2 Member Function Documentation

6.42.2.1 static void edu.ou.asgbook.dataset.GlobalPopulation.main (String[] args) throws Exception [static]

```
57                                     {
58         // create output directory
```

```

59         File out = OutputDirectory.getDefault("globalpop");
60
61         // read input
62         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA).crop(90, 0, 0, 90);
63
64         // write out as image, for viewing
65         KmlWriter.write(popdensity, out, "popdensity", PngWriter.createCoolToWarmColormap());
66         KmlWriter.write(GlobalPopulation.read(GlobalPopulation.WORLD, new LogScaling()), out, "world", PngWriter.createCoolToWarmColormap());
67
68         // show impact of colormap and log scaling
69         KmlWriter.write(popdensity, out, "rainbow", PngWriter.createHotColormap());
70         popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new LogScaling());
71         KmlWriter.write(popdensity, out, "logdensity", PngWriter.createCoolToWarmColormap());
72         KmlWriter.write(popdensity, out, "lograinbow", PngWriter.createHotColormap());
73     }

```

6.42.2.2 static [LatLonGrid](#) edu.ou.asgbook.dataset.GlobalPopulation.read (File *file*) throws IOException [static]

```

53                                     {
54         return read(file, new LinearScaling());
55     }

```

6.42.2.3 static [LatLonGrid](#) edu.ou.asgbook.dataset.GlobalPopulation.read (File *file*, [DataTransform](#) *t*) throws IOException [static]

reads data from a File.

The File can be gzipped or uncompressed.

```

49                                     {
50         return EsriGrid.read(file, t);
51     }

```

6.42.2.4 static [LatLonGrid](#) edu.ou.asgbook.dataset.GlobalPopulation.read (Reader *inputFile*, [DataTransform](#) *t*) [static]

```

38                                     {
39         return EsriGrid.read(inputFile, t);
40     }

```


6.42.3 Member Data Documentation

6.42.3.1 File [edu.ou.asgbook.dataset.GlobalPopulation.NORTHAMERICA](#) =
new File("data/popdensity/nap10ag.asc.gz") [static]

6.42.3.2 File [edu.ou.asgbook.dataset.Global-
Population.NORTHAMERICA1990](#) = new
File("data/popdensity/nap90ag.asc.gz") [static]

6.42.3.3 File [edu.ou.asgbook.dataset.GlobalPopulation.WORLD](#) = new
File("data/popdensity/glp10ag.asc.gz") [static]

6.43 edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix Class Reference

Computes texture properties from a GLCM.

Public Types

- [EASTWARD](#)
- [SOUTHWARD](#)
- [NORTHEAST](#)
- [SOUTHEAST](#)
- enum [Direction](#) {
 [EASTWARD](#), [SOUTHWARD](#), [NORTHEAST](#), [SOUTHEAST](#),
 [xadd](#), [yadd](#) = [xadd](#) [yadd](#) }

Public Member Functions

- [GraylevelCooccurrenceMatrix](#) ([LatLonGrid](#) input, int x, int y, [Direction](#) dir, int hx, int hy, int min, int incr, int bins)
- double [computeUniformity](#) ()
- double [computeEntropy](#) ()
- double [computeMaximumProbability](#) ()
- double [computeDifferenceMoment](#) (int order)

Package Functions

- int [findBin](#) (int val, int missing, int min, int incr, int bins)

6.43.1 Detailed Description

Computes texture properties from a GLCM.

Author:

valliappa.lakshmanan

6.43.2 Member Enumeration Documentation

6.43.2.1 enum edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix::Direction

Enumerator:

EASTWARD
SOUTHWARD
NORTHEAST
SOUTHEAST
xadd
yadd

```

14                                     {
15     EASTWARD(0,1),
16     SOUTHWARD(1,0), // downward
17     NORTHEAST(-1,1),
18     SOUTHEAST(1,-1);
19     public final int xadd;
20     public final int yadd;
21     private Direction(int xadd, int yadd){
22         this.xadd = xadd;
23         this.yadd = yadd;
24     }
25 }
```

6.43.3 Constructor & Destructor Documentation

6.43.3.1 edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix.GraylevelCooccurrenceMatrix (LatLonGrid *input*, int *x*, int *y*, Direction *dir*, int *hx*, int *hy*, int *min*, int *incr*, int *bins*)

```

27
28     p = new double[bins+1][bins+1]; // last bin is for missing data
29     int N = 0;
30     for (int m=-hx; m <= hx; ++m){
31         for (int n=-hy; n <= hy; ++n){
32             int value1 = input.getMissing();
33             if ( input.isValid(x+m,y+n) ){
34                 value1 = input.getValue(x+m,y+n);
35             }
36             int value2 = input.getMissing();
37             if ( input.isValid(x+m+dir.xadd,y+n+dir.yadd) ){
38                 value2 = input.getValue(x+m+dir.xadd,y+n+dir.yadd);
39             }
40             int bin1 = findBin(value1,input.getMissing(),min,incr,bins);
41             int bin2 = findBin(value2,input.getMissing(),min,incr,bins);
42             p[bin1][bin2]++;
43             ++N;

```

```

44         }
45     }
46     for (int i=0; i < p.length; ++i) for (int j=0; j < p.length; ++j){
47         p[i][j] /= N;
48     }
49 }

```

6.43.4 Member Function Documentation

6.43.4.1 double edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix.computeDifferenceMoment (int *order*)

```

96         {
97     double result = 0;
98     int N = p.length;
99     for (int i=0; i < N; ++i) for (int j=0; j < N; ++j){
100         if (i != j){
101             result += p[i][j]*p[i][j]/Math.pow(Math.abs(i-j),order);
102         }
103     }
104     return result;
105 }

```

6.43.4.2 double edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix.computeEntropy ()

```

76         {
77     double result = 0;
78     int N = p.length;
79     for (int i=0; i < N; ++i) for (int j=0; j < N; ++j){
80         if (p[i][j] > 0){
81             result += p[i][j]*Math.log(p[i][j])/Math.log(2);
82         }
83     }
84     return result;
85 }

```

6.43.4.3 double edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix.computeMaximumProbability ()

```

87         {
88     double result = 0;
89     int N = p.length;
90     for (int i=0; i < N; ++i) for (int j=0; j < N; ++j){
91         result = Math.max(result, p[i][j]);
92     }
93     return result;
94 }

```

6.43.4.4 double edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix.computeUniformity ()

```
67                                     {
68     double result = 0;
69     int N = p.length;
70     for (int i=0; i < N; ++i) for (int j=0; j < N; ++j){
71         result += p[i][j]*p[i][j];
72     }
73     return result;
74 }
```

6.43.4.5 int edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix.findBin (int *val*, int *missing*, int *min*, int *incr*, int *bins*) [package]

```
50                                     {
51     if (val != missing && val >= min) {
52         int bin_no = (val - min) / incr;
53         // last bin is unbounded
54         if (bin_no >= bins)
55             bin_no = bins - 1;
56         return bin_no;
57     }
58     return bins; // for missing data
59 }
```

6.44 edu.ou.asgbook.thinning.HilditchSkeletonization Class Reference

Hilditch method of skeletonizing a grid.

Static Public Member Functions

- static [LatLonGrid](#) [findSkeleton](#) ([LatLonGrid](#) input, int thresh, File out) throws Exception
- static void [main](#) (String[] args) throws Exception

Classes

- class [State](#)

6.44.1 Detailed Description

Hilditch method of skeletonizing a grid.

Author:

v.lakshmanan

6.44.2 Member Function Documentation

6.44.2.1 static [LatLonGrid](#) edu.ou.asgbook.thinning.Hilditch-Skeletonization.findSkeleton ([LatLonGrid](#) input, int thresh, File out) throws Exception [static]

```

40
41         // threshold.  object=1 and background=0
42         LatLonGrid binaryImage = new SimpleThresholder(thresh).threshold(input);
43         if (out != null){
44             KmlWriter.write(binaryImage, out, "thresh", PngWriter.createCoolToWarmColormap());
45         }
46
47         final int nx = binaryImage.getNumLat();
48         final int ny = binaryImage.getNumLon();
49         int numChanges;
50         do {
51             // compute ap, bp
52             LatLonGrid ap = new LatLonGrid(nx,ny,-1,binaryImage.getNwCorner(),binaryImage.g
53             LatLonGrid bp = new LatLonGrid(nx,ny,-1,binaryImage.getNwCorner(),binaryImage.g
54             for (int i=1; i < (nx-1); ++i) for (int j=1; j < (ny-1); ++j){
55                 if ( binaryImage.getValue(i, j) > 0){

```

```

56         // find A(p) and B(p)
57         State state = new State( binaryImage.getValue(i-1,j-1) );
58         state.update( binaryImage.getValue(i-1, j) );
59         state.update( binaryImage.getValue(i-1, j+1) );
60         state.update( binaryImage.getValue(i, j+1) );
61         state.update( binaryImage.getValue(i+1, j+1) );
62         state.update( binaryImage.getValue(i+1, j) );
63         state.update( binaryImage.getValue(i+1, j-1) );
64         state.update( binaryImage.getValue(i, j-1) );
65         state.update( binaryImage.getValue(i-1, j-1) );
66         ap.setValue(i,j, state.ap);
67         bp.setValue(i,j, state.bp);
68     }
69 }
70
71 // peel off pixel?
72 numChanges = 0;
73 LatLonGrid after = LatLonGrid.copyOf(binaryImage);
74 for (int i=1; i < (nx-1); ++i) for (int j=1; j < (ny-1); ++j){
75     if ( ap.getValue(i,j) == 1 && bp.getValue(i,j) >= 2 && bp.getValue(i,j) <= 6){
76         if ( ap.getValue(i-1,j) == 0 ||
77             binaryImage.getValue(i-1,j) == 0 ||
78             binaryImage.getValue(i,j+1) == 0 ||
79             binaryImage.getValue(i,j-1) == 0 ){
80             if ( ap.getValue(i,j+1) == 0 ||
81                 binaryImage.getValue(i-1,j) == 0 ||
82                 binaryImage.getValue(i,j+1) == 0 ||
83                 binaryImage.getValue(i+1,j) == 0 ){
84                 // peel
85                 after.setValue(i,j, 0);
86                 ++numChanges;
87             }
88         }
89     }
90 }
91 binaryImage = after;
92 System.out.println(numChanges + " pixels peeled off in iteration");
93 } while (numChanges > 0);
94
95 return binaryImage;
96 }

```

6.44.2.2 static void edu.ou.asgbook.thinning.HilditchSkeletonization.main (String[] args) throws Exception [static]

```

98     {
99         File out = OutputDirectory.getDefault("hilditchskeleton");
100         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulationReader());
101         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
102
103         popdensity = new DilateErodeFilter(2,3).filter(popdensity);
104         popdensity = new ErodeDilateFilter(2,3).filter(popdensity);
105         KmlWriter.write(popdensity, out, "filledin", PngWriter.createCoolToWarmColormap());
106
107         LatLonGrid result = findSkeleton(popdensity, 300, out);

```

```
108         result.setMissing(0); // to make the ls pop out
109         KmlWriter.write(result, out, "skel", PngWriter.createCoolToWarmColormap());
110     }
```


6.45 edu.ou.asgbook.histogram.Histogram Class Reference

A histogram is an empirical probability distribution.

Public Member Functions

- [Histogram](#) (int min, int incr, int nbins)
Values below min are ignored but the last bin is unbounded.
- int [getMin](#) ()
- int [getIncr](#) ()
- int[] [getHist](#) ()
- void [update](#) (LatLonGrid data)
- int [getCenterValue](#) (int bin_no, int missing)
- int [getBinNumber](#) (int val, int missing)
points outside the histogram have bin number of -1
- Override String [toString](#) ()
- float[] [calcProb](#) ()
- int [getNumBins](#) ()

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.45.1 Detailed Description

A histogram is an empirical probability distribution.

Author:

v.lakshmanan

6.45.2 Constructor & Destructor Documentation

6.45.2.1 edu.ou.asgbook.histogram.Histogram.Histogram (int *min*, int *incr*, int *nbins*)

Values below min are ignored but the last bin is unbounded.

Parameters:*min**incr**nbins*

```

32                                     {
33         super();
34         this.min = min;
35         this.incr = incr;
36         this.hist = new int[nbins];
37     }

```

6.45.3 Member Function Documentation**6.45.3.1 float [] edu.ou.asgbook.histogram.Histogram.calcProb ()**

```

99                                     {
100         float[] prob = new float[hist.length];
101         int tot = 0;
102         for (int i = 0; i < hist.length; ++i) {
103             tot += hist[i];
104         }
105         if (tot > 0) {
106             for (int i = 0; i < hist.length; ++i) {
107                 prob[i] = hist[i] / (float) tot;
108             }
109         }
110         return prob;
111     }

```

6.45.3.2 int edu.ou.asgbook.histogram.Histogram.getBinNumber (int *val*, int *missing*)

points outside the histogram have bin number of -1

```

72                                     {
73         if (val != missing && val >= min) {
74             int bin_no = (val - min) / incr;
75             // last bin is unbounded
76             if (bin_no >= hist.length)
77                 bin_no = hist.length - 1;
78             return bin_no;
79         }
80         return -1;
81     }

```

6.45.3.3 int edu.ou.asgbook.histogram.Histogram.getCenterValue (int *bin_no*, int *missing*)

```
64                                     {
65     if (bin_no < 0){
66         return missing;
67     }
68     return min + bin_no*incr + incr/2;
69 }
```

6.45.3.4 int [] edu.ou.asgbook.histogram.Histogram.getHist ()

```
47                                     {
48     return hist;
49 }
```

6.45.3.5 int edu.ou.asgbook.histogram.Histogram.getIncr ()

```
43                                     {
44     return incr;
45 }
```

6.45.3.6 int edu.ou.asgbook.histogram.Histogram.getMin ()

```
39                                     {
40     return min;
41 }
```

6.45.3.7 int edu.ou.asgbook.histogram.Histogram.getNumBins ()

```
137                                     {
138     return hist.length;
139 }
```

6.45.3.8 static void edu.ou.asgbook.histogram.Histogram.main (String[] *args*) throws Exception [static]

```
113                                     {
114     // create output directory
115     File outdir = OutputDirectory.getDefault("hist");
116
117     // read input
118     LatLonGrid conus = SurfaceAlbedo.read(SurfaceAlbedo.CONUS, 100);
```

```

119
120         // find histogram
121         final int MIN = 0;
122         final int MAX = 30;
123         for (int incr = 1; incr < 10; incr += 2) {
124             Histogram hist = new Histogram(MIN, incr, (MAX - MIN) / incr);
125             hist.update(conus);
126             System.out.println("INCR=" + incr + " nbins=" + hist.hist.length);
127             System.out.println(hist);
128             String filename = outdir.getAbsolutePath() + "/hist_" + incr
129                 + ".txt";
130             PrintWriter writer = new PrintWriter(new FileWriter(filename));
131             writer.println(hist);
132             writer.close();
133             System.out.println("Wrote to " + filename);
134         }
135     }

```

6.45.3.9 Override String edu.ou.asgbook.histogram.Histogram.toString ()

```

84         {
85         StringBuilder sb = new StringBuilder();
86         for (int i = 0; i < hist.length; ++i) {
87             int sval = min + i * incr;
88             int eval = sval + incr;
89             sb.append(sval);
90             sb.append(" ");
91             sb.append(eval);
92             sb.append(" ");
93             sb.append(hist[i]);
94             sb.append("\n");
95         }
96         return sb.toString();
97     }

```

6.45.3.10 void edu.ou.asgbook.histogram.Histogram.update ([LatLonGrid](#) data)

```

51         {
52         final int nrows = data.getNumLat();
53         final int ncols = data.getNumLon();
54         for (int i = 0; i < nrows; ++i)
55             for (int j = 0; j < ncols; ++j) {
56                 int val = data.getValue(i, j);
57                 int bin_no = getBinNumber(val, data.getMissing());
58                 if (bin_no != -1 ) {
59                     hist[bin_no]++;
60                 }
61             }
62     }

```

6.46 edu.ou.asgbook.histogram.HistogramBinSelection Class Reference

Tries out different values for the number of bins and replaces each pixel value by the center of its bin.

Static Public Member Functions

- static [LatLonGrid](#) [band](#) ([LatLonGrid](#) data, [Histogram](#) hist)
replaces each pixel by the center of its bin
- static [Histogram](#) [createBasedOnRange](#) ([LatLonGrid](#) data)
Based on range.
- static [Histogram](#) [createHighestResolution](#) ([LatLonGrid](#) data)
Highest resolution possible.
- static [Histogram](#) [createBasedOnStdDev](#) ([LatLonGrid](#) data)
Based on range.
- static [Histogram](#) [createBasedOnNumSamples](#) ([LatLonGrid](#) data)
Based on range.
- static void [main](#) (String[] args) throws Exception

6.46.1 Detailed Description

Tries out different values for the number of bins and replaces each pixel value by the center of its bin.

Author:

valliappa.lakshmanan

6.46.2 Member Function Documentation

- 6.46.2.1** static [LatLonGrid](#) [edu.ou.asgbook.histogram.HistogramBinSelection.band](#) ([LatLonGrid](#) data, [Histogram](#) hist)
[static]

replaces each pixel by the center of its bin

```

29                                     {
30         LatLonGrid result = LatLonGrid.copyOf(data);
31         int nrows = result.getNumLat();
32         int ncols = result.getNumLon();
33         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
34             int bin_no = hist.getBinNumber(data.getValue(i,j), data.getMissing());
35             int cval = hist.getCenterValue(bin_no, data.getMissing());
36             result.setValue(i,j, cval);
37         }
38         return result;
39     }

```

6.46.2.2 static [Histogram](#) edu.ou.asgbook.histogram.HistogramBin- Selection.createBasedOnNumSamples ([LatLonGrid](#) data) [static]

Based on range.

```

125                                     {
126         int min = data.getMissing();
127         int max = data.getMissing();
128         int N = 0;
129         int nrows = data.getNumLat();
130         int ncols = data.getNumLon();
131         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
132             int val = data.getValue(i,j);
133             if ( val != data.getMissing() ){
134                 ++N;
135                 if (min == data.getMissing() || val < min){
136                     min = val;
137                 }
138                 if (max == data.getMissing() || val > max){
139                     max = val;
140                 }
141             }
142         }
143         int nbins = 1 + (int) Math.round(Math.sqrt(N));
144         System.out.println("Based on N="+ N + ", nbins=" + nbins);
145         int incr = (max-min)/nbins;
146         if (incr == 0) incr = 1;
147         Histogram hist = new Histogram(min,incr,nbins);
148         hist.update(data);
149         return hist;
150     }

```

6.46.2.3 static [Histogram](#) edu.ou.asgbook.histogram.Histogram- BinSelection.createBasedOnRange ([LatLonGrid](#) data) [static]

Based on range.

```

42                                     {
43         // find the range
44         int min = data.getMissing();
45         int max = data.getMissing();
46         int nrows = data.getNumLat();
47         int ncols = data.getNumLon();
48         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
49             int val = data.getValue(i,j);
50             if ( val != data.getMissing() ){
51                 if (min == data.getMissing() || val < min){
52                     min = val;
53                 }
54                 if (max == data.getMissing() || val > max){
55                     max = val;
56                 }
57             }
58         }
59         int nbins = 1 + (int) Math.round(Math.log(max-min)/Math.log(2));
60         System.out.println("Based on range: min=" + min + " max=" + max + ", nbins=" + nbins);
61         int incr = (max-min)/nbins;
62         if (incr == 0) incr = 1;
63         Histogram hist = new Histogram(min,incr,nbins);
64         hist.update(data);
65         return hist;
66     }

```

6.46.2.4 static [Histogram](#) edu.ou.asgbook.histogram.Histogram-BinSelection.createBasedOnStdDev ([LatLonGrid](#) data)

[static]

Based on range.

```

95                                     {
96         ScalarStatistic stat = new ScalarStatistic();
97         int min = data.getMissing();
98         int max = data.getMissing();
99         int nrows = data.getNumLat();
100        int ncols = data.getNumLon();
101        for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
102            int val = data.getValue(i,j);
103            if ( val != data.getMissing() ){
104                stat.update(val);
105                if (min == data.getMissing() || val < min){
106                    min = val;
107                }
108                if (max == data.getMissing() || val > max){
109                    max = val;
110                }
111            }
112        }
113        double sigma = stat.getStdDeviation();
114        int N = stat.getNumSamples();
115        int nbins = 1 + (int) Math.round(3.5*sigma/Math.pow(N, 1.0/3));
116        System.out.println("Based on sigma=" + sigma + " N=" + N + ", nbins=" + nbins);

```

```

117         int incr = (max-min)/nbins;
118         if (incr == 0) incr = 1;
119         Histogram hist = new Histogram(min,incr,nbins);
120         hist.update(data);
121         return hist;
122     }

```

6.46.2.5 static [Histogram](#) edu.ou.asgbook.histogram.HistogramBinSelection.createHighestResolution ([LatLonGrid](#) data) [static]

Highest resolution possible.

```

69                                                     {
70         // find the range
71         int min = data.getMissing();
72         int max = data.getMissing();
73         int nrows = data.getNumLat();
74         int ncols = data.getNumLon();
75         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
76             int val = data.getValue(i,j);
77             if ( val != data.getMissing() ){
78                 if (min == data.getMissing() || val < min){
79                     min = val;
80                 }
81                 if (max == data.getMissing() || val > max){
82                     max = val;
83                 }
84             }
85         }
86         System.out.println("full resolution: min="+ min + " max="+ max + ", incr=1");
87         final int incr = 1;
88         int nbins = (max-min)+1;
89         Histogram hist = new Histogram(min,incr,nbins);
90         hist.update(data);
91         return hist;
92     }

```

6.46.2.6 static void edu.ou.asgbook.histogram.HistogramBinSelection.main (String[] args) throws Exception [static]

```

152                                                     {
153         // create output directory
154         File outdir = OutputDirectory.getDefault("histbin");
155
156         // read input
157         LatLonGrid conus = SurfaceAlbedo.read(SurfaceAlbedo.CONUS, 100);
158
159         // find histogram in three different ways
160         Map<String, Histogram> map = new HashMap<String,Histogram>();
161         map.put("range", HistogramBinSelection.createBasedOnRange(conus));

```

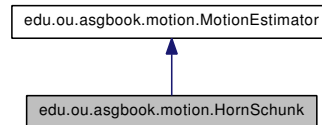


```
162         map.put("numsamples", HistogramBinSelection.createBasedOnNumSamples(conus));
163         map.put("stddev", HistogramBinSelection.createBasedOnStdDev(conus));
164
165         for (Map.Entry<String, Histogram> entry : map.entrySet()) {
166             Histogram hist = entry.getValue();
167             String name = entry.getKey();
168
169             String filename = outdir.getAbsolutePath() + "/hist_" + name
170                 + ".txt";
171             PrintWriter writer = new PrintWriter(new FileWriter(filename));
172             writer.println(hist);
173             writer.close();
174             System.out.println("Wrote to " + filename);
175
176             LatLonGrid banded = HistogramBinSelection.band(conus, hist);
177             KmlWriter.write(banded, outdir, name, PngWriter.createCoolToWarmColormap());
178         }
179     }
```

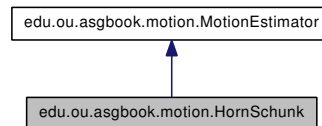
6.47 edu.ou.asgbook.motion.HornSchunk Class Reference

Horn-Schunk optical flow method of motion estimation.

Inheritance diagram for edu.ou.asgbook.motion.HornSchunk:



Collaboration diagram for edu.ou.asgbook.motion.HornSchunk:



Public Member Functions

- Override `Pair< LatLonGrid, LatLonGrid > compute (LatLonGrid data0, LatLonGrid data1, File outdir)`
returns motion in the two directions.

Static Public Member Functions

- static void `main (String[] args)` throws Exception

Static Public Attributes

- static final int `MOT_SCALE = 10`

6.47.1 Detailed Description

Horn-Schunk optical flow method of motion estimation.

Author:

v.lakshmanan

6.47.2 Member Function Documentation

6.47.2.1 Override Pair<LatLonGrid, LatLonGrid>

edu.ou.asgbook.motion.HornSchunk.compute ([LatLonGrid data0](#),
[LatLonGrid data1](#), [File outdir](#))

returns motion in the two directions.

The first one is north to south and the second one is east to west. The data is aligned to second time frame. The output dir is used for intermediate products and may be null.

Implements [edu.ou.asgbook.motion.MotionEstimator](#).

```

32
33         // Grids we need. initialize all of them at zero
34         final int nrows = data1.getNumLat();
35         final int ncols = data1.getNumLon();
36         LatLonGrid I_x = new LatLonGrid(nrows, ncols, 0, data1.getNwCorner(), data1.getLatRes(), data1.getLonRes());
37         LatLonGrid I_y = LatLonGrid.copyOf(I_x);
38         LatLonGrid I_t = LatLonGrid.copyOf(I_x);
39         LatLonGrid u = LatLonGrid.copyOf(I_x);
40         LatLonGrid v = LatLonGrid.copyOf(I_x);
41
42         // compute gradient of intensity in x, y and t directions
43         for (int i=1; i < nrows-1; ++i) for (int j=1; j < ncols-1; ++j){
44             int i_t = data1.getValue(i,j) - data0.getValue(i,j); // time
45             int i_x = data1.getValue(i,j) - data1.getValue(i-1,j); // lat
46             int i_y = data1.getValue(i,j) - data1.getValue(i,j-1); // lon
47             I_x.setValue(i,j, i_x);
48             I_y.setValue(i,j, i_y);
49             I_t.setValue(i,j, i_t);
50         }
51
52         // write intermediates
53         if (outdir != null){
54             try {
55                 KmlWriter.write(I_x, outdir, "I_x", PngWriter.createCoolToWarmColormap());
56                 KmlWriter.write(I_y, outdir, "I_y", PngWriter.createCoolToWarmColormap());
57                 KmlWriter.write(I_t, outdir, "I_t", PngWriter.createCoolToWarmColormap());
58             } catch (Exception e) {
59                 e.printStackTrace();
60             }
61         }
62
63         // now iterate
64         for (int iter=0; iter < MAX_ITER; ++iter){
65             // compute meanu, meanv
66             LatLonGrid meanu, meanv;
67             if ( iter == 0 ){
68                 meanu = LatLonGrid.copyOf(u);
69                 meanv = LatLonGrid.copyOf(v);
70             } else {
71                 ConvolutionFilter boxcar = new ConvolutionFilter(ConvolutionFilter.boxcar(2*SM_HALFSIZE));
72                 meanu = boxcar.smooth(u);
73                 meanv = boxcar.smooth(v);
74             }

```

```

75
76         for (int i=1; i < nrows-1; ++i) for (int j=1; j < ncols-1; ++j){
77             double u_k = meanu.getValue(i, j)/(double)MOT_SCALE;
78             double v_k = meanv.getValue(i, j)/(double)MOT_SCALE;
79             int i_x = I_x.getValue(i, j);
80             int i_y = I_y.getValue(i, j);
81             int i_t = I_t.getValue(i, j);
82             double corr = (i_x*u_k + i_y*v_k + i_t) / (ALPHASQ + i_x*i_x + i_y*i_y);
83             u.setValue(i, j, (int) Math.round((u_k - i_x*corr)*MOT_SCALE));
84             v.setValue(i, j, (int) Math.round((v_k - i_y*corr)*MOT_SCALE));
85         }
86
87         if (outdir != null && iter == 0 || iter == 1 || iter == MAX_ITER/2){
88             try {
89                 KmlWriter.write(u, outdir, "motionNS_"+iter, PngWriter.createCoolToWarmColorMap());
90                 KmlWriter.write(v, outdir, "motionEW_"+iter, PngWriter.createCoolToWarmColorMap());
91             } catch (Exception e) {
92                 e.printStackTrace();
93             }
94         }
95     }
96
97     return new Pair<LatLonGrid, LatLonGrid>(u, v);
98 }

```

6.47.2.2 static void edu.ou.asgbook.motion.HornSchunk.main (String[] args) throws Exception [static]

```

100
101         // create output directory
102         File out = OutputDirectory.getDefault("hornschunk");
103
104         // read
105         File f = new File("data/seviri");
106         Pair<LatLonGrid, Date>[] grids = SeviriInfraredTemperature.readAll(f);
107
108         // do alg
109         MotionEstimator alg = new HornSchunk();
110         Pair<LatLonGrid, LatLonGrid> motion = alg.compute(grids[0].first, grids[1].first, out);
111
112         // write
113         KmlWriter.write(motion.first, out, "opticflow_u", PngWriter.createCoolToWarmColorMap());
114         KmlWriter.write(motion.second, out, "opticflow_v", PngWriter.createCoolToWarmColorMap());
115
116         // align and compute difference
117         LatLonGrid diff = new AlignAndDifference().compute(grids[0].first, grids[1].first, out);
118         KmlWriter.write(diff, out, "opticflow_diff", PngWriter.createCoolToWarmColorMap());
119     }

```

6.47.3 Member Data Documentation

6.47.3.1 **final int** [edu.ou.asgbook.motion.HornSchunk.MOT_SCALE](#) = 10
[static]

6.48 edu.ou.asgbook.transforms.HoughTransform Class Reference

Finds lines in image.

Public Member Functions

- `Line[] findLines (LatLonGrid grid, int datathresh)`
Find best lines that connect points > thresh.

Static Public Member Functions

- static void `main` (String[] args) throws Exception

Classes

- class `Line`

6.48.1 Detailed Description

Finds lines in image.

Author:

v.lakshmanan

6.48.2 Member Function Documentation

6.48.2.1 `Line [] edu.ou.asgbook.transforms.HoughTransform.findLines (LatLonGrid grid, int datathresh)`

Find best lines that connect points > thresh.

Parameters:

grid
datathresh

Returns:

```

99                                     {
100         int maxr = grid.getNumLat() + grid.getNumLon();
101         int numr = (int) Math.round(maxr / DELTA_RHO);
102         int numtheta = (int) Math.round(360 / DELTA_THETA);
103
104         // update vote
105         Line[] lines = new Line[numr * numtheta];
106         for (int i=0; i < lines.length; ++i){
107             lines[i] = new Line();
108         }
109         for (int i = 0; i < grid.getNumLat(); ++i) {
110             for (int j = 0; j < grid.getNumLon(); ++j) {
111                 if (grid.getValue(i, j) > datathresh) {
112                     // use this point to cast votes ...
113                     for (int theta = 0; theta < numtheta; ++theta) {
114                         double theta_radians = (theta * DELTA_THETA * Math.PI) / 180.0;
115                         double rho = i * Math.cos(theta_radians) + j
116                             * Math.sin(theta_radians);
117                         int r = (int) Math.round(rho / DELTA_RHO);
118                         if (r >= 0 && r < maxr) {
119                             Line line = lines[r * numtheta + theta];
120                             line.rho = rho;
121                             line.theta = theta_radians;
122                             line.numVotes++;
123                             line.x1 = Math.min(line.x1, i);
124                             line.x2 = Math.max(line.x2, i);
125                             line.y1 = Math.min(line.y1, j);
126                             line.y2 = Math.max(line.y2, j);
127                         }
128                     }
129                 }
130             }
131         }
132
133         // sort the lines by vote
134         Arrays.sort(lines);
135         return lines;
136     }

```

6.48.2.2 static void edu.ou.asgbook.transforms.HoughTransform.main (String[] args) throws Exception [static]

```

138                                     {
139         // create output directory
140         File out = OutputDirectory.getDefault("hough");
141
142         // read input
143         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulat
144         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
145
146         // fill in
147         popdensity = new DilateErodeFilter(2,3).filter(popdensity);
148         popdensity = new ErosionFilter(3).filter(popdensity);
149         KmlWriter.write(popdensity, out, "filledin", PngWriter.createCoolToWarmColormap());
150     }

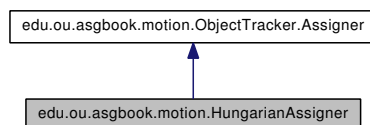
```

```
151         // skeletonize
152         LatLonGrid skel = HilditchSkeletonization.findSkeleton(popdensity, 300, out);
153         KmlWriter.write(skel, out, "skel", PngWriter.createCoolToWarmColormap());
154
155         // find lines
156         HoughTransform hough = new HoughTransform();
157         HoughTransform.Line[] lines = hough.findLines(skel, 0);
158         final int NBEST = 3;
159         for (int i=0; i < Math.min(lines.length,NBEST); ++i){ // NBEST lines
160             HoughTransform.Line line = lines[i];
161             System.out.println(line);
162             List<Pixel> pixels = line.computePixels(popdensity.getNumLat(), popdensity.getNumLon());
163             for (Pixel p : pixels){
164                 popdensity.setValue(p.getX(), p.getY(), 1000);
165             }
166         }
167         KmlWriter.write(popdensity, out, "lines",
168             PngWriter.createCoolToWarmColormap());
169     }
```

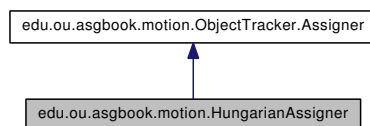

6.49 edu.ou.asgbook.motion.HungarianAssigner Class Reference

Optimal assignment algorithm.

Inheritance diagram for edu.ou.asgbook.motion.HungarianAssigner:



Collaboration diagram for edu.ou.asgbook.motion.HungarianAssigner:



Public Member Functions

- Override `int[] getAssignments (int[][] cost, int maxcost)`

Static Public Member Functions

- static void `main (String[] args)` throws Exception

Classes

- class **HungarianMatch**

6.49.1 Detailed Description

Optimal assignment algorithm.

Author:

valliappa.lakshmanan

6.49.2 Member Function Documentation

6.49.2.1 Override `int [] edu.ou.asgbook.motion.HungarianAssigner.getAssignments (int cost[][], int maxcost)`

Implements [edu.ou.asgbook.motion.ObjectTracker.Assigner](#).

```

31                                     {
32     // intialize result to be all unassigned
33     int[] result = new int[cost.length];
34     for (int i=0; i < result.length; ++i){
35         result[i] = -1;
36     }
37
38     // if number of objects is zero, then can't do any assignment
39     if (cost.length == 0 || cost[0].length == 0){
40         return result;
41     }
42
43     if (cost[0].length < cost.length){
44         // rotate so that we have more columns than rows
45         int[][] rot = new int[ cost[0].length ][ cost.length ];
46         for (int i=0; i < cost.length; ++i){
47             for (int j=0; j < cost[i].length; ++j){
48                 rot[j][i] = cost[i][j];
49             }
50         }
51         // do the assignment process on rotated cost function
52         int[] col_to_row = getAssignments(rot, maxcost);
53         // fix result: we need row_to_col
54         for (int col = 0; col < col_to_row.length; ++col){
55             int row = col_to_row[col];
56             if (row >= 0){
57                 result[row] = col;
58             }
59         }
60         return result;
61     }
62
63     // threshold just in case some cost > maxcost
64     for (int i=0; i < cost.length; ++i){
65         for (int j=0; j < cost[i].length; ++j){
66             if (cost[i][j] > maxcost){
67                 cost[i][j] = maxcost;
68             }
69         }
70     }
71     HungarianMatch match = new HungarianMatch(cost);
72     match.do_step1();
73     match.do_step2();
74     match.do_step3();
75     for (int i=0; i < cost.length; ++i){
76         for (int j=0; j < cost[i].length; ++j){
77             if (match.starred_zero[i][j] && cost[i][j] < maxcost ){
78                 result[i] = j;
79             }

```

```
80         }
81     }
82     return result;
83 }
```

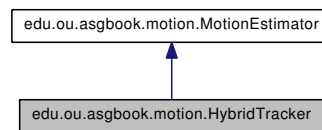
6.49.2.2 static void edu.ou.asgbook.motion.HungarianAssigner.main (String[] args) throws Exception [static]

```
313                                     {
314     // create output directory
315     File out = OutputDirectory.getDefault("hungarian");
316
317     // read
318     File f = new File("data/seviri");
319     Pair<LatLonGrid,Date>[] grids = SeviriInfraredTemperature.readAll(f);
320
321     // do alg
322     Segmenter seg = new ObjectTracker.SimpleSegmenter(100, 110, 1000);
323     ObjectTracker alg = new ObjectTracker(seg, new ObjectTracker.CentroidDistance(), new HungarianAssigner());
324     MedianFilter smoother = new MedianFilter(10);
325     LatLonGrid grid0 = smoother.filter(grids[0].first);
326     LatLonGrid grid1 = smoother.filter(grids[1].first);
327     Pair<LatLonGrid,LatLonGrid> motion = alg.compute(grid0, grid1, out);
328
329     // write
330     SaturateFilter filter = new SaturateFilter(-150, 150);
331     LatLonGrid u = filter.filter(motion.first);
332     LatLonGrid v = filter.filter(motion.second);
333     KmlWriter.write(u, out, "hungarian_u", PngWriter.createCoolToWarmColormap());
334     KmlWriter.write(v, out, "hungarian_v", PngWriter.createCoolToWarmColormap());
335 }
```

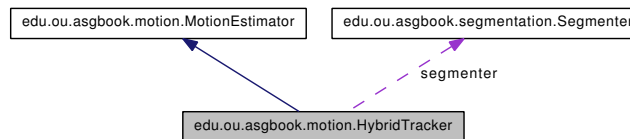
6.50 edu.ou.asgbook.motion.HybridTracker Class Reference

Estimates motion by finding cross-correlation of objects in one frame to the pixels in the previous frame.

Inheritance diagram for edu.ou.asgbook.motion.HybridTracker:



Collaboration diagram for edu.ou.asgbook.motion.HybridTracker:



Public Member Functions

- [HybridTracker](#) ([Segmenter](#) seg, int maxmotionx, int maxmotiony)
- Override Pair< [LatLonGrid](#), [LatLonGrid](#) > [compute](#) ([LatLonGrid](#) data0, [LatLonGrid](#) data1, File outdir)

returns motion in the two directions.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

Classes

- class [Centroid](#)

6.50.1 Detailed Description

Estimates motion by finding cross-correlation of objects in one frame to the pixels in the previous frame.

Author:

v.lakshmanan

6.50.2 Constructor & Destructor Documentation

6.50.2.1 edu.ou.asgbook.motion.HybridTracker.HybridTracker (Segmenter seg, int maxmotionx, int maxmotiony)

```

37                                     {
38         MAX_U = maxmotionx;
39         MAX_V = maxmotiony;
40         segmenter = seg;
41     }
```

6.50.3 Member Function Documentation

6.50.3.1 Override Pair<LatLonGrid, LatLonGrid> edu.ou.asgbook.motion.HybridTracker.compute (LatLonGrid data0, LatLonGrid data1, File outdir)

returns motion in the two directions.

The first one is north to south and the second one is east to west. The data is aligned to second time frame. The output dir is used for intermediate products and may be null.

Implements [edu.ou.asgbook.motion.MotionEstimator](#).

```

52                                     {
53         LabelResult objects1 = segmenter.label(data1);
54         if (outdir != null){
55             try {
56                 KmlWriter.write(objects1.label, outdir, "hybobjects1", PngWriter.createRandomColormap());
57             } catch (Exception e) {
58                 e.printStackTrace();
59             }
60         }
61
62         // find motion for each region and apply it to all pixels for that region
63         Pixel[][] regions = RegionProperty.getPixelsInRegions(data1, objects1);
64         LatLonGrid u = new LatLonGrid(data0.getNumLat(), data0.getNumLon(), 0, data0.getNwCorner(), data0.getNeCorner());
65         LatLonGrid v = LatLonGrid.copyOf(u);
66         RegionProperty[] regprop = RegionProperty.compute(objects1, data1);
67         List<Centroid> centroids = new ArrayList<Centroid>();
68         for (int reg=1; reg < regions.length; ++reg){
```

```

69         Pair<Integer,Integer> motion = computeMotion(regions[reg], data0);
70         int motx = motion.first;
71         int moty = motion.second;
72         Centroid c = new Centroid();
73         c.cx = regprop[reg].getCx();
74         c.cy = regprop[reg].getCy();
75         c.motx = motx;
76         c.moty = moty;
77         c.size = regprop[reg].getSize();
78         centroids.add(c);
79         for (Pixel p : regions[reg]){
80             u.setValue(p.getX(), p.getY(), motx);
81             v.setValue(p.getX(), p.getY(), moty);
82         }
83     }
84
85     if (outdir != null){
86         try {
87             KmlWriter.write(u, outdir, "u_beforeinterp", PngWriter.createCoolToWarmColorMap());
88             KmlWriter.write(v, outdir, "v_beforeinterp", PngWriter.createCoolToWarmColorMap());
89         } catch (Exception e) {
90             e.printStackTrace();
91         }
92     }
93
94     // interpolate inbetween regions if you have enough of them ...
95     if ( centroids.size() > 1 ){
96         LatLonGrid interpu = LatLonGrid.copyOf(u);
97         LatLonGrid interpv = LatLonGrid.copyOf(v);
98         for (int i=0; i < interpu.getNumLat(); ++i) for (int j=0; j < interpu.getNumLon(); ++j){
99             double totu = 0;
100             double totv = 0;
101             double totwt = 0;
102             for (Centroid c : centroids){
103                 double distx = c.cx - i;
104                 double disty = c.cy - j;
105                 double distsq = distx*distx + disty*disty;
106                 double wt = c.size * 1.0/(distsq*distsq + 0.0001); // 1/r^2
107                 totu += c.motx * wt;
108                 totv += c.moty * wt;
109                 totwt += wt;
110             }
111             interpu.setValue(i, j, (int) Math.round(totu/totwt));
112             interpv.setValue(i, j, (int) Math.round(totv/totwt));
113         }
114
115         if (outdir != null){
116             try {
117                 KmlWriter.write(interpu, outdir, "u_interp", PngWriter.createCoolToWarmColorMap());
118                 KmlWriter.write(interpv, outdir, "v_interp", PngWriter.createCoolToWarmColorMap());
119             } catch (Exception e) {
120                 e.printStackTrace();
121             }
122         }
123
124         return new Pair<LatLonGrid,LatLonGrid>(interpu,interp);
125     }

```

```
126
127         return new Pair<LatLonGrid, LatLonGrid>(u, v);
128     }
```

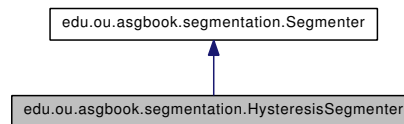
6.50.3.2 static void edu.ou.asgbook.motion.HybridTracker.main (String[] args) throws Exception [static]

```
153                                     {
154         // create output directory
155         File out = OutputDirectory.getDefault("hybridtracker");
156
157         // read
158         File f = new File("data/seviri");
159         Pair<LatLonGrid, Date>[] grids = SeviriInfraredTemperature.readAll(f);
160
161         // do alg
162         Segmenter seg = new ObjectTracker.SimpleSegmenter(100, 110, 100);
163         MotionEstimator alg = new HybridTracker( seg, 20, 20 );
164         MedianFilter smoother = new MedianFilter(10);
165         LatLonGrid grid0 = smoother.filter(grids[0].first);
166         LatLonGrid grid1 = smoother.filter(grids[1].first);
167         Pair<LatLonGrid, LatLonGrid> motion = alg.compute(grid0, grid1, out);
168
169         LatLonGrid diff = new AlignAndDifference().compute(grids[0].first, grids[1].first, motion);
170         KmlWriter.write(diff, out, "hybriddiff", PngWriter.createCoolToWarmColormap());
171
172         // write
173         SaturateFilter filter = new SaturateFilter(-15, 15);
174         LatLonGrid u = filter.filter(motion.first);
175         LatLonGrid v = filter.filter(motion.second);
176         KmlWriter.write(u, out, "opticflow_u", PngWriter.createCoolToWarmColormap());
177         KmlWriter.write(v, out, "opticflow_v", PngWriter.createCoolToWarmColormap());
178     }
```

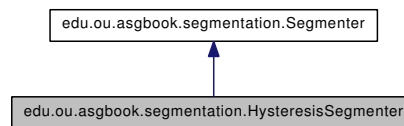
6.51 edu.ou.asgbook.segmentation.Hysteresis-Segmenter Class Reference

Objects consist of pixels that are $> \text{thresh2}$ but have at least one pixel $> \text{thresh1}$.

Inheritance diagram for edu.ou.asgbook.segmentation.HysteresisSegmenter:



Collaboration diagram for edu.ou.asgbook.segmentation.HysteresisSegmenter:



Public Member Functions

- [HysteresisSegmenter](#) (int thresh1, int thresh2)
- Override [LabelResult label](#) ([LatLonGrid](#) data)

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.51.1 Detailed Description

Objects consist of pixels that are $> \text{thresh2}$ but have at least one pixel $> \text{thresh1}$.

Author:

v.lakshmanan

6.51.2 Constructor & Destructor Documentation

6.51.2.1 edu.ou.asgbook.segmentation.HysteresisSegmenter.HysteresisSegmenter (int *thresh1*, int *thresh2*)

```

22                                     {
23         super();
24         this.t1 = thresh1;
25         this.t2 = thresh2;
26         if (t1 < t2){
27             // swap
28             int t = t1;
29             t1 = t2;
30             t2 = t;
31         }
32     }

```

6.51.3 Member Function Documentation

6.51.3.1 Override [LabelResult](#) edu.ou.asgbook.segmentation.HysteresisSegmenter.label ([LatLonGrid](#) *data*) [virtual]

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

.. All pixels > thresh are part of an object.

Implements [edu.ou.asgbook.segmentation.Segmenter](#).

```

35                                     {
36         final int UNSET = 0;
37         int nrows = data.getNumLat();
38         int ncols = data.getNumLon();
39         LatLonGrid label = new LatLonGrid(nrows,ncols,0,data.getNwCorner(),data.getLatRes(),data.getLonRes());
40         // label.fill(UNSET); java default is to zero-out arrays
41         int regno = 0;
42         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
43             if ( data.getValue(i, j) > t1 && label.getValue(i, j) == UNSET ){
44                 ++regno;
45                 RegionGrowing.growRegion(i,j, data, t2, label, regno);
46             }
47         }
48         System.out.println("Found " + (regno+1) + " objects");
49         return new LabelResult(label, regno);
50     }

```

6.51.3.2 static void edu.ou.asgbook.segmentation.HysteresisSegmenter.main (String[] *args*) throws Exception [static]

```

52                                     {

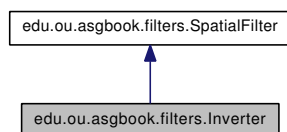
```

```
53         File out = OutputDirectory.getDefault("hysteresis");
54
55         // data
56         LatLonGrid grid = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPop
57         KmlWriter.write(grid, out, "orig", PngWriter.createCoolToWarmColormap());
58
59         // hysteresis thresh
60         for (int thresh = 10; thresh <= 30; thresh += 10){
61             int t1 = thresh;
62             int t2 = thresh-5;
63             Segmenter seg = new HysteresisSegmenter(t1, t2);
64             LatLonGrid label = seg.label(grid).label;
65             // label.setMissing(-1); // so background is present
66             KmlWriter.write(label, out, "cities_"+t1+"_"+t2, PngWriter.createRandomColormap
67         }
68     }
```

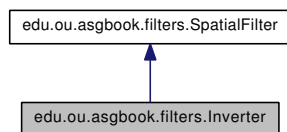
6.52 edu.ou.asgbook.filters.Inverter Class Reference

at every pixel, replaces its value (val) by (A - val)

Inheritance diagram for edu.ou.asgbook.filters.Inverter:



Collaboration diagram for edu.ou.asgbook.filters.Inverter:



Public Member Functions

- [Inverter](#) (int A)
- Override [LatLonGrid filter](#) (LatLonGrid input)
- [LatLonGrid invert](#) (final LatLonGrid input)

6.52.1 Detailed Description

at every pixel, replaces its value (val) by (A - val)

Author:

Valliappa.Lakshmanan

6.52.2 Constructor & Destructor Documentation

6.52.2.1 edu.ou.asgbook.filters.Inverter.Inverter (int A)

```

16         {
17             this.A = A;
18         }
  
```

6.52.3 Member Function Documentation

6.52.3.1 Override [LatLonGrid](#) edu.ou.asgbook.filters.Inverter.filter ([LatLonGrid](#) *input*)

```
21                                     {
22         return invert(input);
23     }
```

6.52.3.2 [LatLonGrid](#) edu.ou.asgbook.filters.Inverter.invert (final [LatLonGrid](#) *input*)

```
25                                     {
26         LatLonGrid output = LatLonGrid.copyOf(input);
27         int[][] outData = output.getData();
28         int[][] inData = input.getData();
29         for (int i=0; i < output.getNumLat(); ++i){
30             for (int j=0; j < output.getNumLon(); ++j){
31                 if ( inData[i][j] != input.getMissing() ){
32                     outData[i][j] = A - inData[i][j];
33                 }
34             }
35         }
36         return output;
37     }
```

6.53 edu.ou.asgbook.motion.KalmanFilter Class Reference

For the time smoothing of motion vectors.

Public Member Functions

- [KalmanFilter](#) (double x_0 , double dx_0)
Start off with an initial estimate for the position and velocity.
- void [init](#) (double x_0 , double dx_0)
- boolean [updated](#) ()
- void [update](#) (double z_k)
- double [getValue](#) ()
get the smoothed centroid position
- double [getRateOfChange](#) ()

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.53.1 Detailed Description

For the time smoothing of motion vectors.

Author:

valliappa.lakshmanan

6.53.2 Constructor & Destructor Documentation

6.53.2.1 edu.ou.asgbook.motion.KalmanFilter.KalmanFilter (double x_0 , double dx_0)

Start off with an initial estimate for the position and velocity.

```
33                                     {
34         init( $x_0$ ,  $dx_0$ );
35     }
```

6.53.3 Member Function Documentation

6.53.3.1 `double edu.ou.asgbook.motion.KalmanFilter.getRateOfChange ()`

```

105                                     {
106         return x_k.get(1,0);
107     }

```

6.53.3.2 `double edu.ou.asgbook.motion.KalmanFilter.getValue ()`

get the smoothed centroid position

```

102                                     {
103         return x_k.get(0, 0);
104     }

```

6.53.3.3 `void edu.ou.asgbook.motion.KalmanFilter.init (double x_0, double dx_0)`

```

37                                     {
38         k = 0;
39         // x_k
40         x_k = new Matrix(2,1);
41         x_k.set(0,0, x_0);
42         x_k.set(1,0, dx_0);
43
44         // p_k
45         p_k = new Matrix( 2, 2 ); // all zero
46
47         // assume unit white noise for errors before we see any observations.
48         R_k = 1;
49         Q_k = Matrix.identity(2,2);
50     }

```

6.53.3.4 `static void edu.ou.asgbook.motion.KalmanFilter.main (String[] args)` `throws Exception` [static]

```

132                                     {
133         double[] truex = new double[20];
134         double[] trueu = new double[truex.length];
135         double[] obsx = new double[truex.length];
136         truex[0] = 5;
137         trueu[0] = 3;
138         obsx[0] = truex[0] + noise();
139
140         KalmanFilter kalman = new KalmanFilter(obsx[0], trueu[0]); // assume that we have a
141         double true_acc = 0.2;

```

```

142
143         System.out.println("true x & true velocity & observed x & estimate of x & estimate of velocity
144         for (int i=1; i < truex.length; ++i){
145             trueu[i] = trueu[i-1] + true_acc;
146             truex[i] = truex[i-1] + trueu[i-1];
147             obsx[i] = noise() + truex[i];
148             kalman.update(obsx[i]);
149             System.out.println( df(truex[i]) + " & " + df(trueu[i]) + " & " + df(obsx[i]) + " & " + df
150         }
151
152
153
154     }

```

6.53.3.5 void edu.ou.asgbook.motion.KalmanFilter.update (double z_k)

```

56         {
57             ++k; // observation number ...
58             if ( MAX_HISTORY > 0 && k > MAX_HISTORY ){
59                 k = MAX_HISTORY; // k is used in computing Q_k and R_k
60             }
61
62             // P_k+1 and x_k+1 will be computed on next turn around so that getValue()
63             // works correctly ...
64             p_k = phi.copy().times(p_k).times(phiT).plus(Q_k);
65             x_k = phi.copy().times(x_k);
66
67
68             // Kalman gain
69             double inv = H.copy().times(p_k).times(HT).get(0,0) + R_k;
70             final Matrix K_k = p_k.copy().times(HT).times( 1.0 / inv );
71
72             // observation error
73             final double v_k = z_k - H.copy().times(x_k).get(0,0);
74
75             // update x_k
76             final Matrix update = K_k.copy().times(v_k);
77             x_k = x_k.plus( update );
78
79             // estimate R_k, covariance of observation error to use next time 'round
80             R_k = ( (k-1) * R_k + v_k * v_k ) / k;
81
82             // estimate Q_k, covariance of model error to use in P_k+1 computation
83             if ( k != 1 ){ // when k is 1, x_k=old_x_k and so Q_k would become 0
84                 final Matrix wkT = update.copy().transpose();
85                 final Matrix wk_wkT = update.copy().times(wkT);
86                 Q_k = Q_k.times(k-1).plus(wk_wkT).times(1.0/k);
87             }
88
89             // update error covariance for updated estimate
90             p_k = Matrix.identity(2,2).minus(K_k.copy().times(H)).times(p_k);
91
92             if ( finite(getValue()) == false || finite(getRateOfChange()) == false ){
93                 double new_val = getValue();
94                 if ( finite(new_val) == false ) new_val = 0;

```

```
95         double new_rate = getRateOfChange();
96         if ( finite(new_rate) == false ) new_rate = 0;
97         init( new_val, new_rate );
98     }
99 }
```

6.53.3.6 boolean edu.ou.asgbook.motion.KalmanFilter.updated ()

```
52         {
53         return ( k > 0 );
54     }
```


6.54 edu.ou.asgbook.io.KmlWriter Class Reference

Writes data out in KML form, for display in Google Earth or similar program.

Static Public Member Functions

- static void [write](#) ([LatLonGrid](#) grid, File outputDir, String dataName, ColorModel colormap) throws Exception
- static void [write](#) (List< [LatLon](#) > points, File outputDir, String dataName) throws Exception
- static void [write](#) (List< [LatLon](#) > points, List< String > names, File outputDir, String dataName) throws Exception
- static void [debugWrite](#) ([LatLonGrid](#) grid, File out, String name)
- static void [main](#) (String[] args) throws Exception

6.54.1 Detailed Description

Writes data out in KML form, for display in Google Earth or similar program.

Author:

Valliappa.Lakshmanan

6.54.2 Member Function Documentation

6.54.2.1 static void edu.ou.asgbook.io.KmlWriter.debugWrite ([LatLonGrid](#) *grid*, File *out*, String *name*) [static]

```

121                                     {
122         if (out != null){
123             try {
124                 KmlWriter.write(grid, out, name, PngWriter.createCoolToWarmColormap());
125             } catch (Exception e) {
126                 e.printStackTrace();
127             }
128         }
129     }
```

6.54.2.2 static void edu.ou.asgbook.io.KmlWriter.main (String[] *args*) throws Exception [static]

```

132                                     {
133         LatLonGrid grid = new LatLonGrid(100, 200, -1, new LatLon(35,-97), 0.1, 0.1);
134         for (int i=0; i < grid.getNumLat(); ++i){
```

```

135         for (int j=0; j < grid.getNumLon(); ++j){
136             grid.getData()[i][j] = i + j;
137             if ( i%10 == 0 || j%20 == 0){
138                 grid.getData()[i][j] = grid.getMissing();
139             }
140         }
141     }
142     File outputDir = OutputDirectory.getDefault("kmlwriter");
143     KmlWriter.write(grid, outputDir, "kmlwriter", PngWriter.createHotColormap());
144 }

```

6.54.2.3 `static void edu.ou.asgbook.io.KmlWriter.write (List< LatLon > points, List< String > names, File outputDir, String dataName) throws Exception` [static]

```

88
89     // create KML doc
90     Document doc = DocumentBuilderFactory.newInstance().newDocumentBuilder().newDocument();
91     Element root = doc.createElement("kml");
92     doc.appendChild(root);
93     Element docE = doc.createElement("Document");
94     root.appendChild(docE);
95     for (int i=0; i < points.size(); ++i){
96         Element placemark = doc.createElement("Placemark");
97         docE.appendChild(placemark);
98         Element name = doc.createElement("name");
99         placemark.appendChild(name);
100         if (names != null && i < names.size()){
101             name.setTextContent(names.get(i));
102         } else {
103             name.setTextContent(dataName + "#" + (i+1));
104         }
105         Element point = doc.createElement("Point");
106         placemark.appendChild(point);
107         Element coords = doc.createElement("coordinates");
108         point.appendChild(coords);
109         coords.setTextContent(points.get(i).getLon() + "," + points.get(i).getLat() + " ");
110     }
111
112     // write out
113     File filename = new File(outputDir.getAbsolutePath() + "/" + dataName + ".kml");
114     Transformer t = TransformerFactory.newInstance().newTransformer();
115     t.setOutputProperty(OutputKeys.INDENT, "yes");
116     t.setOutputProperty("{http://xml.apache.org/xslt}indent-amount", "2");
117     t.transform(new DOMSource(doc), new StreamResult(filename));
118     System.out.println("Wrote " + filename + " to refer to " + names.size() + " placemarks");
119 }

```

6.54.2.4 `static void edu.ou.asgbook.io.KmlWriter.write (List< LatLon > points, File outputDir, String dataName) throws Exception` [static]

```

80

```

```

81         List<String> names = new ArrayList<String>();
82         for (int i=0; i < points.size(); ++i){
83             names.add(dataName + " " + (i+1) );
84         }
85         write(points, names, outputDir, dataName);
86     }

```

6.54.2.5 static void edu.ou.asgbook.io.KmlWriter.write ([LatLonGrid](#) grid, File outputDir, String dataName, ColorModel colormap) throws Exception [static]

```

32
33         // write image
34         File imgFileName = new File(outputDir.getAbsolutePath() + "/" + dataName + ".png");
35         PngWriter.writeAutoScaled(grid, imgFileName, colormap);
36
37         // create KML
38         Document doc = DocumentBuilderFactory.newInstance().newDocumentBuilder().newDocument();
39         Element root = doc.createElement("kml");
40         doc.appendChild(root);
41         Element folder = doc.createElement("Folder");
42         root.appendChild(folder);
43         Element folderName = doc.createElement("name");
44         folder.appendChild(folderName);
45         folderName.setTextContent(dataName);
46         Element folderDesc = doc.createElement("description");
47         folderDesc.setTextContent(dataName + " created by " + KmlWriter.class.getCanonicalName() + " fo
48         Element goverlay = doc.createElement("GroundOverlay");
49         folder.appendChild(goverlay);
50         Element icon = doc.createElement("Icon");
51         goverlay.appendChild(icon);
52         Element href = doc.createElement("href");
53         icon.appendChild(href);
54         href.setTextContent(dataName + ".png");
55         Element box = doc.createElement("LatLonBox");
56         goverlay.appendChild(box);
57         Element north = doc.createElement("north");
58         north.setTextContent("" + grid.getNwCorner().getLat());
59         box.appendChild(north);
60         Element south = doc.createElement("south");
61         south.setTextContent("" + grid.getSeCorner().getLat());
62         box.appendChild(south);
63         Element east = doc.createElement("east");
64         east.setTextContent("" + grid.getSeCorner().getLon());
65         box.appendChild(east);
66         Element west = doc.createElement("west");
67         west.setTextContent("" + grid.getNwCorner().getLon());
68         box.appendChild(west);
69         box.appendChild(north);
70
71         // write KML
72         File kmlFileName = new File(outputDir.getAbsolutePath() + "/" + dataName + ".kml");
73         Transformer t = TransformerFactory.newInstance().newTransformer();
74         t.setOutputProperty(OutputKeys.INDENT, "yes");

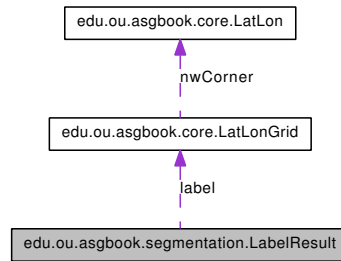
```

```
75         t.setOutputProperty("{http://xml.apache.org/xslt}indent-amount", "2");
76         t.transform(new DOMSource(doc), new StreamResult(kmlFileName));
77         System.out.println("Wrote " + kmlFileName + " to refer to image");
78     }
```

6.55 edu.ou.asgbook.segmentation.LabelResult Class Reference

Result of segmentation.

Collaboration diagram for edu.ou.asgbook.segmentation.LabelResult:



Public Member Functions

- [LabelResult](#) ([LatLonGrid](#) label, int maxlabel)

Public Attributes

- final [LatLonGrid](#) label
- final int maxlabel

6.55.1 Detailed Description

Result of segmentation.

Each pixel holds the region number that it belongs to. Zero is the background value.

Author:

valliappa.lakshmanan

6.55.2 Constructor & Destructor Documentation

6.55.2.1 edu.ou.asgbook.segmentation.LabelResult.LabelResult ([LatLonGrid](#) label, int maxlabel)

```

15                                     {
16         this->label = label;

```

```
17         this.maxlabel = maxlabel;
18     }
```

6.55.3 Member Data Documentation

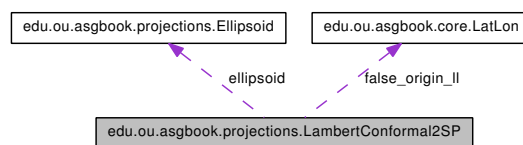
6.55.3.1 final [LatLonGrid](#) [edu.ou.asgbook.segmentation.LabelResult.label](#)

6.55.3.2 final int [edu.ou.asgbook.segmentation.LabelResult.maxlabel](#)

6.56 edu.ou.asgbook.projections.LambertConformal2SP Class Reference

Lambert Conformation 2 Standard Parallels map projection.

Collaboration diagram for edu.ou.asgbook.projections.LambertConformal2SP:



Public Member Functions

- [LambertConformal2SP](#) ([Ellipsoid](#) ellipsoid, [LatLon](#) falseOriginLl, double lat_1, double lat_2, Coord falseOriginLam)
- Coord [getLambert](#) ([LatLon](#) in)
- [LatLon](#) [getLatLon](#) (Coord lam)

Static Public Member Functions

- static void [main](#) (String[] args)

Classes

- class [Coord](#)

6.56.1 Detailed Description

Lambert Conformation 2 Standard Parallels map projection.

Author:

v.lakshmanan

6.56.2 Constructor & Destructor Documentation

6.56.2.1 `edu.ou.asgbook.projections.LambertConformal2SP.LambertConformal2SP` ([Ellipsoid](#) *ellipsoid*, [LatLon](#) *falseOriginLl*, double *lat_1*, double *lat_2*, [Coord](#) *falseOriginLam*)

```

35                                     {
36         this.ellipsoid = ellipsoid;
37         this.false_origin_ll = falseOriginLl;
38         this.lat_1 = lat_1;
39         this.lat_2 = lat_2;
40         this.false_origin_lam = falseOriginLam;
41
42         this.e = Math.sqrt(ellipsoid.eccsq);
43         double phi1 = Math.toRadians(this.lat_1);
44         double phi2 = Math.toRadians(this.lat_2);
45         double t1 = compute_t(e, phi1);
46         double t2 = compute_t(e, phi2);
47         double m1 = compute_m(e, phi1);
48         double m2 = compute_m(e, phi2);
49
50         this.n = (Math.log(m1) - Math.log(m2)) / (Math.log(t1) - Math.log(t2));
51         this.F = m1 / (n * Math.pow(t1, n));
52
53         double phiF = Math.toRadians(false_origin_ll.getLat());
54         double tF = compute_t(e, phiF);
55         this.rF = ellipsoid.eqr * F * Math.pow(tF, n);
56     }
```

6.56.3 Member Function Documentation

6.56.3.1 `Coord` `edu.ou.asgbook.projections.LambertConformal2SP.getLambert` ([LatLon](#) *in*)

```

58                                     {
59         double phi = Math.toRadians(in.getLat());
60         double t = compute_t(e, phi);
61         double r = ellipsoid.eqr * F * Math.pow(t, n);
62         double lambda = Math.toRadians(in.getLon());
63         double lambdaF = Math.toRadians(false_origin_ll.getLon());
64         double theta = n * (lambda - lambdaF);
65
66         double easting = false_origin_lam.easting + r * Math.sin(theta);
67         double northing = false_origin_lam.northing + rF - r * Math.cos(theta);
68         return new Coord(northing, easting);
69     }
```

6.56.3.2 [LatLon](#) `edu.ou.asgbook.projections.LambertConformal2SP.getLatLon` (`Coord` *lam*)

```

71                                     {
```



```

72     double eastdiff = (lam.easting - false_origin_lam.easting);
73     double northdiff = (lam.northing - false_origin_lam.northing);
74     double rFnorthdiff = rF - northdiff;
75     double r = Math.sqrt( eastdiff*eastdiff + rFnorthdiff*rFnorthdiff );
76     if ( n < 0 ) r = -r;
77     double t = Math.pow( r / (ellipsoid.eqr*F) , 1/n );
78     double theta = Math.atan( eastdiff/rFnorthdiff );
79
80     double lon = Math.toDegrees(theta/n) + false_origin_ll.getLon();
81
82     // iterate to find phi
83     double phi = Math.PI/2 - 2 * Math.atan(t);
84     double old_phi;
85     int iter=0;
86     do{
87         old_phi = phi;
88         ++iter;
89         phi = Math.PI/2 - 2 * Math.atan( t*Math.pow( (1-e*Math.sin(phi))/(1+e*Math.sin(phi)), e/2 ) );
90     } while ( Math.abs(phi-old_phi) > 0.00001 && iter < 5 );
91
92     double lat = Math.toDegrees(phi);
93     return new LatLon( lat, lon );
94 }

```

6.56.3.3 static void edu.ou.asgbook.projections.LambertConformal2SP.main (String[] args) [static]

```

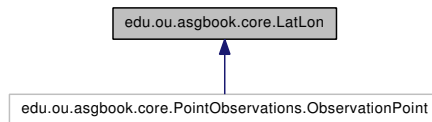
107     {
108         LambertConformal2SP conv = new LambertConformal2SP(Ellipsoid.WGS84(), new LatLon(51,-127), 43.
109         LatLon ll = new LatLon(36,-96);
110         LambertConformal2SP.Coord lam = conv.getLambert(ll);
111         LatLon ll2 = conv.getLatLon(lam);
112         System.out.println(ll + "->" + lam + "->" + ll2);
113
114         ll = new LatLon(51,-96);
115         lam = conv.getLambert(ll);
116         ll2 = conv.getLatLon(lam);
117         System.out.println(ll + "->" + lam + "->" + ll2);
118
119         ll = new LatLon(21,-96);
120         lam = conv.getLambert(ll);
121         ll2 = conv.getLatLon(lam);
122         System.out.println(ll + "->" + lam + "->" + ll2);
123
124         ll = new LatLon(36,-127);
125         lam = conv.getLambert(ll);
126         ll2 = conv.getLatLon(lam);
127         System.out.println(ll + "->" + lam + "->" + ll2);
128
129         ll = new LatLon(36,-65);
130         lam = conv.getLambert(ll);
131         ll2 = conv.getLatLon(lam);
132         System.out.println(ll + "->" + lam + "->" + ll2);
133     }

```

6.57 edu.ou.asgbook.core.LatLon Class Reference

A point on the earth's surface typically in WGS84.

Inheritance diagram for edu.ou.asgbook.core.LatLon:



Public Member Functions

- double [getLat](#) ()
- double [getLon](#) ()
- [LatLon](#) (double lat, double lon)
- double [distanceInKms](#) ([LatLon](#) other)
- Override String [toString](#) ()

Static Public Member Functions

- static void [main](#) (String[] args)

6.57.1 Detailed Description

A point on the earth's surface typically in WGS84.

Author:

Valliappa.Lakshmanan

6.57.2 Constructor & Destructor Documentation

6.57.2.1 edu.ou.asgbook.core.LatLon.LatLon (double *lat*, double *lon*)

```

22                                     {
23         super();
24         this.lat = lat;
25         this.lon = lon;
26     }
```

6.57.3 Member Function Documentation

6.57.3.1 double edu.ou.asgbook.core.LatLon.distanceInKms ([LatLon](#) *other*)

```

32         {
33             double lat1 = Math.toRadians(this.lat);
34             double lat2 = Math.toRadians(other.lat);
35             double lon1 = Math.toRadians(this.lon);
36             double lon2 = Math.toRadians(other.lon);
37
38             // double R = 6371; // spherical earth radius
39             double lat0 = (lat2+lat1)/2;
40             double a = 6378.137; // WGS-84
41             double f = 1.0/298.257223563;
42             double esq = f*(2-f);
43             double R=a * (1-esq)/Math.pow(sq(1-esq*(Math.sin(lat0))),1.5);
44
45             double dlon = lon2 - lon1;
46             double dlat = lat2 - lat1;
47             double term = sq(Math.sin(dlat/2)) + Math.cos(lat1) * Math.cos(lat2) * sq(Math.sin(dlon/2));
48             return (2 * R * Math.asin(Math.min(1,Math.sqrt(term))));
49         }

```

6.57.3.2 double edu.ou.asgbook.core.LatLon.getLat ()

```

16         {
17             return lat;
18         }

```

6.57.3.3 double edu.ou.asgbook.core.LatLon.getLon ()

```

19         {
20             return lon;
21         }

```

6.57.3.4 static void edu.ou.asgbook.core.LatLon.main (String[] *args*) [static]

```

57         {
58             LatLon pt1 = new LatLon(35,-97);
59             LatLon pt2 = new LatLon(35.01, -97);
60             LatLon pt3 = new LatLon(35, -97.01);
61             System.out.println("sph: 0.01 in lat = " + pt1.distanceInKms(pt2) + " kms at " + pt1);
62             System.out.println("sph: 0.01 in lon = " + pt1.distanceInKms(pt3) + " kms at " + pt1);
63         }

```

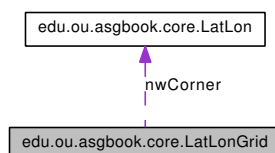
6.57.3.5 Override String edu.ou.asgbook.core.LatLon.toString ()

```
52         {  
53         return new StringBuilder().append("[").append(lat).append(", ")  
54             .append(lon).append("]").toString();  
55     }
```

6.58 edu.ou.asgbook.core.LatLonGrid Class Reference

A geospatial grid of data in equilateral coordinates typically in WGS84 ellipsoid.

Collaboration diagram for edu.ou.asgbook.core.LatLonGrid:



Public Member Functions

- [LatLonGrid](#) (int[][] data, int missing, [LatLon](#) nwCorner, double latres, double lonres)
- [LatLonGrid crop](#) (int startRow, int startCol, int numLat, int numLon)

Crop this grid.

- [LatLonGrid](#) (int nrows, int ncols, int missing, [LatLon](#) nwCorner, double latres, double lonres)

Initialize a grid of data at zero.

- int[][] [getData](#) ()
- int [getMissing](#) ()
- [LatLon](#) [getNwCorner](#) ()

Note that this is the true corner, not the center of the first grid point.

- double [getLatRes](#) ()
- double [getLonRes](#) ()
- [LatLon](#) [getLocation](#) (int row, int col)
- [LatLon](#) [getLocation](#) ([Pixel](#) p)
- [LatLon](#) [getLocation](#) (double row, double col)
- [LatLon](#) [getSeCorner](#) ()

This is the true corner, not the middle of the last grid point.

- int [getNumLon](#) ()
- int [getNumLat](#) ()
- int [getValue](#) (int row, int col)
- void [setValue](#) (int row, int col, int value)
- void [setMissing](#) (int i)
- final int [getRow](#) ([LatLon](#) location)

The returned row may be outside this grid's dimensions.

- final [Pixel](#) [getPixel](#) ([LatLon](#) location)
- int [getCol](#) ([LatLon](#) location)

The returned col may be outside this grid's dimensions.

- final int [getValue](#) ([LatLon](#) location)
- final boolean [isValid](#) (int row, int col)

Are the pixel coordinates in bounds?

- void [fill](#) (int newval)
- void [replace](#) (int oldval, int newval)
- int [getValue](#) ([Pixel](#) pixel)
- [Pixel](#)[] [asPixels](#) ()
- int[][] [longitudewrap](#) (int Ny)
- [LatLonGrid](#) [remapTo](#) ([LatLonGrid](#) other)

Static Public Member Functions

- static [LatLonGrid](#) [copyOf](#) (final [LatLonGrid](#) original)
Make a deep copy.
- static [LatLonGrid](#) [add](#) ([LatLonGrid](#) a, [LatLonGrid](#) b)

6.58.1 Detailed Description

A geospatial grid of data in equilat equilon coordinates typically in WGS84 ellipsoid.

Author:

Valliappa.Lakshmanan

6.58.2 Constructor & Destructor Documentation

6.58.2.1 [edu.ou.asgbook.core.LatLonGrid.LatLonGrid](#) (int *data*[], int *missing*, [LatLon](#) *nwCorner*, double *latres*, double *lonres*)

Parameters:

data Holds on to provided data (does not clone the data)

missing Missing data value, typically -9999 or similar

nwCorner the true corner, not the center of the first grid point

latres A positive number

lonres A positive number

```

32                                     {
33         super();
34         this.data = data;
35         this.missing = missing;
36         this.nwCorner = nwCorner;
37         this.latRes = latres;
38         this.lonRes = lonres;
39     }
```

6.58.2.2 edu.ou.asgbook.core.LatLonGrid.LatLonGrid (int *nrows*, int *ncols*, int *missing*, [LatLon](#) *nwCorner*, double *latres*, double *lonres*)

Initialize a grid of data at zero.

Parameters:

nrows

ncols

missing Missing data value, typically -9999 or similar

nwCorner the true corner, not the center of the first grid point

latres A positive number

lonres A positive number

```

80                                     {
81         this( new int[nrows][ncols], missing, nwCorner, latres, lonres );
82     }
```

6.58.3 Member Function Documentation

6.58.3.1 static [LatLonGrid](#) edu.ou.asgbook.core.LatLonGrid.add ([LatLonGrid](#) *a*, [LatLonGrid](#) *b*) [static]

```

213                                     {
214         int nrows = a.getNumLat();
215         int ncols = a.getNumLon();
216         if (b.getNumLat() != nrows || b.getNumLon() != ncols){
217             throw new IllegalArgumentException("Grids are of different dimensions: first grid is " + r
218         }
219         LatLonGrid result = LatLonGrid.copyOf(a);
220         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
221             if (result.data[i][j] != result.missing){
222                 int bval = b.data[i][j];
223                 if (bval != b.missing){
```

```

224             result.data[i][j] += bval;
225         } else {
226             result.data[i][j] = result.missing;
227         }
228     }
229 }
230 return result;
231 }

```

6.58.3.2 [Pixel \[\]](#) edu.ou.asgbook.core.LatLonGrid.asPixels ()

```

237         {
238         List<Pixel> pixels = new ArrayList<Pixel>();
239         for (int i=0; i < data.length; ++i) for (int j=0; j < data[i].length; ++j){
240             if (data[i][j] != missing){
241                 pixels.add(new Pixel(i, j, data[i][j]));
242             }
243         }
244         return pixels.toArray(new Pixel[0]);
245     }

```

6.58.3.3 static [LatLonGrid](#) edu.ou.asgbook.core.LatLonGrid.copyOf (final [LatLonGrid original](#)) [static]

Make a deep copy.

```

44         {
45         int[][] copy = new int[original.getNumLat()][original.getNumLon()];
46         for (int i=0; i < original.getNumLat(); ++i){
47             for (int j=0; j < original.getNumLon(); ++j){
48                 copy[i][j] = original.data[i][j];
49             }
50         }
51         return new LatLonGrid(copy, original.missing, original.nwCorner, original.latRes, original.lonRes);
52     }

```

6.58.3.4 [LatLonGrid](#) edu.ou.asgbook.core.LatLonGrid.crop (int *startRow*, int *startCol*, int *numLat*, int *numLon*)

Crop this grid.

Does not check dimensions

```

57         {
58         int[][] copy = new int[numLat][numLon];
59         for (int i=0; i < numLat; ++i){
60             for (int j=0; j < numLon; ++j){

```



```

61         copy[i][j] = data[i+startRow][j+startCol];
62     }
63 }
64 LatLon origin = this.getLocation(startRow, startCol);
65 LatLon nwCorner = new LatLon( origin.getLat() + latRes/2 , origin.getLon() - lonRes/2 );
66 return new LatLonGrid(copy, missing, nwCorner, latRes, lonRes);
67 }

```

6.58.3.5 void edu.ou.asgbook.core.LatLonGrid.fill (int newval)

```

191     {
192         final int nrows = data.length;
193         final int ncols = data[0].length;
194         for (int i=0; i < nrows; ++i){
195             for (int j=0; j < ncols; ++j){
196                 data[i][j] = newval;
197             }
198         }
199     }

```

6.58.3.6 int edu.ou.asgbook.core.LatLonGrid.getCol ([LatLon](#) location)

The returned col may be outside this grid's dimensions.

```

170     {
171         int col = (int) ( (location.getLon() - nwCorner.getLon())/lonRes );
172         return col;
173     }

```

6.58.3.7 int [][] edu.ou.asgbook.core.LatLonGrid.getData ()

```

84     {
85         return data;
86     }

```

6.58.3.8 double edu.ou.asgbook.core.LatLonGrid.getLatRes ()

```

100     {
101         return latRes;
102     }

```

6.58.3.9 [LatLon](#) edu.ou.asgbook.core.LatLonGrid.getLocation (double *row*, double *col*)

```
118                                     {
119     // latitude decreases, longitude increases
120     return new LatLon( nwCorner.getLat() - (row+0.5)*latRes,
121                       nwCorner.getLon() + (col+0.5)*lonRes );
122 }
```

6.58.3.10 [LatLon](#) edu.ou.asgbook.core.LatLonGrid.getLocation ([Pixel](#) *p*)

```
114                                     {
115     return getLocation(p.getRow(), p.getCol());
116 }
```

6.58.3.11 [LatLon](#) edu.ou.asgbook.core.LatLonGrid.getLocation (int *row*, int *col*)

```
108                                     {
109     // latitude decreases, longitude increases
110     return new LatLon( nwCorner.getLat() - (row+0.5)*latRes,
111                       nwCorner.getLon() + (col+0.5)*lonRes );
112 }
```

6.58.3.12 double edu.ou.asgbook.core.LatLonGrid.getLonRes ()

```
104                                     {
105     return lonRes;
106 }
```

6.58.3.13 int edu.ou.asgbook.core.LatLonGrid.getMissing ()

```
88                                     {
89     return missing;
90 }
```

6.58.3.14 int edu.ou.asgbook.core.LatLonGrid.getNumLat ()

```
137                                     {
138     return data.length;
139 }
```

6.58.3.15 `int edu.ou.asgbook.core.LatLonGrid.getNumLon ()`

```
133         {
134     return data[0].length;
135 }
```

6.58.3.16 `LatLon edu.ou.asgbook.core.LatLonGrid.getNwCorner ()`

Note that this is the true corner, not the center of the first grid point.

Returns:

```
96         {
97     return nwCorner;
98 }
```

6.58.3.17 `final Pixel edu.ou.asgbook.core.LatLonGrid.getPixel (LatLon location)`

```
161         {
162     int row = getRow(location);
163     int col = getCol(location);
164     return new Pixel(row, col, data[row][col]);
165 }
```

6.58.3.18 `final int edu.ou.asgbook.core.LatLonGrid.getRow (LatLon location)`

The returned row may be outside this grid's dimensions.

```
156         {
157     int row = (int) ( (nwCorner.getLat() - location.getLat())/latRes );
158     return row;
159 }
```

6.58.3.19 `LatLon edu.ou.asgbook.core.LatLonGrid.getSeCorner ()`

This is the true corner, not the middle of the last grid point.

```
127         {
128     // latitude decreases, longitude increases
129     return new LatLon( nwCorner.getLat() - getNumLat()*latRes,
130         nwCorner.getLon() + getNumLon()*lonRes );
131 }
```

6.58.3.20 `int edu.ou.asgbook.core.LatLonGrid.getValue (Pixel pixel)`

```

233         {
234         return getValue(pixel.getX(), pixel.getY());
235     }

```

6.58.3.21 `final int edu.ou.asgbook.core.LatLonGrid.getValue (LatLon location)`

```

175         {
176         int row = getRow(location);
177         int col = getCol(location);
178         if ( isValid(row, col) ){
179             return data[row][col];
180         }
181         return missing;
182     }

```

6.58.3.22 `int edu.ou.asgbook.core.LatLonGrid.getValue (int row, int col)`

```

141         {
142         return data[row][col];
143     }

```

6.58.3.23 `final boolean edu.ou.asgbook.core.LatLonGrid.isValid (int row, int col)`

Are the pixel coordinates in bounds?

```

187         {
188         return row >= 0 && row < data.length && col >= 0 && col < data[row].length;
189     }

```

6.58.3.24 `int [][] edu.ou.asgbook.core.LatLonGrid.longitudewrap (int Ny)`

```

247         {
248         int nrows = data.length;
249         int ncols = data[0].length;
250         int hy = Ny/2;
251         int outcols = ncols + 2*hy;
252         int [ ][ ] result = new int[nrows][outcols];
253         for (int i=0; i < nrows; ++i) for (int j=0; j < outcols; ++j){
254             int incol = j - hy;
255             if (incol < 0) incol += ncols; // wrap
256             else if (incol >= ncols) incol -= ncols;
257             result[i][j] = data[i][incol];

```

```

258         }
259         return result;
260     }

```

6.58.3.25 **LatLonGrid** edu.ou.asgbook.core.LatLonGrid.remapTo (**LatLonGrid** *other*)

```

262         {
263         LatLonGrid result = LatLonGrid.copyOf(other);
264         result.setMissing(this.getMissing());
265         for (int i=0; i < other.getNumLat(); ++i){
266             int row = getRow( other.getLocation(i,0) );
267             for (int j=0; j < other.getNumLon(); ++j){
268                 int col = getCol( other.getLocation(i,j) );
269                 if (this.isValid(row,col)){
270                     result.setValue(i,j, data[row][col]);
271                 } else {
272                     result.setValue(i,j, result.missing());
273                 }
274             }
275         }
276         return result;
277     }

```

6.58.3.26 **void** edu.ou.asgbook.core.LatLonGrid.replace (int *oldval*, int *newval*)

```

201         {
202         final int nrows = data.length;
203         final int ncols = data[0].length;
204         for (int i=0; i < nrows; ++i){
205             for (int j=0; j < ncols; ++j){
206                 if (data[i][j] == oldval){
207                     data[i][j] = newval;
208                 }
209             }
210         }
211     }

```

6.58.3.27 **void** edu.ou.asgbook.core.LatLonGrid.setMissing (int *i*)

```

149         {
150         missing = i;
151     }

```

6.58.3.28 **void** edu.ou.asgbook.core.LatLonGrid.setValue (int *row*, int *col*, int *value*)

```

145         {

```

```
146         data[row][col] = value;
147     }
```

6.59 edu.ou.asgbook.core.LevelSet Class Reference

A representation of a spatial grid as a set of levels.

6.59.1 Detailed Description

A representation of a spatial grid as a set of levels.

Author:

valliappa.lakshmanan

6.60 edu.ou.asgbook.rasterization.Line Class Reference

A line that connects two points on the earth's surface.

Public Member Functions

- [Line](#) (double lat0, double lon0, double lat1, double lon1)
- [Line](#) ([LatLon](#) p0, [LatLon](#) p1)
- double [getLat0](#) ()
- double [getLon0](#) ()
- double [getLat1](#) ()
- double [getLon1](#) ()
- List< [Pixel](#) > [getPositionIn](#) ([LatLonGrid](#) grid)
- Double [getXIntercept](#) (double y)

Find the intersection point.

- Double [getYIntercept](#) (double x)

Find the intersection point.

Static Public Member Functions

- static void [main](#) (String args[]) throws Exception

6.60.1 Detailed Description

A line that connects two points on the earth's surface.

Author:

valliappa.lakshmanan

6.60.2 Constructor & Destructor Documentation

6.60.2.1 edu.ou.asgbook.rasterization.Line.Line (double lat0, double lon0, double lat1, double lon1)

```

27                                     {
28         this.lat0 = lat0;
29         this.lon0 = lon0;
30         this.lat1 = lat1;

```



```
31         this.lon1 = lon1;
32     }
```

6.60.2.2 edu.ou.asgbook.rasterization.Line.Line (LatLon *p0*, LatLon *p1*)

```
34                                     {
35         this.lat0 = p0.getLat();
36         this.lon0 = p0.getLon();
37         this.lat1 = p1.getLat();
38         this.lon1 = p1.getLon();
39     }
```

6.60.3 Member Function Documentation

6.60.3.1 double edu.ou.asgbook.rasterization.Line.getLat0 ()

```
41                                     {
42         return lat0;
43     }
```

6.60.3.2 double edu.ou.asgbook.rasterization.Line.getLat1 ()

```
49                                     {
50         return lat1;
51     }
```

6.60.3.3 double edu.ou.asgbook.rasterization.Line.getLon0 ()

```
45                                     {
46         return lon0;
47     }
```

6.60.3.4 double edu.ou.asgbook.rasterization.Line.getLon1 ()

```
53                                     {
54         return lon1;
55     }
```

6.60.3.5 List<Pixel> edu.ou.asgbook.rasterization.Line.getPositionIn (LatLonGrid grid)

```

57                                     {
58     List<Pixel> result = new ArrayList<Pixel>();
59     Pixel p0 = grid.getPixel( new LatLon(lat0, lon0) );
60     Pixel p1 = grid.getPixel( new LatLon(lat1, lon1) );
61     System.out.println("Line from " + p0 + " to " + p1);
62     int rowlen = Math.abs(p0.getRow() - p1.getRow());
63     int collen = Math.abs(p0.getCol() - p1.getCol());
64     // avoid divide by zero in slope calculations below
65     if ( rowlen == 0 && collen == 0 ){
66         result.add(p0);
67         return result;
68     }
69     if ( rowlen > collen ){
70         // increment in row
71         int startrow = Math.min(p0.getRow(), p1.getRow());
72         int endrow = Math.max(p0.getRow(), p1.getRow());
73         double slope = (p1.getCol() - p0.getCol()) / ((double) (p1.getRow() - p0.getRow()));
74         for (int row=startrow; row <= endrow; ++row){
75             int col = (int) Math.round(slope*(row-p0.getRow())+p0.getCol());
76             if (grid.isValid(row, col)){
77                 result.add( new Pixel(row, col, grid.getValue(row, col)) );
78             }
79         }
80     } else {
81         int startcol = Math.min(p0.getCol(), p1.getCol());
82         int endcol = Math.max(p0.getCol(), p1.getCol());
83         double slope = (p1.getRow() - p0.getRow()) / ((double) (p1.getCol() - p0.getCol()));
84         for (int col=startcol; col <= endcol; ++col){
85             int row = (int) Math.round(slope*(col-p0.getCol())+p0.getRow());
86             if (grid.isValid(row, col)){
87                 result.add( new Pixel(row, col, grid.getValue(row, col)) );
88             }
89         }
90     }
91     return result;
92 }

```

6.60.3.6 Double edu.ou.asgbook.rasterization.Line.getXIntercept (double y)

Find the intersection point.

Returns null if not in range.

```

100                                     {
101     if (!isBetween(lon0, y, lon1)) {
102         return null;
103     }
104     // if y0=y1, then inrange would be false
105     double x;
106     if (lon0 != lon1) {
107         x = lat0 + (y - lon0) * (lat1 - lat0) / (lon1 - lon0);

```

```
108         } else {
109             x = (lat1 + lat0) / 2;
110         }
111         return x;
112     }
```

6.60.3.7 Double edu.ou.asgbook.rasterization.Line.getYIntercept (double x)

Find the intersection point.

Returns null if not in range.

```
115         {
116             if (!isBetween(lat0, x, lat1)) {
117                 return null;
118             }
119             double y;
120             if (lat0 != lat1) {
121                 y = lon0 + (x - lat0) * (lon1 - lon0) / (lat1 - lat0);
122             } else {
123                 y = (lon1 + lon0) / 2;
124             }
125             return y;
126         }
```

6.60.3.8 static void edu.ou.asgbook.rasterization.Line.main (String args[]) throws Exception [static]

```
128         {
129             LatLonGrid grid = new LatLonGrid(100,100,0,new LatLon(100,-90),0.01,0.01);
130             List<Pixel> ver = new Line(99.3,-89.3,99.7,-89.4).getPositionIn(grid);
131             List<Pixel> hor = new Line(99.3,-89.3,99.4,-89.7).getPositionIn(grid);
132             for (Pixel p : ver){
133                 grid.setValue(p.getRow(), p.getCol(), 10);
134             }
135             for (Pixel p : hor){
136                 grid.setValue(p.getRow(), p.getCol(), 20);
137             }
138
139             File out = OutputDirectory.getDefault("raster");
140             KmlWriter.write(grid, out, "drawlines", PngWriter.createCoolToWarmColormap());
141         }
```

6.61 edu.ou.asgbook.linearity.LinearityVerifier Class Reference

Given a 2D array of points, reports error measures of assuming linearity.

Static Public Member Functions

- static [ScalarStatistic verify](#) (int[][] data, DataSelector selector, [DataTransform](#) transform, int neighSize)
Returns the Mean Square Error statistic.
- static void [main](#) (String[] args) throws Exception

Classes

- interface **DataSelector**
- class **InRange**
- class **NotMissing**

6.61.1 Detailed Description

Given a 2D array of points, reports error measures of assuming linearity.

By passing in different transformations, it is possible to compare potential ways of transforming the data.

Author:

valliappa.lakshmanan

6.61.2 Member Function Documentation

6.61.2.1 static void edu.ou.asgbook.linearity.LinearityVerifier.main (String[] args) throws Exception [static]

```

95                                     {
96         // read input
97         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new Lin
98         for (int i=0; i < popdensity.getNumLat(); ++i){
99             for (int j=0; j < popdensity.getNumLon(); ++j){
100                 if (popdensity.getValue(i, j) < 1){
101                     popdensity.setValue(i, j, popdensity.getMissing());
102                 }
103             }

```

```

104         }
105
106         // DataSelector selector = new NotMissing(popdensity.getMissing());
107         DecimalFormat df = new DecimalFormat("0.0");
108         // int maxval = new NHighest(1).findHighestValued(popdensity)[0].getValue();
109         int[] neighSize = new int[]{ 1, 3, 5, 11, 21, 31, 41 };
110         int[] thresh1 = new int[]{ 1, 1, 50, 500, 5000, 50000 };
111         int[] thresh2 = new int[]{ 500000, 50, 500, 5000, 50000, 500000 };
112         final String sep = " & ";
113         System.out.println("D & data range & N & RMSE (raw) & RMSE (log) \\\n");
114         for (int D : neighSize){
115             for (int i=0; i < thresh1.length; ++i){
116                 int minval = thresh1[i];
117                 int maxval = thresh2[i];
118                 DataSelector selector = new InRange(minval, maxval, popdensity.getMissing());
119                 // check linearity two ways
120                 ScalarStatistic logstat = verify(popdensity.getData(), selector, new LogScaling(10), D);
121                 ScalarStatistic rawstat = verify(popdensity.getData(), selector, new LinearScaling(100), D);
122                 System.out.println(D + sep +
123                     minval + "-" + maxval + sep +
124                     rawstat.getNumSamples() + sep +
125                     df.format(Math.sqrt(rawstat.getMean())) + sep +
126                     df.format(Math.sqrt(logstat.getMean())) + " \\\n");
127             }
128             System.out.println("\\n");
129         }
130     }

```

6.61.2.2 static [ScalarStatistic](#) edu.ou.asgbook.linearity.LinearityVerifier.verify (int data[][], DataSelector selector, [DataTransform](#) transform, int neighSize) [static]

Returns the Mean Square Error statistic.

```

50
51     // setup
52     ScalarStatistic errorstat = new ScalarStatistic();
53     int nrows = data.length;
54     if ( nrows == 0 ){
55         return errorstat;
56     }
57     int ncols = data[0].length;
58     if ( ncols == 0 ){
59         return errorstat;
60     }
61
62     // find the error in every triad interpolating along rows
63     for (int col=0; col < ncols; ++col){
64         for (int row=neighSize; row < nrows-neighSize; ++row){
65             if (selector.shouldSelect(data[row][col], data[row-neighSize][col], data[row+neighSize][col])){
66                 int actualValue = data[row][col];
67                 double trans0 = transform.transform(data[row-neighSize][col]);
68                 double trans1 = transform.transform(data[row+neighSize][col]);
69                 double trans_interp = (trans0 + trans1)/2;

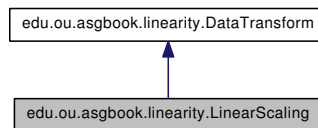
```

```
70         double interpValue = transform.inverse(trans_interp);
71         double error = (interpValue - actualValue);
72         errorstat.update(error*error);
73     }
74 }
75 }
76
77 // repeat for columns
78 for (int row=0; row < nrows; ++row){
79     for (int col=neighSize; col < ncols-neighSize; ++col){
80         if (selector.shouldSelect(data[row][col], data[row][col-neighSize], data[row][col+neighSize])){
81             int actualValue = data[row][col];
82             double trans0 = transform.transform(data[row][col-neighSize]);
83             double trans1 = transform.transform(data[row][col+neighSize]);
84             double trans_interp = (trans0 + trans1)/2;
85             double interpValue = transform.inverse(trans_interp);
86             double error = (interpValue - actualValue);
87             errorstat.update(error*error);
88         }
89     }
90 }
91
92 return errorstat;
93 }
```

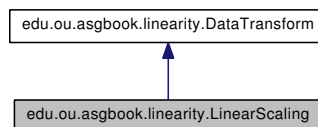
6.62 edu.ou.asgbook.linearity.LinearScaling Class Reference

Scales pixel values as Ax .

Inheritance diagram for edu.ou.asgbook.linearity.LinearScaling:



Collaboration diagram for edu.ou.asgbook.linearity.LinearScaling:



Public Member Functions

- [LinearScaling](#) (double multiplier)
Multiply input values by this amount.
- Override double [transform](#) (double value)
- Override double [inverse](#) (double value)

6.62.1 Detailed Description

Scales pixel values as Ax .

This is useful since the LatLonGrid stores integers.

Author:

valliappa.lakshmanan

6.62.2 Constructor & Destructor Documentation

6.62.2.1 `edu.ou.asgbook.linearity.LinearScaling.LinearScaling` (double *multiplier*)

Multiply input values by this amount.

```
17                                     {
18         this.scale = multiplier;
19     }
```

6.62.3 Member Function Documentation

6.62.3.1 Override double `edu.ou.asgbook.linearity.LinearScaling.inverse` (double *value*) [virtual]

Implements [edu.ou.asgbook.linearity.DataTransform](#).

```
27                                     {
28         return (value / scale);
29     }
```

6.62.3.2 Override double `edu.ou.asgbook.linearity.LinearScaling.transform` (double *value*) [virtual]

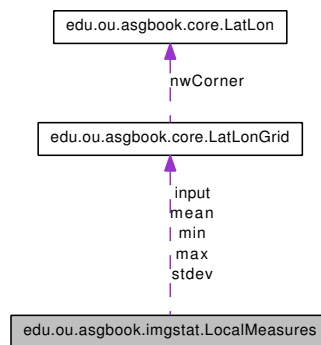
Implements [edu.ou.asgbook.linearity.DataTransform](#).

```
22                                     {
23         return (scale * value);
24     }
```


6.63 edu.ou.asgbook.imgstat.LocalMeasures Class Reference

Statistics computed in the neighborhood of a pixel.

Collaboration diagram for edu.ou.asgbook.imgstat.LocalMeasures:



Public Member Functions

- `LatLonGrid` `getMean` ()
- `LatLonGrid` `getStdDeviation` ()
- `LatLonGrid` `getMin` ()
- `LatLonGrid` `getMax` ()
- `LocalMeasures` (`LatLonGrid` `input`, `int` `Nx`, `int` `Ny`)

Static Public Member Functions

- `static void` `main` (`String`[] `args`) throws `Exception`

6.63.1 Detailed Description

Statistics computed in the neighborhood of a pixel.

Author:

valliappa.lakshmanan

6.63.2 Constructor & Destructor Documentation

6.63.2.1 `edu.ou.asgbook.imgstat.LocalMeasures.LocalMeasures` ([LatLonGrid](#) *input*, *int Nx*, *int Ny*)

```
41                                     {
42         this.hx = Nx/2;
43         this.hy = Ny/2;
44         this.input = input;
45         this.mean = LatLonGrid.copyOf(input);
46         this.stdev = LatLonGrid.copyOf(input);
47         this.min = LatLonGrid.copyOf(input);
48         this.max = LatLonGrid.copyOf(input);
49         compute();
50     }
```

6.63.3 Member Function Documentation

6.63.3.1 [LatLonGrid](#) `edu.ou.asgbook.imgstat.LocalMeasures.getMax` ()

```
37                                     {
38         return max;
39     }
```

6.63.3.2 [LatLonGrid](#) `edu.ou.asgbook.imgstat.LocalMeasures.getMean` ()

```
25                                     {
26         return mean;
27     }
```

6.63.3.3 [LatLonGrid](#) `edu.ou.asgbook.imgstat.LocalMeasures.getMin` ()

```
33                                     {
34         return min;
35     }
```

6.63.3.4 [LatLonGrid](#) `edu.ou.asgbook.imgstat.LocalMeasures.getStdDeviation` ()

```
29                                     {
30         return stdev;
31     }
```

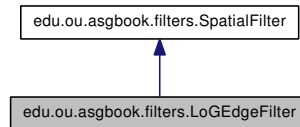
**6.63.3.5 static void edu.ou.asgbook.imgstat.LocalMeasures.main (String[] args)
throws Exception [static]**

```
86                                     {
87     // log-scaled population density
88     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulati
89     popdensity = popdensity.crop(900, 2500, 200, 200);
90     File out = OutputDirectory.getDefault("localstat");
91
92     KmlWriter.write(popdensity, out, "popdensity", PngWriter.createCoolToWarmColormap());
93     for (int neigh = 5; neigh < 12; neigh += 6){ // 5, 11
94         LocalMeasures stat = new LocalMeasures(popdensity, neigh, neigh);
95         KmlWriter.write(stat.getMean(), out, "mean_" + neigh, PngWriter.createCoolToWarmColormap());
96         KmlWriter.write(stat.getStdDeviation(), out, "stdev_" + neigh, PngWriter.createCoolToWarmCo
97         KmlWriter.write(stat.getMin(), out, "min_" + neigh, PngWriter.createCoolToWarmColormap());
98         KmlWriter.write(stat.getMax(), out, "max_" + neigh, PngWriter.createCoolToWarmColormap());
99     }
100 }
```

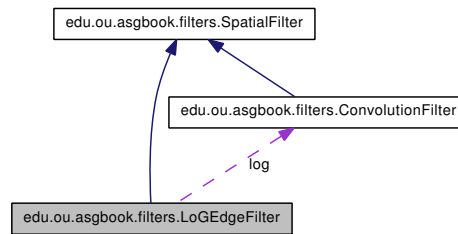
6.64 edu.ou.asgbook.filters.LoEdgeFilter Class Reference

Laplacian of a Gaussian edge filter.

Inheritance diagram for edu.ou.asgbook.filters.LoEdgeFilter:



Collaboration diagram for edu.ou.asgbook.filters.LoEdgeFilter:



Public Member Functions

- [LoEdgeFilter](#) (int halfsize, int edgethresh)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)
- [LatLonGrid edgeFilter](#) (final [LatLonGrid](#) input)
- [LatLonGrid edgeFilter](#) (final [LatLonGrid](#) input, File out)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.64.1 Detailed Description

Laplacian of a Gaussian edge filter.

Author:

valliappa.lakshmanan

6.64.2 Constructor & Destructor Documentation

6.64.2.1 edu.ou.asgbook.filters.LoEdgeFilter.LoEdgeFilter (int *halfsize*, int *edgethresh*)

```

21                                     {
22     double sigma = halfsize/3.0;
23     double[][] coeffs = new double[2*halfsize+1][2*halfsize+1];
24     double tot = 0;
25     for (int x=-halfsize; x <= halfsize; ++x){
26         for (int y=-halfsize; y <= halfsize; ++y){
27             double term1 = (x*x + y*y - sigma*sigma)/Math.pow(sigma,4);
28             double term2 = Math.exp(-(x*x + y*y)/(2*sigma*sigma));
29             double coeff = term1 * term2;
30             coeffs[x+halfsize][y+halfsize] = coeff;
31             tot += coeff;
32         }
33     }
34     // ensure that coeffs add up to zero
35     coeffs[halfsize][halfsize] -= tot;
36     this.log = new ConvolutionFilter(coeffs);
37     this.thresh = edgethresh;
38 }

```

6.64.3 Member Function Documentation

6.64.3.1 [LatLonGrid](#) edu.ou.asgbook.filters.LoEdgeFilter.edgeFilter (final [LatLonGrid](#) *input*, File *out*)

```

49                                     {
50     // Laplacian of a Gaussian
51     LatLonGrid dgg = log.convolve(input);
52     KmlWriter.debugWrite(dgg, out, "laplacianofgaussian");
53     int bound = log.getFilterNumRows();
54     // find zero crossings in 3x3 neighborhood
55     int nrows = dgg.getNumLat();
56     int ncols = dgg.getNumLon();
57     LatLonGrid result = new LatLonGrid(nrows,ncols,-1,dgg.getNwCorner(),dgg.getLatRes(),dgg.getLonRes());
58     for (int i=bound; i < (nrows-bound); ++i) for (int j=bound; j < (ncols-bound); ++j){
59         int mag = 0;
60         // ver
61         mag = checkZeroCrossing(dgg.getValue(i-1,j), dgg.getValue(i+1,j), mag);
62         // hor
63         mag = checkZeroCrossing(dgg.getValue(i,j-1), dgg.getValue(i,j+1), mag);
64         // diag1
65         mag = checkZeroCrossing(dgg.getValue(i-1,j-1), dgg.getValue(i+1,j+1), mag);
66         // diag2
67         mag = checkZeroCrossing(dgg.getValue(i+1,j-1), dgg.getValue(i-1,j+1), mag);
68         // mag is over 2 bins
69         mag /= 2;
70         if ( mag > thresh ){
71             result.setValue(i,j, mag);
72         }
73     }

```

```
74         return result;
75     }
```

6.64.3.2 [LatLonGrid](#) edu.ou.asgbook.filters.LoGEdgeFilter.edgeFilter (final [LatLonGrid](#) *input*)

```
45                                     {
46         return edgeFilter(input, null);
47     }
```

6.64.3.3 Override [LatLonGrid](#) edu.ou.asgbook.filters.LoGEdgeFilter.filter ([LatLonGrid](#) *input*)

```
41                                     {
42         return edgeFilter(input);
43     }
```

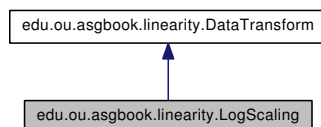
6.64.3.4 static void edu.ou.asgbook.filters.LoGEdgeFilter.main (String[] *args*) throws Exception [static]

```
84                                     {
85         // create output directory
86         File out = OutputDirectory.getDefault("logedge");
87
88         // read input
89         DataTransform t = new GlobalPopulation.LogScaling();
90         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, t).crop();
91         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
92
93         LoGEdgeFilter filter = new LoGEdgeFilter(5, 400);
94         LatLonGrid edges = filter.edgeFilter(popdensity, out);
95         KmlWriter.write(edges, out, "logedge", PngWriter.createCoolToWarmColormap());
96     }
```

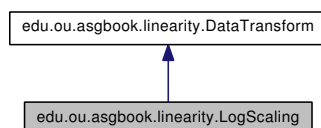
6.65 edu.ou.asgbook.linearity.LogScaling Class Reference

Transforms pixel values as $\log(x)$.

Inheritance diagram for edu.ou.asgbook.linearity.LogScaling:



Collaboration diagram for edu.ou.asgbook.linearity.LogScaling:



Public Member Functions

- [LogScaling](#) (double multiplier)
Multiply $\log(\text{input})$ values by this amount i.e.
- Override double [transform](#) (double value)
- Override double [inverse](#) (double value)

6.65.1 Detailed Description

Transforms pixel values as $\log(x)$.

Author:

valliappa.lakshmanan

6.65.2 Constructor & Destructor Documentation

6.65.2.1 edu.ou.asgbook.linearity.LogScaling.LogScaling (double *multiplier*)

Multiply $\log(\text{input})$ values by this amount i.e.

it is `multiplier*log(value)`

```
16                                     {
17         this.scale = multiplier;
18     }
```

6.65.3 Member Function Documentation

6.65.3.1 Override double `edu.ou.asgbook.linearity.LogScaling.inverse` (double *value*) [virtual]

Implements [edu.ou.asgbook.linearity.DataTransform](#).

```
30                                     {
31         if ( value == 0 ){
32             return 1;
33         } else {
34             return Math.pow(10, value/scale);
35         }
36     }
```

6.65.3.2 Override double `edu.ou.asgbook.linearity.LogScaling.transform` (double *value*) [virtual]

Implements [edu.ou.asgbook.linearity.DataTransform](#).

```
21                                     {
22         if ( value > 1 ){
23             return (scale*Math.log10(value));
24         } else {
25             return 0;
26         }
27     }
```


6.66 edu.ou.asgbook.dataset.MadisTemperature Class Reference

Reads the ASCII temperature data available at <http://madis-data.noaa.gov/public/sfcdumpguest.html>.

Static Public Member Functions

- static [PointObservations read](#) (File file) throws IOException
- static void [main](#) (String[] args) throws Exception

Static Public Attributes

- static final File [TN_Oct2010](#) = new File("data/madishydro/tn_oct2010_temp.txt")

Package Functions

- [SuppressWarnings](#) ("unused") public static [PointObservations read](#)(Reader r) throws IOException

6.66.1 Detailed Description

Reads the ASCII temperature data available at <http://madis-data.noaa.gov/public/sfcdumpguest.html>.

Author:

Valliappa.Lakshmanan

6.66.2 Member Function Documentation

6.66.2.1 static void edu.ou.asgbook.dataset.MadisTemperature.main (String[] args) throws Exception [static]

```
62                                     {
63     PointObservations data = MadisTemperature.read(MadisTemperature.TN_Oct2010);
64     for (int i=0; i < data.getPoints().length; ++i){
65         System.out.println(data.getPoints()[i]);
66     }
67 }
```

6.66.2.2 static [PointObservations](#) edu.ou.asgbook.dataset.MadisTemperature.read (File *file*) throws IOException [static]

```

29                                     {
30         Reader f = null;
31         if (file.getAbsolutePath().endsWith(".gz")) {
32             f = new InputStreamReader(new GZIPInputStream(new FileInputStream(
33                 file)));
34         } else {
35             f = new FileReader(file);
36         }
37         return read(f);
38     }

```

6.66.2.3 edu.ou.asgbook.dataset.MadisTemperature.SuppressWarnings ("unused") throws IOException [package]

```

41                                     {
42         Scanner s = new Scanner(r);
43         List<PointObservations.ObservationPoint> result = new ArrayList<PointObservations.ObservationPoint>();
44
45         final int FACTOR = 10;
46         final int MISSING = -99999 * FACTOR;
47         s.nextLine(); // header
48         while (s.hasNext()){
49             String station = s.next();
50             String date = s.next();
51             String time = s.next();
52             int precip = (int) Math.round( s.nextDouble() * FACTOR );
53             double lat = s.nextDouble();
54             double lon = s.nextDouble();
55             result.add(new PointObservations.ObservationPoint(lat,lon,precip));
56         }
57
58         PointObservations.ObservationPoint[] pts = result.toArray(new PointObservations.ObservationPoint[result.size()]);
59         return new PointObservations(pts, MISSING);
60     }

```

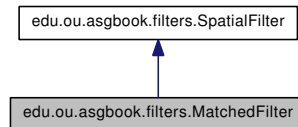
6.66.3 Member Data Documentation

6.66.3.1 final File [edu.ou.asgbook.dataset.MadisTemperature.TN_Oct2010](#) = new File("data/madishydro/tn_oct2010_temp.txt") [static]

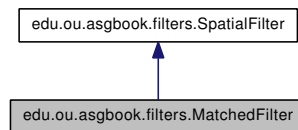
6.67 edu.ou.asgbook.filters.MatchedFilter Class Reference

Convolve an image by a window that is akin to the features we want to extract.

Inheritance diagram for edu.ou.asgbook.filters.MatchedFilter:



Collaboration diagram for edu.ou.asgbook.filters.MatchedFilter:



Public Member Functions

- [MatchedFilter](#) (double[][] coeffs)
- [LatLonGrid match](#) (final [LatLonGrid](#) input)
returns a grid with values in the range 0-100
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.67.1 Detailed Description

Convolve an image by a window that is akin to the features we want to extract.

Author:

Valliappa.Lakshmanan

6.67.2 Constructor & Destructor Documentation

6.67.2.1 `edu.ou.asgbook.filters.MatchedFilter.MatchedFilter (double coeffs [] [])`

```

24                                     {
25         this.coeffs = coeffs;
26         if ( coeffs.length % 2 == 0 || coeffs[0].length % 2 == 0 ){
27             throw new IllegalArgumentException("Dimensions of coefficients array needs to be even");
28         }
29         // normalize
30         double sum = 0;
31         for (int i=0; i < coeffs.length; ++i){
32             for (int j=0; j < coeffs[i].length; ++j){
33                 sum += coeffs[i][j];
34             }
35         }
36         System.out.println("Normalizing coefficients by " + sum);
37         DecimalFormat df = new DecimalFormat("0.000");
38         for (int i=0; i < coeffs.length; ++i){
39             for (int j=0; j < coeffs[i].length; ++j){
40                 coeffs[i][j] /= sum;
41                 System.out.print(df.format(coeffs[i][j]) + "&"); // for LaTeX
42             }
43             System.out.println("\\\\");
44         }
45     }

```

6.67.3 Member Function Documentation

6.67.3.1 Override [LatLonGrid](#) `edu.ou.asgbook.filters.MatchedFilter.filter (LatLonGrid input)`

```

124                                     {
125         return match(input);
126     }

```

6.67.3.2 `static void edu.ou.asgbook.filters.MatchedFilter.main (String [] args) throws Exception` [static]

```

83                                     {
84         // create output directory
85         File out = OutputDirectory.getDefault("matched");
86
87         // read input
88         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulation());
89         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
90
91         for (int hx=1; hx <= 5; hx += 2){
92             int hy = hx*2;
93             double [ ] [ ] coeffs = new double[2*hx+1][2*hy+1];
94             for (int i=0; i < coeffs.length; ++i){

```

```

95         for (int j=0; j < coeffs[i].length; ++j){
96             int t = i + j;
97             if ( t < coeffs.length ) coeffs[i][j] = 1;
98         }
99     }
100     MatchedFilter filter = new MatchedFilter(coeffs);
101     LatLonGrid sm = filter.match(popdensity);
102     KmlWriter.write(sm, out, "northwest"+hx, PngWriter.createCoolToWarmColormap());
103 }
104
105 for (int hx=5; hx < 15; hx += 3){
106     // int hx = 8;
107     int hy = hx;
108     double[][] coeffs = new double[2*hx+1][2*hy+1];
109     for (int i=0; i < coeffs.length; ++i){
110         for (int j=0; j < coeffs[i].length; ++j){
111             int dx = i - hx;
112             int dy = j - hy;
113             if ( Math.abs(dx) < hx/2 && Math.abs(dy) < hy/2 )
114                 coeffs[i][j] = 1;
115         }
116     }
117     MatchedFilter filter = new MatchedFilter(coeffs);
118     LatLonGrid sm = filter.match(popdensity);
119     KmlWriter.write(sm, out, "isolated"+hx, PngWriter.createCoolToWarmColormap());
120 }
121 }

```

6.67.3.3 [LatLonGrid](#) edu.ou.asgbook.filters.MatchedFilter.match (final [LatLonGrid](#) input)

returns a grid with values in the range 0-100

```

50     {
51         LatLonGrid output = LatLonGrid.copyOf(input);
52         output.setMissing(-1);
53         output.fill(output.getMissing());
54         int[][] outData = output.getData();
55         int[][] inData = input.getData();
56         final int hx = coeffs.length / 2;
57         final int hy = coeffs[0].length / 2;
58         final int nx = output.getNumLat();
59         final int ny = output.getNumLon();
60         for (int i=hx; i < (nx-hx); ++i){
61             for (int j=hy; j < (ny-hy); ++j){
62                 double tot = 0;
63                 int totval = 0; // normalize values in window
64                 for (int m=-hx; m <= hx; ++m){
65                     for (int n=-hy; n <= hy; ++n){
66                         double coeff = coeffs[m+hx][n+hy];
67                         int inval = inData[i+m][j+n];
68                         if (inval != input.getMissing()){
69                             tot += inval*coeff;
70                             totval += inval;

```

```
71         }
72     }
73 }
74     if (totval != 0){
75         outData[i][j] = (int) Math.round(10000 * tot / totval);
76     }
77 }
78 }
79     return output;
80 }
```

6.68 edu.ou.asgbook.filters.MaxValueFilter Class Reference

Finds the highest value pixel in the image.

Public Member Functions

- Result [findHighestValued](#) ([LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

Classes

- class **Result**

6.68.1 Detailed Description

Finds the highest value pixel in the image.

Author:

Valliappa.Lakshmanan

6.68.2 Member Function Documentation

6.68.2.1 Result edu.ou.asgbook.filters.MaxValueFilter.findHighestValued ([LatLonGrid](#) input)

```
27                                     {
28     int[][] data = input.getData();
29     int x = -1;
30     int y = -1;
31     int maxval = input.getMissing();
32     for (int i=0; i < input.getNumLat(); ++i){
33         for (int j=0; j < input.getNumLon(); ++j){
34             if ( data[i][j] != input.getMissing() ){
35                 if ( maxval == input.getMissing() ||
36                     maxval < data[i][j] ){
37                     x = i; // new maximum
38                     y = j;
39                     maxval = data[x][y];
40                 }
41             }
42         }
43     }
```

```
41         }
42     }
43 }
44 if ( x >=0 && y >= 0 ){
45     LatLon loc = input.getLocation(x, y);
46     return new Result(data[x][y], loc);
47 }
48 return null;
49 }
```

6.68.2.2 static void edu.ou.asgbook.filters.MaxValueFilter.main (String[] args)
throws Exception [static]

```
51                                     {
52     // read input
53     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
54     popdensity.setMissing(0); // will get to process less data this way
55
56     // find 10 highest
57     MaxValueFilter filter = new MaxValueFilter();
58     Result result = filter.findHighestValued(popdensity);
59     System.out.println("Maximum is " + result.value + " at " + result.location);
60 }
```


6.69 edu.ou.asgbook.thinning.MedialAxis-Skeletonization Class Reference

The MAT method of skeletonizing a grid.

Static Public Member Functions

- static [LatLonGrid](#) [findSkeleton](#) ([LatLonGrid](#) input, int thresh, File out) throws Exception
- static void [main](#) (String[] args) throws Exception

6.69.1 Detailed Description

The MAT method of skeletonizing a grid.

Author:

v.lakshmanan

6.69.2 Member Function Documentation

6.69.2.1 static [LatLonGrid](#) edu.ou.asgbook.thinning.MedialAxis-Skeletonization.findSkeleton ([LatLonGrid](#) input, int thresh, File out) throws Exception [static]

```

27
28         // threshold and invert
29         LatLonGrid binaryImage = new SimpleThresholder(thresh).threshold(input);
30         if (out != null){
31             KmlWriter.write(binaryImage, out, "thresh", PngWriter.createCoolToWarmColormap());
32         }
33         binaryImage = new Inverter(1).invert(binaryImage);
34
35         // compute distance to pts > 0 i.e. boundary pixels
36         LatLonGrid edt = new EuclideanDTSaito().getDistanceTransform(binaryImage, 0);
37         if (out != null){
38             KmlWriter.write(edt, out, "edt", PngWriter.createCoolToWarmColormap());
39         }
40
41         // retain local maximum in 4-neighborhood
42         LatLonGrid result = new LatLonGrid(edt.getNumLat(),edt.getNumLon(),edt.getMissing(),edt.getNwCo
43         for (int i=1; i < edt.getNumLat()-1; ++i){
44             for (int j=1; j < edt.getNumLon()-1; ++j){
45                 int edtval = edt.getValue(i, j);
46                 if ( edtval != 0 &&
47                     edt.getValue(i-1,j) <= edtval &&
48                     edt.getValue(i,j-1) <= edtval &&
```

```
49         edt.getValue(i+1,j) <= edtval &&
50         edt.getValue(i,j+1) <= edtval ){
51             result.setValue(i,j, 1);
52         } else {
53             result.setValue(i,j, 0);
54         }
55     }
56 }
57 return result;
58 }
```

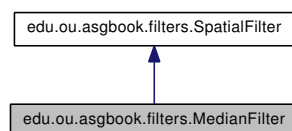
6.69.2.2 static void edu.ou.asgbook.thinning.MedialAxisSkeletonization.main (String[] args) throws Exception [static]

```
60 {
61     File out = OutputDirectory.getDefault("matskeleton");
62     LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new Glo
63     KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
64
65     popdensity = new DilateErodeFilter(2,3).filter(popdensity);
66     // popdensity = new ErosionFilter(3).filter(popdensity);
67     popdensity = new ErodeDilateFilter(2,3).filter(popdensity);
68     KmlWriter.write(popdensity, out, "filledin", PngWriter.createCoolToWarmColormap());
69
70     LatLonGrid result = findSkeleton(popdensity, 300, out);
71     result.setMissing(0); // to make the ls pop out
72     KmlWriter.write(result, out, "mat", PngWriter.createCoolToWarmColormap());
73 }
```

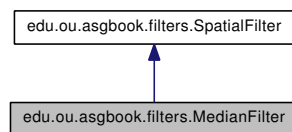
6.70 edu.ou.asgbook.filters.MedianFilter Class Reference

A smoothing operation that involves replacing a pixel by the local median.

Inheritance diagram for edu.ou.asgbook.filters.MedianFilter:



Collaboration diagram for edu.ou.asgbook.filters.MedianFilter:



Public Member Functions

- [MedianFilter](#) (int halfSize)
- Override [LatLonGrid filter](#) (LatLonGrid input)
- [LatLonGrid smooth](#) (final LatLonGrid input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.70.1 Detailed Description

A smoothing operation that involves replacing a pixel by the local median.

Author:

Valliappa.Lakshmanan

6.70.2 Constructor & Destructor Documentation

6.70.2.1 `edu.ou.asgbook.filters.MedianFilter.MedianFilter (int halfSize)`

```

23                                     {
24         this.halfSize = halfSize;
25     }
```

6.70.3 Member Function Documentation

6.70.3.1 Override [LatLonGrid](#) `edu.ou.asgbook.filters.MedianFilter.filter (LatLonGrid input)`

```

28                                     {
29         return smooth(input);
30     }
```

6.70.3.2 `static void edu.ou.asgbook.filters.MedianFilter.main (String[] args) throws Exception [static]`

```

62                                     {
63         // create output directory
64         File out = OutputDirectory.getDefault("median");
65
66         // read input
67         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GL
68         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
69
70         // dilate
71         LatLonGrid dilate1 = new MedianFilter(1).smooth(popdensity);
72         KmlWriter.write(dilate1, out, "median_3", PngWriter.createCoolToWarmColormap());
73         LatLonGrid dilate3 = new MedianFilter(3).smooth(popdensity);
74         KmlWriter.write(dilate3, out, "median_7", PngWriter.createCoolToWarmColormap());
75         LatLonGrid dilate5 = new MedianFilter(5).smooth(popdensity);
76         KmlWriter.write(dilate5, out, "median_11", PngWriter.createCoolToWarmColormap());
77     }
```

6.70.3.3 [LatLonGrid](#) `edu.ou.asgbook.filters.MedianFilter.smooth (final LatLonGrid input)`

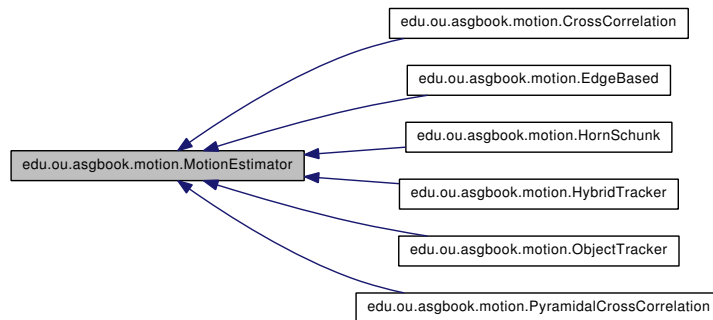
```

32                                     {
33         LatLonGrid output = LatLonGrid.copyOf(input);
34         output.fill(output.getMissing());
35         int[][] outData = output.getData();
36         int[][] inData = input.getData();
37         int hx = halfSize;
38         int hy = halfSize;
39         int nx = inData.length;
```

```
40         int ny = inputData[0].length;
41         int[] arr = new int[(2*hx+1)*(2*hy+1)];
42         for (int i=hx; i < (nx-hx); ++i){
43             for (int j=hy; j < (ny-hy); ++j){
44                 int nelements = 0;
45                 for (int m=-hx; m <= hx; ++m){
46                     for (int n=-hy; n <= hy; ++n){
47                         int inval = inputData[i+m][j+n];
48                         if (inval != input.getMissing()){
49                             arr[nelements] = inval;
50                             ++nelements;
51                         }
52                     }
53                 }
54                 if (nelements > 0){
55                     outData[i][j] = QuickSelect.kth_element(arr, nelements, nelements/2);
56                 }
57             }
58         }
59         return output;
60     }
```

6.71 edu.ou.asgbook.motion.MotionEstimator Interface Reference

Inheritance diagram for edu.ou.asgbook.motion.MotionEstimator:



Public Member Functions

- Pair< [LatLonGrid](#), [LatLonGrid](#) > [compute](#) ([LatLonGrid](#) data0, [LatLonGrid](#) data1, File outdir)

returns motion in the two directions.

6.71.1 Member Function Documentation

6.71.1.1 Pair<[LatLonGrid](#), [LatLonGrid](#)> edu.ou.asgbook.motion.MotionEstimator.compute ([LatLonGrid](#) data0, [LatLonGrid](#) data1, File outdir)

returns motion in the two directions.

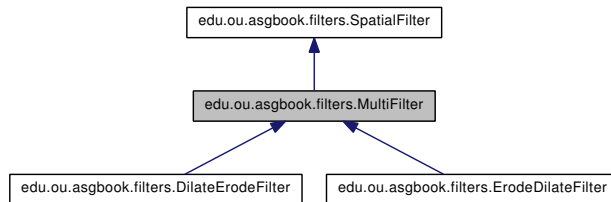
The first one is north to south and the second one is east to west. The data is aligned to second time frame. The output dir is used for intermediate products and may be null.

Implemented in [edu.ou.asgbook.motion.CrossCorrelation](#), [edu.ou.asgbook.motion.EdgeBased](#), [edu.ou.asgbook.motion.HornSchunk](#), [edu.ou.asgbook.motion.HybridTracker](#), [edu.ou.asgbook.motion.ObjectTracker](#), and [edu.ou.asgbook.motion.PyramidalCrossCorrelation](#).

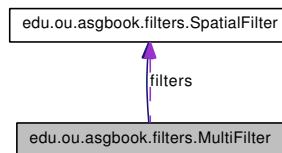
6.72 edu.ou.asgbook.filters.MultiFilter Class Reference

Carries out multiple operations.

Inheritance diagram for edu.ou.asgbook.filters.MultiFilter:



Collaboration diagram for edu.ou.asgbook.filters.MultiFilter:



Public Member Functions

- [MultiFilter](#) ([SpatialFilter](#)[] filters, int numTimes)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)

6.72.1 Detailed Description

Carries out multiple operations.

Author:

Valliappa.Lakshmanan

6.72.2 Constructor & Destructor Documentation

6.72.2.1 edu.ou.asgbook.filters.MultiFilter.MultiFilter ([SpatialFilter](#)[] filters, int numTimes)

17

{

```
18         super();
19         this.filters = filters;
20         this.numTimes = numTimes;
21     }
```

6.72.3 Member Function Documentation

6.72.3.1 Override [LatLonGrid](#) edu.ou.asgbook.filters.MultiFilter.filter ([LatLonGrid](#) *input*)

```
24         {
25         LatLonGrid output = LatLonGrid.copyOf(input);
26         for (int i=0; i < numTimes; ++i){
27             for (SpatialFilter filter : filters){
28                 output = filter.filter(output);
29             }
30         }
31         return output;
32     }
```


6.73 edu.ou.asgbook.segmentation.MultiscaleKMeans-Segmenter Class Reference

Quantizes image into K levels, then does multiscale segmentation Does not implement the pruning techniques discussed in the paper.

Public Member Functions

- [MultiscaleKMeansSegmenter](#) (int thresh1, int thresh2, int K)

Specify contouring levels.

- [LatLonGrid quantize](#) ([LatLonGrid](#) data, File out)

Contours grid into levels 1,2,3.

- List< [LabelResult](#) > [label](#) ([LatLonGrid](#) data, File out)

Returns K scales of output.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

Classes

- class [Cluster](#)

6.73.1 Detailed Description

Quantizes image into K levels, then does multiscale segmentation Does not implement the pruning techniques discussed in the paper.

See: V. Lakshmanan, R. Rabin, and V. DeBrunner, "Multiscale storm identification and forecast," J. Atm. Res., vol. 67, pp. 367-380, July 2003

Author:

v.lakshmanan

6.73.2 Constructor & Destructor Documentation

6.73.2.1 `edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter.MultiscaleKMeansSegmenter (int thresh1, int thresh2, int K)`

Specify contouring levels.

```

37                                     {
38         this.THRESH1 = thresh1;
39         this.THRESH2 = thresh2;
40         this.K = K;
41         this.INCR = (double) (THRESH2 - THRESH1) / K;
42     }
```

6.73.3 Member Function Documentation

6.73.3.1 `List<LabelResult> edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter.label (LatLonGrid data, File out)`

Returns K scales of output.

```

185                                     {
186         List<LabelResult> result = new ArrayList<LabelResult>();
187         LatLonGrid levels = quantize(data, out);
188         for (int thresh=1; thresh <= K; ++thresh){
189             ThresholdSegmenter seg = new ThresholdSegmenter(thresh);
190             LabelResult label = seg.label(levels);
191             if (out != null){
192                 try {
193                     KmlWriter.write(label.label, out, "label_" + thresh, PngWriter.createC
194                 } catch (Exception e) {
195                     System.err.println(e);
196                 }
197             }
198             result.add(label);
199         }
200         return result;
201     }
```

6.73.3.2 `static void edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter.main (String[] args) throws Exception` [static]

```

203                                     {
204         File out = OutputDirectory.getDefault("multiscalekmeans");
205     }
```

6.73 edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter Class Reference

```
206         // data
207         LatLonGrid conus = SurfaceAlbedo.read(SurfaceAlbedo.CONUS, 100).crop(100, 100, 500, 200);
208         KmlWriter.write(conus, out, "orig", PngWriter.createCoolToWarmColormap());
209
210         new MultiscaleKMeansSegmenter(20,25,5).label(conus, out);
211     }
```

6.73.3.3 [LatLonGrid](#) edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter.quantize ([LatLonGrid](#) data, File out)

Contours grid into levels 1,2,3.

..K

```
123                                     {
124         final int nrows = data.getNumLat();
125         final int ncols = data.getNumLon();
126
127         // initialize based on simple quantization
128         LatLonGrid seed = LatLonGrid.copyOf(data);
129         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
130             int levelno = 0;
131             if (data.getValue(i, j) != data.getMissing()){
132                 levelno = (int) Math.round((data.getValue(i, j) - THRESH1)/INCR);
133                 if ( levelno < 0 ) levelno = 0;
134                 else if ( levelno > K ) levelno = K;
135             }
136             seed.setValue(i, j, levelno);
137         }
138         if (out != null){
139             try {
140                 KmlWriter.write(seed, out, "levels_0", PngWriter.createCoolToWarmColormap());
141             } catch (Exception e) {
142                 System.err.println(e);
143             }
144         }
145
146         // Start K-means
147         int iter = 1;
148         int n_changed = 0;
149         do {
150             // compute means: could get away with simply using center of data range ...
151             Cluster[] clusters = findClusters(data, seed, K);
152             // move pixels
153             LatLonGrid next = LatLonGrid.copyOf(seed);
154             n_changed = 0;
155             for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
156                 if ( data.getValue(i, j) != data.getMissing() ){
157                     int closest = findClosestCluster(data.getValue(i, j), i, j, seed, clusters);
158                     if (closest != seed.getValue(i, j)){
159                         // change the label to closest
160                         next.setValue(i, j, closest);
161                         ++n_changed;
162                     }
163                 }
164             }
165             iter++;
166             if (n_changed == 0) break;
167         } while (true);
168     }
```

```
162         }
163     }
164 }
165 System.out.println("Changing " + n_changed + " at " + iter + " th iteration");
166 // for next step
167 seed = next;
168
169 if (out != null){
170     try {
171         KmlWriter.write(seed, out, "levels_" + iter, PngWriter.createCoolToWarm);
172     } catch (Exception e) {
173         System.err.println(e);
174     }
175 }
176
177 ++iter;
178 } while (iter < MAX_ITER && n_changed > 0);
179 return seed;
180 }
```

6.74 edu.ou.asgbook.filters.NHighest Class Reference

Finds the N highest valued-pixels in image.

Public Member Functions

- [NHighest](#) (int n)
- [Pixel\[\] findHighestValued](#) ([LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.74.1 Detailed Description

Finds the N highest valued-pixels in image.

Author:

Valliappa.Lakshmanan

6.74.2 Constructor & Destructor Documentation

6.74.2.1 edu.ou.asgbook.filters.NHighest.NHighest (int n)

```

27                                     {
28         this.n = n;
29     }
```

6.74.3 Member Function Documentation

6.74.3.1 [Pixel \[\]](#) edu.ou.asgbook.filters.NHighest.findHighestValued ([LatLonGrid](#) input)

```

31                                     {
32         // create array of pixels
33         int[][] data = input.getData();
34         final int initialCapacity = (input.getNumLat() * input.getNumLon()) / 10;
35         List<Pixel> a = new ArrayList<Pixel>(initialCapacity);
36         for (int i=0; i < input.getNumLat(); ++i){
37             for (int j=0; j < input.getNumLon(); ++j){
38                 if ( data[i][j] != input.getMissing() ){
39                     a.add(new Pixel(i, j, data[i][j]));
40                 }
41             }
42         }
```

```

41         }
42     }
43     System.out.println("Finding the " + n + " highest values out of " + a.size() + " pixels");
44
45     // selection sort this array to find n highest
46     Pixel[] result = new Pixel[n];
47     Pixel.CompareValue comparator = new Pixel.CompareValue();
48     for (int i=0; i < n; ++i){
49         int p = i;
50         for (int j=i; j < a.size(); ++j){
51             if ( comparator.compare(a.get(j), a.get(p)) > 0 ){
52                 p = j;
53             }
54         }
55         result[i] = a.get(p);
56         // swap a[i] and a[p]
57         Pixel temp = a.get(i);
58         a.set(i, a.get(p));
59         a.set(p, temp);
60     }
61     return result;
62 }

```

6.74.3.2 static void edu.ou.asgbook.filters.NHighest.main (String[] args) throws Exception [static]

```

64         {
65             // read input
66             LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
67             popdensity.setMissing(0); // will get to process less data this way
68
69             // find 10 highest
70             NHighest filter = new NHighest(10);
71             Pixel[] result = filter.findHighestValued(popdensity);
72             for (int i=0; i < result.length; ++i){
73                 System.out.println(i + " " + result[i] + " loc=" + popdensity.getLocation(result[i]));
74             }
75
76             // plot the result on a map
77             popdensity.fill(popdensity.getMissing());
78             for (int i=0; i < result.length; ++i){
79                 popdensity.setValue(result[i].getX(), result[i].getY(), 1);
80             }
81             File out = OutputDirectory.getDefault("nhighest");
82             KmlWriter.write(popdensity, out, "highest10", PngWriter.createHotColormap());
83
84             // plot as KML points
85             List<LatLon> points = new ArrayList<LatLon>();
86             List<String> names = new ArrayList<String>();
87             for (int i=0; i < result.length; ++i){
88                 Pixel p = result[i];
89                 points.add( popdensity.getLocation(p.getRow(), p.getCol()));
90                 names.add("Pixel#" + (i+1) );
91             }
92             KmlWriter.write(points, names, out, "top10pixels");

```

93 }

6.75 edu.ou.asgbook.filters.NHighestLevelSetImpl Class Reference

Finds the N highest valued-pixels in image using a levelset implementation.

Public Member Functions

- [NHighestLevelSetImpl](#) (int nth)
- [Pixel](#)[] [findHighestValued](#) ([LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.75.1 Detailed Description

Finds the N highest valued-pixels in image using a levelset implementation.

Author:

Valliappa.Lakshmanan

6.75.2 Constructor & Destructor Documentation

6.75.2.1 edu.ou.asgbook.filters.NHighestLevelSetImpl.NHighestLevelSetImpl (int *nth*)

```

30                                     {
31         this.nth = nth;
32     }
```

6.75.3 Member Function Documentation

6.75.3.1 [Pixel](#)[] [edu.ou.asgbook.filters.NHighestLevelSetImpl.findHighest-Valued](#) ([LatLonGrid](#) *input*)

```

34                                     {
35         // create level set
36         LevelSet levelset = LevelSet.newInstance(input);
37
38         // find the top n pixels
39         Map.Entry<Integer, List<Pixel>>[] levels = levelset.getLevels();
40         List<Pixel> result = new ArrayList<Pixel>();
```



```

41         int curr = levels.length;
42         while (result.size() < nth && curr > 0){
43             --curr; // next
44             result.addAll(levels[curr].getValue()); // all pixels at this level
45         }
46
47         return result.toArray(new Pixel[0]);
48     }

```

6.75.3.2 static void edu.ou.asgbook.filters.NHighestLevelSetImpl.main (String[] args) throws Exception [static]

```

50                                     {
51         // read input
52         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
53         popdensity.setMissing(0); // will get to process less data this way
54
55         // find 10 highest
56         NHighestLevelSetImpl filter = new NHighestLevelSetImpl(10);
57         Pixel[] result = filter.findHighestValued(popdensity);
58         for (int i=0; i < result.length; ++i){
59             System.out.println(i + " " + result[i] + " loc=" + popdensity.getLocation(result[i].getX(),
60         )
61
62         // plot the result on a map
63         popdensity.fill(popdensity.getMissing());
64         for (int i=0; i < result.length; ++i){
65             popdensity.setValue(result[i].getX(), result[i].getY(), 1);
66         }
67         File out = OutputDirectory.getDefault("levelset");
68         KmlWriter.write(popdensity, out, "highest10", PngWriter.createHotColormap());
69
70         // plot as KML points
71         List<LatLon> points = new ArrayList<LatLon>();
72         List<String> names = new ArrayList<String>();
73         for (int i=0; i < result.length; ++i){
74             Pixel p = result[i];
75             points.add( popdensity.getLocation(p.getRow(), p.getCol()));
76             names.add("Pixel#"+ (i+1) );
77         }
78         KmlWriter.write(points, names, out, "top10pixels");
79     }

```

6.76 edu.ou.asgbook.dataset.NighttimeLights Class Reference

Reads night-time lights data in ESRI grid format.

Static Public Member Functions

- static [LatLonGrid](#) [read](#) (File file) throws IOException
- static void [main](#) (String[] args) throws Exception

Static Public Attributes

- static File [WORLD](#) = new File("data/nighttime/nighttimelights.txt.gz")

6.76.1 Detailed Description

Reads night-time lights data in ESRI grid format.

Author:

Valliappa.Lakshmanan

6.76.2 Member Function Documentation

6.76.2.1 static void edu.ou.asgbook.dataset.NighttimeLights.main (String[] args) throws Exception [static]

```

30                                     {
31         // create output directory
32         File out = OutputDirectory.getDefault("nighttime");
33
34         // read input
35         LatLonGrid lights = NighttimeLights.read(NighttimeLights.WORLD);
36
37         // write out as image, for viewing
38         KmlWriter.write(lights, out, "lights", PngWriter.createCoolToWarmColormap());
39     }
```

6.76.2.2 static [LatLonGrid](#) edu.ou.asgbook.dataset.NighttimeLights.read (File file) throws IOException [static]

```

26                                     {
27         return EsriGrid.read(file, new LinearScaling(100.0/63)); // 0-100
28     }
```

6.76.3 Member Data Documentation

6.76.3.1 File `edu.ou.asgbook.dataset.NighttimeLights.WORLD` = `new File("data/nighttime/nighttimelights.txt.gz")` [static]

6.77 edu.ou.asgbook.oban.ObjectiveAnalysisUtils Class Reference

Utility functions for objective analysis.

Static Public Member Functions

- static [LatLonGrid](#) [createBoundingGrid](#) ([PointObservations](#) data, double latres, double lonres)
- static double [computeMeanDistance](#) ([PointObservations](#) data)

6.77.1 Detailed Description

Utility functions for objective analysis.

Author:

Valliappa.Lakshmanan

6.77.2 Member Function Documentation

6.77.2.1 static double edu.ou.asgbook.oban.ObjectiveAnalysisUtils.computeMeanDistance ([PointObservations](#) data) [static]

```

60                                     {
61         PointObservations.ObservationPoint[] points = data.getPoints();
62         if ( points.length < 1 ){
63             throw new IllegalArgumentException("Number of points has be greater than one");
64         }
65
66         double totdist = 0;
67         for (int i=0; i < points.length; ++i){
68             double mindistsq = Double.MAX_VALUE;
69             for (int j=0; j < points.length; ++j){
70                 if ( j != i ){
71                     double latdist = points[i].getLat() - points[j].getLat();
72                     double londist = points[i].getLon() - points[j].getLon();
73                     double distsq = (latdist*latdist + londist*londist);
74                     if ( distsq < mindistsq ){
75                         mindistsq = distsq;
76                     }
77                 }
78             }
79             totdist += Math.sqrt(mindistsq);
80         }
81     }

```

```

82         return totdist / points.length;
83     }

```

6.77.2.2 static [LatLonGrid](#) edu.ou.asgbook.oban.ObjectiveAnalysisUtils.createBoundingGrid ([PointObservations](#) data, double latres, double lonres) [static]

```

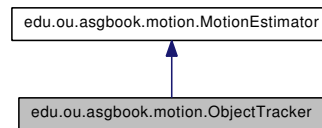
17
18     PointObservations.ObservationPoint[] points = data.getPoints();
19     if ( points.length == 0 ){
20         throw new IllegalArgumentException("Number of points has be greater than zero");
21     }
22
23     // find bounding box
24     double minlat = 90;
25     double maxlat = -90;
26     double minlon = 180;
27     double maxlon = -180;
28     for (int i=0; i < points.length; ++i){
29         if (points[i].getLat() > maxlat){
30             maxlat = points[i].getLat();
31         }
32         if (points[i].getLat() < minlat){
33             minlat = points[i].getLat();
34         }
35         if (points[i].getLon() > maxlon){
36             maxlon = points[i].getLon();
37         }
38         if (points[i].getLon() < minlon){
39             minlon = points[i].getLon();
40         }
41     }
42
43     // go a little bit off to the side and roundoff so that grid bounds are multiples of res
44     minlat = round(minlat - latres, latres);
45     maxlat = round(maxlat + latres, latres);
46     minlon = round(minlon - lonres, lonres);
47     maxlon = round(maxlon + lonres, lonres);
48
49     int nrows = (int) Math.round((maxlat - minlat)/latres);
50     int ncols = (int) Math.round((maxlon - minlon)/lonres);
51
52     System.out.println(points.length + " points will fit inside a " + nrows + "x" + ncols + " grid"
53     return new LatLonGrid(nrows, ncols, data.getMissing(), new LatLon(maxlat,minlon), latres, lonre
54 }

```

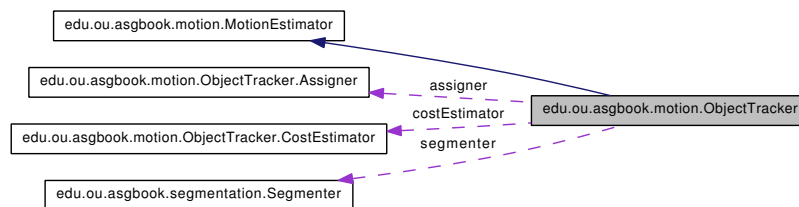
6.78 edu.ou.asgbook.motion.ObjectTracker Class Reference

Estimates motion based on assigning objects in one frame to objects in the previous frame.

Inheritance diagram for edu.ou.asgbook.motion.ObjectTracker:



Collaboration diagram for edu.ou.asgbook.motion.ObjectTracker:



Public Member Functions

- [ObjectTracker](#) (int hysThresh1, int hysThresh2, int minsize)
- [ObjectTracker](#) ([Segmenter](#) seg, [CostEstimator](#) cost, [Assigner](#) a)
- Override `Pair< LatLonGrid, LatLonGrid > compute (LatLonGrid data0, LatLonGrid data1, File outdir)`

returns motion in the two directions.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

Classes

- interface [Assigner](#)
- class [CentroidDistance](#)

- interface [CostEstimator](#)
- class **GreedyAssignment**
- class **SimpleSegmenter**

6.78.1 Detailed Description

Estimates motion based on assigning objects in one frame to objects in the previous frame.

Author:

v.lakshmanan

6.78.2 Constructor & Destructor Documentation

6.78.2.1 edu.ou.asgbook.motion.ObjectTracker.ObjectTracker (int *hysThresh1*, int *hysThresh2*, int *minsize*)

```

101                                     {
102         this(new SimpleSegmenter(hysThresh1, hysThresh2, minsize),
103             new CentroidDistance(), new GreedyAssignment());
104     }
```

6.78.2.2 edu.ou.asgbook.motion.ObjectTracker.ObjectTracker ([Segmenter](#) *seg*, [CostEstimator](#) *cost*, [Assigner](#) *a*)

```

106                                     {
107         segmenter = seg;
108         costEstimator = cost;
109         assigner = a;
110     }
```

6.78.3 Member Function Documentation

6.78.3.1 Override [Pair](#)<[LatLonGrid](#), [LatLonGrid](#)> edu.ou.asgbook.motion.ObjectTracker.compute ([LatLonGrid](#) *data0*, [LatLonGrid](#) *data1*, *File outdir*)

returns motion in the two directions.

The first one is north to south and the second one is east to west. The data is aligned to second time frame. The output dir is used for intermediate products and may be null.

Implements [edu.ou.asgbook.motion.MotionEstimator](#).

```

161
162     LabelResult objects0 = segmenter.label(data0);
163     LabelResult objects1 = segmenter.label(data1);
164
165     if (outdir != null){
166         try {
167             KmlWriter.write(objects0.label, outdir, "objects0", PngWriter.createRandomName());
168             KmlWriter.write(objects1.label, outdir, "objects1", PngWriter.createRandomName());
169         } catch (Exception e) {
170             e.printStackTrace();
171         }
172     }
173
174     // match the objects across frames
175     RegionProperty[] regions0 = RegionProperty.compute(objects0, data0);
176     RegionProperty[] regions1 = RegionProperty.compute(objects1, data1);
177     int[][] cost = computeCost(regions0, regions1);
178     int[] assigned = getAssignments(cost, outdir);
179
180     // find motion for each region
181     int[] regu = new int[assigned.length];
182     int[] regv = new int[assigned.length];
183     for (int i=1; i < assigned.length; ++i){
184         int oldregno = assigned[i];
185         if ( oldregno > 0 ){
186             double cx = regions1[i].getCx();
187             double cy = regions1[i].getCy();
188             double oldcx = regions0[oldregno].getCx();
189             double oldcy = regions0[oldregno].getCy();
190             regu[i] = (int) Math.round( (cx - oldcx)*MOT_SCALE );
191             regv[i] = (int) Math.round( (cy - oldcy)*MOT_SCALE );
192             // System.out.println("Object at " + cx + "," + cy + " moving at " + regu[i] + " " + regv[i]);
193         }
194     }
195
196     // apply the motion estimate based on assignment to all pixels
197     LatLonGrid u = new LatLonGrid(data0.getNumLat(), data0.getNumLon(), 0, data0.getNwPixel());
198     LatLonGrid v = LatLonGrid.copyOf(u);
199     for (int i=0; i < u.getNumLat(); ++i) for (int j=0; j < u.getNumLon(); ++j){
200         int regno = objects1.label.getValue(i,j);
201         if ( regno > 0 ){
202             u.setValue(i,j, regu[regno]);
203             v.setValue(i,j, regv[regno]);
204         }
205     }
206
207     return new Pair<LatLonGrid,LatLonGrid>(u,v);
208 }

```

6.78.3.2 static void edu.ou.asgbook.motion.ObjectTracker.main (String[] args) throws Exception [static]

```

210                                     {
211     // create output directory
212     File out = OutputDirectory.getDefault("objecttracker");

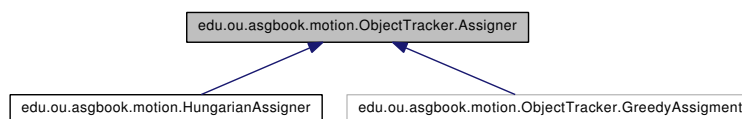
```



```
213
214     // read
215     File f = new File("data/seviri");
216     Pair<LatLonGrid,Date>[] grids = SeviriInfraredTemperature.readAll(f);
217
218     // do alg
219     MotionEstimator alg = new ObjectTracker(100, 110, 1000);
220     MedianFilter smoother = new MedianFilter(10);
221     LatLonGrid grid0 = smoother.filter(grids[0].first);
222     LatLonGrid grid1 = smoother.filter(grids[1].first);
223     Pair<LatLonGrid,LatLonGrid> motion = alg.compute(grid0, grid1, out);
224
225     // write
226     SaturateFilter filter = new SaturateFilter(-150, 150);
227     LatLonGrid u = filter.filter(motion.first);
228     LatLonGrid v = filter.filter(motion.second);
229     KmlWriter.write(u, out, "closest_u", PngWriter.createCoolToWarmColormap());
230     KmlWriter.write(v, out, "closest_v", PngWriter.createCoolToWarmColormap());
231 }
```

6.79 edu.ou.asgbook.motion.ObjectTracker.Assigner Interface Reference

Inheritance diagram for edu.ou.asgbook.motion.ObjectTracker.Assigner:



Public Member Functions

- `int[] getAssignments (int[][] cost, int maxcost)`

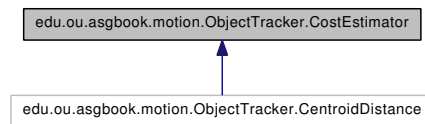
6.79.1 Member Function Documentation

6.79.1.1 `int[] edu.ou.asgbook.motion.ObjectTracker.Assigner.getAssignments (int cost[], int maxcost)`

Implemented in [edu.ou.asgbook.motion.HungarianAssigner](#).

6.80 edu.ou.asgbook.motion.ObjectTracker.CostEstimator Interface Reference

Inheritance diagram for edu.ou.asgbook.motion.ObjectTracker.CostEstimator:



Public Member Functions

- int `computeCost` (`RegionProperty` a, `RegionProperty` b)
- int `getMaxCost` ()

6.80.1 Member Function Documentation

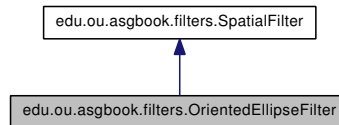
6.80.1.1 int `edu.ou.asgbook.motion.ObjectTracker.CostEstimator.computeCost` (`RegionProperty` a, `RegionProperty` b)

6.80.1.2 int `edu.ou.asgbook.motion.ObjectTracker.CostEstimator.getMaxCost` ()

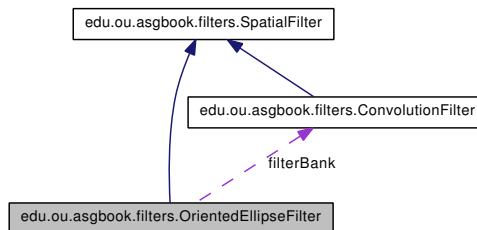
6.81 edu.ou.asgbook.filters.OrientedEllipseFilter Class Reference

A non-isotropic smoothing filter.

Inheritance diagram for edu.ou.asgbook.filters.OrientedEllipseFilter:



Collaboration diagram for edu.ou.asgbook.filters.OrientedEllipseFilter:



Public Member Functions

- [OrientedEllipseFilter](#) (int numFilters, int a, int b)
- [LatLonGrid smooth](#) ([LatLonGrid](#) input, File out)
Finds the maximum response of all the oriented filters.
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.81.1 Detailed Description

A non-isotropic smoothing filter.

Author:

valliappa.lakshmanan

6.81.2 Constructor & Destructor Documentation

6.81.2.1 edu.ou.asgbook.filters.OrientedEllipseFilter.OrientedEllipseFilter (int numFilters, int a, int b)

```

23                                     {
24         if (a == b){
25             throw new IllegalArgumentException("For an ellipse, a != b");
26         }
27         filterBank = new ConvolutionFilter[numFilters];
28         int size = Math.max(a,b) * 2 + 1;
29         for (int f=0; f < numFilters; ++f){
30             double[][] coeffs = new double[size][size];
31             double theta = (f * Math.PI) / numFilters; // 0 to 180
32             for (int i=0; i < size; ++i) for (int j=0; j < size; ++j){
33                 double x = i - a;
34                 double y = j - b;
35                 double term1 = x*Math.cos(theta) - y*Math.sin(theta);
36                 term1 = (term1*term1) / (a*a);
37                 double term2 = x*Math.sin(theta) + y*Math.cos(theta);
38                 term2 = (term2*term2) / (b*b);
39                 if ( (term1+term2) <= 1 ){
40                     coeffs[i][j] = 1;
41                 } // else zero
42             }
43             filterBank[f] = new ConvolutionFilter(coeffs);
44         }
45     }

```

6.81.3 Member Function Documentation

6.81.3.1 Override [LatLonGrid](#) edu.ou.asgbook.filters.OrientedEllipseFilter.filter ([LatLonGrid](#) input)

```

72                                     {
73         return smooth(input, null);
74     }

```

6.81.3.2 static void edu.ou.asgbook.filters.OrientedEllipseFilter.main (String[] args) throws Exception [static]

```

76                                     {
77         // create output directory
78         File out = OutputDirectory.getDefault("oriented");
79
80         // read input
81         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulati
82         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
83
84         OrientedEllipseFilter filter = new OrientedEllipseFilter(8, 1, 5);
85         LatLonGrid sm = filter.smooth(popdensity, out);

```

```
86         KmlWriter.write(sm, out, "ellipse", PngWriter.createCoolToWarmColormap());
87     }
```

6.81.3.3 [LatLonGrid](#) edu.ou.asgbook.filters.OrientedEllipseFilter.smooth ([LatLonGrid](#) *input*, *File out*)

Finds the maximum response of all the oriented filters.

```
50                                     {
51     LatLonGrid result = filterBank[0].smooth(input);
52     KmlWriter.debugWrite(result, out, "ellipse0");
53     for (int f=1; f < filterBank.length; ++f){
54         LatLonGrid fth = filterBank[f].smooth(input);
55         KmlWriter.debugWrite(fth, out, "ellipse"+f);
56         int nrows = fth.getNumLat();
57         int ncols = fth.getNumLon();
58         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
59             int maxval = result.getValue(i, j);
60             int fthval = fth.getValue(i, j);
61             if (maxval == input.getMissing() ||
62                 (fthval != input.getMissing() && fthval > maxval) ){
63                 maxval = fthval;
64             }
65             result.setValue(i, j, maxval);
66         }
67     }
68     return result;
69 }
```

6.82 edu.ou.asgbook.histogram.OtsuThresholdSelector Class Reference

Uses Otsu (1979) to select optimal threshold.

Public Member Functions

- [OtsuThresholdSelector](#) ([Histogram](#) hist)
If x is returned, then values $< x$ are one category and values $\geq x$ are another.
- `int` [getOptimalThreshold](#) ()

Static Public Member Functions

- `static void` [main](#) (String[] args) throws Exception

6.82.1 Detailed Description

Uses Otsu (1979) to select optimal threshold.

Author:

v.lakshmanan

6.82.2 Constructor & Destructor Documentation

6.82.2.1 edu.ou.asgbook.histogram.OtsuThresholdSelector.OtsuThresholdSelector ([Histogram](#) hist)

If x is returned, then values $< x$ are one category and values $\geq x$ are another.

```
30                                     {
31         // compute p_i
32         float[] prob = hist.calcProb();
33
34         // mu_T
35         float mu_T = 0;
36         for (int i=0; i < hist.getHist().length; ++i){
37             mu_T += (i+1) * prob[i];
38         }
39
40         // find k*
41         var = new float[hist.getHist().length];
42         int best_k = -1;
```

```

43         float maxvar = 0;
44         float w_k = 0;
45         float mu_k = 0;
46         for (int k=0; k < hist.getHist().length; ++k){
47             w_k += prob[k];
48             mu_k += (k+1) * prob[k];
49             float denom = w_k * (1-w_k);
50             float num = mu_T*w_k - mu_k;
51             if ( denom > 0 ){
52                 var[k] = (num * num) / denom;
53                 // System.out.println(k + " " + var[k]);
54                 if ( var[k] > maxvar ){
55                     maxvar = var[k];
56                     best_k = k;
57                 }
58             }
59         }
60
61         // return min value of (k+1)th bin
62         optimalThreshold = (hist.getMin() + (best_k+1)* hist.getIncr());
63     }

```

6.82.3 Member Function Documentation

6.82.3.1 `int edu.ou.asgbook.histogram.OtsuThresholdSelector.getOptimalThreshold ()`

```

65         {
66         return optimalThreshold;
67     }

```

6.82.3.2 `static void edu.ou.asgbook.histogram.OtsuThresholdSelector.main (String[] args) throws Exception` [static]

```

69         {
70         // create output directory
71         File outdir = OutputDirectory.getDefault("otsu");
72
73         // read input
74         LatLonGrid conus = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
75
76         // find threshold
77         final int MIN = 0;
78         final int MAX = 100;
79         final int incr = 1;
80         Histogram hist = new Histogram(MIN, incr, (MAX-MIN)/incr );
81         hist.update(conus);
82         // System.out.println(hist);
83         OtsuThresholdSelector thresholder = new OtsuThresholdSelector(hist);
84         int thresh = thresholder.optimalThreshold;
85         System.out.println("Optimal threshold=" + thresh);
86     }

```



```
87         // plot histogram and variance
88         PrintWriter writer = new PrintWriter(new FileWriter(outdir + "/var.txt"));
89         for (int i=0; i < hist.getHist().length; ++i){
90             int val = (int) (0.5 + hist.getMin() + (i+0.5)*hist.getIncr());
91             writer.println(val + " " + hist.getHist()[i] + " " + thresholder.var[i]);
92         }
93         writer.close();
94
95         // threshold
96         SimpleThresholder filter = new SimpleThresholder(thresh);
97         LatLonGrid binaryImage = filter.threshold(conus);
98
99         KmlWriter.write(binaryImage, outdir, "highpop", PngWriter.createCoolToWarmColormap());
100     }
```

6.83 edu.ou.asgbook.io.OutputDirectory Class Reference

Change this to change the output directory that is used by all the main().

Static Public Member Functions

- static File [temporary](#) (String prefix) throws IOException
- static File [relative](#) (String prefix) throws IOException
- static File [getDefault](#) (String prefix) throws IOException

Change this to change the output directory that is used by all the main().

6.83.1 Detailed Description

Change this to change the output directory that is used by all the main().

Author:

Valliappa.Lakshmanan

6.83.2 Member Function Documentation

6.83.2.1 static File edu.ou.asgbook.io.OutputDirectory.getDefault (String *prefix*) throws IOException [static]

Change this to change the output directory that is used by all the main().

```

40                                                                 {
41         // return temporary(prefix);
42         return relative(prefix);
43     }
```

6.83.2.2 static File edu.ou.asgbook.io.OutputDirectory.relative (String *prefix*) throws IOException [static]

```

28                                                                 {
29         // current directory
30         File out = new File("output/" + prefix + "_files");
31         out.delete();
32         out.mkdirs();
33         System.out.println("Output will be in " + out);
34         return out;
35     }
```

6.83.2.3 static File edu.ou.asgbook.io.OutputDirectory temporary (String *prefix*) throws IOException [static]

```
17         {
18
19         File out = File.createTempFile(prefix, "_files");
20         out.delete();
21         out.mkdirs();
22         System.out.println("Output will be in " + out);
23         return out;
24
25     }
```

6.84 edu.ou.asgbook.core.Pair< X, Y > Class Reference

An utility class so that methods can return two objects.

Public Member Functions

- [Pair](#) (X a, Y b)

Public Attributes

- final X [first](#)
- final Y [second](#)

6.84.1 Detailed Description

An utility class so that methods can return two objects.

Author:

v.lakshmanan

6.84.2 Constructor & Destructor Documentation

6.84.2.1 edu.ou.asgbook.core.Pair< X, Y >.Pair (X a, Y b)

```
15         {  
16             first = a;  
17             second = b;  
18         }
```

6.84.3 Member Data Documentation

6.84.3.1 final X edu.ou.asgbook.core.Pair< X, Y >.[first](#)

6.84.3.2 final Y edu.ou.asgbook.core.Pair< X, Y >.[second](#)

6.85 edu.ou.asgbook.motion.PhaseCorrelation Class Reference

Estimate motion based on FFT.

Public Member Functions

- [PhaseCorrelation](#) (int maxu, int maxv)
- Pair< Integer, Integer > [compute](#) ([LatLonGrid](#) data0, [LatLonGrid](#) data1)

Static Public Member Functions

- static [Pixel](#) [computeCentroid](#) ([LatLonGrid](#) a)
- static void [test](#) (File out) throws Exception
- static void [main](#) (String[] args) throws Exception

Package Attributes

- final int [MAXU](#)
- final int [MAXV](#)

6.85.1 Detailed Description

Estimate motion based on FFT.

Author:

v.lakshmanan

6.85.2 Constructor & Destructor Documentation

6.85.2.1 edu.ou.asgbook.motion.PhaseCorrelation.PhaseCorrelation (int maxu, int maxv)

```
30                                     {
31         this.MAXU = maxu;
32         this.MAXV = maxv;
33     }
```

6.85.3 Member Function Documentation

6.85.3.1 Pair<Integer,Integer> edu.ou.asgbook.motion.PhaseCorrelation.compute (LatLonGrid data0, LatLonGrid data1)

```

35                                                                                                     {
36         int motNS = 0, motEW = 0;
37         // a
38         Complex[][] in1 = FFT2D.fft(FFT2D.zeropad(data0));
39
40         // zero-out an area of thickness MAXU/MAXV around the boundary to avoid boundary is
41         LatLonGrid centerb = LatLonGrid.copyOf(data1);
42         int minx = MAXU;
43         int miny = MAXV;
44         int maxx = centerb.getNumLat() - minx;
45         int maxy = centerb.getNumLon() - miny;
46         for (int i=0; i < data1.getNumLat(); ++i){
47             for (int j=0; j < data1.getNumLon(); ++j){
48                 if (i < minx || j < miny || i > maxx || j > maxy){
49                     centerb.setValue(i, j, 0);
50                 }
51             }
52         }
53         Complex[][] in2 = FFT2D.fft(FFT2D.zeropad(centerb));
54
55         // find phase shift at this point
56         for (int i=0; i < in1.length; ++i) for (int j=0; j < in1[0].length; ++j){
57             in1[i][j] = in1[i][j].multiply(in2[i][j].conjugate());
58             in1[i][j] = in1[i][j].multiply( 1.0 / in1[i][j].norm() );
59         }
60         // take ifft
61         Complex[][] result = FFT2D.ifft(in1);
62
63         // find location at which the cross-power spectrum is maximum
64         double bestValue = Integer.MIN_VALUE;
65         int startx = 0; // result.length/2 - MAXU;
66         int starty = 0; // result[0].length/2 - MAXV;
67         int endx = result.length; // /2 + MAXU;
68         int endy = result[0].length; // /2 + MAXV;
69         for (int i=startx; i < endx; ++i) for (int j=starty; j < endy; ++j){
70             if ( result[i][j].normsq() > bestValue ){
71                 bestValue = result[i][j].real;
72                 motNS = -i;
73                 motEW = -j;
74             }
75         }
76
77         // we don't want a 345-degree phase shift; we want it to be 15-degrees
78         if ( Math.abs(motNS) > result.length/2 ){
79             if (motNS < 0) motNS += result.length;
80             else motNS -= result.length;
81         }
82         if ( Math.abs(motEW) > result[0].length/2 ){
83             if (motEW < 0) motEW += result[0].length;
84             else motEW -= result[0].length;

```

```

85         }
86
87         return new Pair<Integer,Integer>(motNS, motEW);
88     }

```

6.85.3.2 static [Pixel](#) edu.ou.asgbook.motion.PhaseCorrelation.computeCentroid ([LatLonGrid a](#)) [static]

```

90                                     {
91         double sumx = 0;
92         double sumy = 0;
93         double sumwt = 0;
94         int N = 0;
95         for (int i=0; i < a.getNumLat(); ++i) for (int j=0; j < a.getNumLon(); ++j){
96             double wt = a.getValue(i, j);
97             sumx += i * wt;
98             sumy += j * wt;
99             sumwt += wt;
100            ++N;
101        }
102        return new Pixel((int)Math.round(sumx/sumwt), (int)Math.round(sumy/sumwt), (int)Math.round(sum
103    }

```

6.85.3.3 static void edu.ou.asgbook.motion.PhaseCorrelation.main (String[] *args*) throws Exception [static]

```

134                                     {
135         // create output directory
136         File out = OutputDirectory.getDefault("phasecorr");
137         test(out);
138
139         // read
140         File f = new File("data/seviri");
141         Pair<LatLonGrid,Date>[] grids = SeviriInfraredTemperature.readAll(f);
142
143         // do alg
144         Pair<Integer,Integer> uv = new PhaseCorrelation(5, 5).compute(grids[0].first, grids[1].first);
145         System.out.println("u=" + uv.first + " v=" + uv.second);
146     }

```

6.85.3.4 static void edu.ou.asgbook.motion.PhaseCorrelation.test (File *out*) throws Exception [static]

```

105                                     {
106         // because the alignment doesn't really check lat-lon extents,
107         // cropping from offset corners will look like translation ...
108         LatLonGrid conus = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
109         LatLonGrid[] grids = new LatLonGrid[2];
110         grids[0] = conus.crop(900, 2500, 256, 256);

```

```
111         int motx = 5; int moty = 9;
112         grids[1] = conus.crop(900-motx, 2500-moty, 256, 256);
113
114         // do alg
115         Pair<Integer,Integer> motion = new PhaseCorrelation(30,30).compute(grids[0], grids[1]);
116         System.out.println("Motion N/S =" + motion.first + " true N/S=" + motx);
117         System.out.println("Motion E/W =" + motion.second + " true E/W=" + moty);
118
119         System.out.println("Centroid of first = " + computeCentroid(grids[0]));
120         System.out.println("Centroid of second = " + computeCentroid(grids[1]));
121
122         KmlWriter.write(grids[0], out, "popdensityA", PngWriter.createCoolToWarmColormap());
123         KmlWriter.write(grids[1], out, "popdensityB", PngWriter.createCoolToWarmColormap());
124
125         // based on edges alone
126         SobelEdgeFilter edgeFilter = new SobelEdgeFilter();
127         LatLonGrid edge1 = edgeFilter.edgeFilter(grids[0]);
128         LatLonGrid edge2 = edgeFilter.edgeFilter(grids[1]);
129         motion = new PhaseCorrelation(30,30).compute(edge1, edge2);
130         System.out.println("Motion N/S =" + motion.first + " true N/S=" + motx);
131         System.out.println("Motion E/W =" + motion.second + " true E/W=" + moty);
132     }
```

6.85.4 Member Data Documentation

6.85.4.1 **final int** [edu.ou.asgbook.motion.PhaseCorrelation.MAXU](#)
[package]

6.85.4.2 **final int** [edu.ou.asgbook.motion.PhaseCorrelation.MAXV](#)
[package]

6.86 edu.ou.asgbook.core.Pixel Class Reference

A grid point in a spatial grid consists of a location and value.

Public Member Functions

- int [getX](#) ()
- int [getY](#) ()
- int [getRow](#) ()
- int [getCol](#) ()
- int [getValue](#) ()
- [Pixel](#) (int x, int y, int value)
- Override boolean [equals](#) (Object o)
- int [getDistanceSquared](#) ([Pixel](#) other)
- int [getDistanceSquared](#) (int otherx, int othery)
- Override int [compareTo](#) ([Pixel](#) other)

Compares both location and value.

- Override String [toString](#) ()

Static Public Member Functions

- static void [main](#) (String[] args)

Classes

- class [CompareLocation](#)
- class [CompareValue](#)

6.86.1 Detailed Description

A grid point in a spatial grid consists of a location and value.

Author:

Valliappa.Lakshmanan

6.86.2 Constructor & Destructor Documentation

6.86.2.1 edu.ou.asgbook.core.Pixel.Pixel (int *x*, int *y*, int *value*)

```

37                                     {
38         super();
39         this.x = x;
40         this.y = y;
41         this.value = value;
42     }
```

6.86.3 Member Function Documentation

6.86.3.1 Override int edu.ou.asgbook.core.Pixel.compareTo ([Pixel](#) *other*)

Compares both location and value.

To compare only based on location or based on value

See also:

CompareValue
CompareLocation

```

91                                     {
92         if ( other == null ){
93             return 1;
94         }
95         if ( other.value == value ){
96             if ( other.x == x ){
97                 return (y - other.y);
98             } else {
99                 return (x - other.x);
100            }
101        } else {
102            return value - other.value;
103        }
104    }
```

6.86.3.2 Override boolean edu.ou.asgbook.core.Pixel.equals (Object *o*)

```

45                                     {
46         if (o == this){
47             return true;
48         }
49         if (o == null || !o.getClass().equals(this.getClass())){
50             return false;
51         }
52         Pixel other = (Pixel) o;
53         return (other.x == x && other.y == y && other.value == value);
54     }
```

6.86.3.3 int edu.ou.asgbook.core.Pixel.getCol ()

```
28         {
29     return y;
30 }
```

6.86.3.4 int edu.ou.asgbook.core.Pixel.getDistanceSquared (int *otherx*, int *othery*)

```
60         {
61     int distx = this.getX() - otherx;
62     int disty = this.getY() - othery;
63     return (distx*distx) + (disty*disty);
64 }
```

6.86.3.5 int edu.ou.asgbook.core.Pixel.getDistanceSquared ([Pixel](#) *other*)

```
56         {
57     return getDistanceSquared(other.x, other.y);
58 }
```

6.86.3.6 int edu.ou.asgbook.core.Pixel.getRow ()

```
24         {
25     return x;
26 }
```

6.86.3.7 int edu.ou.asgbook.core.Pixel.getValue ()

```
33         {
34     return value;
35 }
```

6.86.3.8 int edu.ou.asgbook.core.Pixel.getX ()

```
16         {
17     return x;
18 }
```

6.86.3.9 int edu.ou.asgbook.core.Pixel.getY ()

```
20      {
21      return y;
22      }
```

6.86.3.10 static void edu.ou.asgbook.core.Pixel.main (String[] args)
[static]

```
111      {
112      Pixel a = new Pixel(3,4,3);
113      System.out.println("should be +ve: " + new CompareValue().compare(a,new Pixel(-1,-1,1)));
114      System.out.println("should be -ve: " + new CompareValue().compare(a,new Pixel(-1,-1,-1)));
115      System.out.println("should be 0: " + new CompareValue().compare(a,new Pixel(-1,-1,0)));
116
117      System.out.println("should be +ve: " + new CompareLocation().compare(a,new Pixel(2,3,1)));
118      System.out.println("should be -ve: " + new CompareLocation().compare(a,new Pixel(5,4,1)));
119      System.out.println("should be 0: " + new CompareLocation().compare(a,new Pixel(3,4,3)));
120
121      System.out.println("should be +ve: " + a.compareTo(new Pixel(2,3,1)));
122      System.out.println("should be -ve: " + a.compareTo(new Pixel(3,4,11)));
123      System.out.println("should be 0: " + a.compareTo(new Pixel(3,4,3)));
124
125      System.out.println("Dist = " + a.getDistanceSquared(new Pixel(4,5,6)));
126      }
```

6.86.3.11 Override String edu.ou.asgbook.core.Pixel.toString ()

```
107      {
108      return new StringBuilder().append("[ ").append(x).append(", ").append(y).append(" ")
109      }
```

6.87 edu.ou.asgbook.io.PngWriter Class Reference

Writes a spatial grid out as PNG file.

Static Public Member Functions

- static void [writeAutoScaled](#) ([LatLonGrid](#) grid, File outputFile, ColorModel colormap) throws Exception
- static IndexColorModel [createGrayScaleColormap](#) ()
black-to-white colormap
- static IndexColorModel [createHotColormap](#) ()
a colormap that goes from blue to red through purple
- static IndexColorModel [createCoolToWarmColormap](#) ()
a colormap that goes from green to red through white.
- static IndexColorModel [createRandomColormap](#) ()
Randomized colormap, useful for displaying object labels, for example where the data values themselves do not mean anything beyond being a means to distinguish between objects.
- static void [writeScaled](#) ([LatLonGrid](#) grid, File outputFile, int min, int max, ColorModel colormap) throws Exception
- static void [main](#) (String[] args) throws Exception

Static Package Attributes

- static final byte [DEFAULT_TRANSPARENCY](#) = (byte) 200

6.87.1 Detailed Description

Writes a spatial grid out as PNG file.

Author:

Valliappa.Lakshmanan

6.87.2 Member Function Documentation

6.87.2.1 static IndexColorModel edu.ou.asgbook.io.PngWriter.createCoolToWarmColormap () [static]

a colormap that goes from green to red through white.

See Candidate2 in http://www.paraview.org/ParaView3/index.php/Default_Color_Map

Adapted from the work of Cindy Brewer for use in ParaView

```

111                                     {
112         byte[] red = new byte[256];
113         byte[] green = new byte[red.length];
114         byte[] blue = new byte[red.length];
115         byte[] alpha = new byte[red.length];
116
117         interpolate(red, green, blue, 0, 25, 0.0196078, 0.188235, 0.380392, 0.129412, 0.4);
118         interpolate(red, green, blue, 25, 51, 0.129412, 0.4, 0.67451, 0.262745, 0.576471, 0.3);
119         interpolate(red, green, blue, 51, 76, 0.262745, 0.576471, 0.764706, 0.572549, 0.772549, 0.2);
120         interpolate(red, green, blue, 76, 102, 0.572549, 0.772549, 0.870588, 0.819608, 0.819608, 0.1);
121         interpolate(red, green, blue, 102, 127, 0.819608, 0.898039, 0.941176, 0.968627, 0.968627, 0.05);
122         interpolate(red, green, blue, 127, 153, 0.968627, 0.968627, 0.968627, 0.992157, 0.992157, 0.01);
123         interpolate(red, green, blue, 153, 178, 0.992157, 0.858824, 0.780392, 0.956863, 0.956863, 0.005);
124         interpolate(red, green, blue, 178, 204, 0.956863, 0.647059, 0.509804, 0.839216, 0.839216, 0.001);
125         interpolate(red, green, blue, 204, 229, 0.839216, 0.376471, 0.301961, 0.698039, 0.698039, 0.0005);
126         interpolate(red, green, blue, 229, 256, 0.698039, 0.0941176, 0.168627, 0.403922, 0.403922, 0.0001);
127
128         alpha[0] = 0;
129         for (int i=1; i < alpha.length; ++i){
130             alpha[i] = DEFAULT TRANSPARENCY;
131         }
132
133         IndexColorModel colormap = new IndexColorModel(16, red.length, red, green, blue, alpha);
134         return colormap;
135     }

```

6.87.2.2 static IndexColorModel edu.ou.asgbook.io.PngWriter.createGrayScaleColormap () [static]

black-to-white colormap

```

52                                     {
53         byte[] red = new byte[256];
54         byte[] alpha = new byte[red.length];
55         for (int i=0; i < red.length; ++i){
56             red[i] = (byte) i;
57         }
58         alpha[0] = 0;
59         for (int i=1; i < alpha.length; ++i){
60             alpha[i] = DEFAULT TRANSPARENCY;
61         }
62         IndexColorModel colormap = new IndexColorModel(16, red.length, red, red, red, alpha);

```

```
63         return colormap;
64     }
```

6.87.2.3 static IndexColorModel edu.ou.asgbook.io.PngWriter.createHot-Colormap () [static]

a colormap that goes from blue to red through purple

```
69                                     {
70         byte[] red = new byte[256];
71         byte[] green = new byte[red.length];
72         byte[] blue = new byte[red.length];
73         byte[] alpha = new byte[red.length];
74         for (int i=0; i < red.length; ++i){
75             red[i] = (byte) i;
76             blue[i] = (byte) ( 255 - red[i]);
77             green[i] = (byte) ( (red[i] + blue[i])/2 );
78         }
79         alpha[0] = 0;
80         for (int i=1; i < alpha.length; ++i){
81             alpha[i] = DEFAULT_TRANSPARENCY;
82         }
83         IndexColorModel colormap = new IndexColorModel(16, red.length, red, green, blue, alpha);
84         return colormap;
85     }
```

6.87.2.4 static IndexColorModel edu.ou.asgbook.io.PngWriter.createRandom-Colormap () [static]

Randomized colormap, useful for displaying object labels, for example where the datavalues themselves do not mean anything beyond being a means to distinguish between objects.

```
142                                     {
143         byte[] red = new byte[256];
144         byte[] green = new byte[red.length];
145         byte[] blue = new byte[red.length];
146         byte[] alpha = new byte[red.length];
147
148         Random rnd = new Random();
149
150         // random colors for the three channels
151         for (int i=0; i < red.length; ++i){
152             red[i] = (byte) rnd.nextInt(255);
153             green[i] = (byte) rnd.nextInt(255);
154             blue[i] = (byte) rnd.nextInt(255);
155         }
156
157         // 0 is transparent; everything else is opaque
158         alpha[0] = 0;
```

```

159         for (int i=1; i < alpha.length; ++i){
160             alpha[i] = DEFAULT_TRANSPARENCY;
161         }
162
163         IndexColorModel colormap = new IndexColorModel(16, red.length, red, green, blue, a);
164         return colormap;
165     }

```

6.87.2.5 static void edu.ou.asgbook.io.PngWriter.main (String[] args) throws Exception [static]

```

195         {
196             LatLonGrid grid = new LatLonGrid(100, 200, -1, new LatLon(35,-97), 0.1, 0.1);
197             for (int i=0; i < grid.getNumLat(); ++i){
198                 for (int j=0; j < grid.getNumLon(); ++j){
199                     grid.getData()[i][j] = i + j;
200                     if ( i%10 == 0 || j%20 == 0){
201                         grid.getData()[i][j] = grid.getMissing();
202                     }
203                 }
204             }
205
206             File outdir = OutputDirectory.getDefault("pngwriter");
207             File out = new File(outdir.getAbsoluteFile() + "/autoscaled_cooltowarm.png");
208             PngWriter.writeAutoScaled(grid, out, PngWriter.createCoolToWarmColormap());
209
210             out = new File(outdir.getAbsoluteFile() + "/autoscaled_hot.png");
211             PngWriter.writeAutoScaled(grid, out, PngWriter.createHotColormap());
212
213             out = new File(outdir.getAbsoluteFile() + "/autoscaled_gray.png");
214             PngWriter.writeAutoScaled(grid, out, PngWriter.createGrayScaleColormap());
215
216             out = new File(outdir.getAbsoluteFile() + "/scaled_0_10.png");
217             PngWriter.writeScaled(grid, out, 10, 100, PngWriter.createCoolToWarmColormap());
218         }

```

6.87.2.6 static void edu.ou.asgbook.io.PngWriter.writeAutoScaled ([LatLonGrid grid](#), [File outputFile](#), [ColorModel colormap](#)) throws Exception [static]

```

25
26         // find min, max in data
27         int[][] data = grid.getData();
28         int min = Integer.MAX_VALUE;
29         int max = Integer.MIN_VALUE;
30         int numvalid = 0;
31         for (int i=0; i < data.length; ++i){
32             for (int j=0; j < data[0].length; ++j){
33                 if ( data[i][j] != grid.getMissing() ){
34                     ++numvalid;
35                     if ( data[i][j] < min ){

```



```

36             min = data[i][j];
37         }
38         if ( data[i][j] > max ){
39             max = data[i][j];
40         }
41     }
42 }
43 }
44 System.out.println("Autoscaling " + numvalid + " valid pixels between " + min + " and " + max);
45 writeScaled(grid, outputFile, min, max, colormap);
46 }

```

6.87.2.7 static void edu.ou.asgbook.io.PngWriter.writeScaled ([LatLonGrid](#) *grid*, *File* *outputFile*, *int* *min*, *int* *max*, *ColorModel* *colormap*) throws *Exception* [static]

```

167
168     // scale the data and lookup the color
169     int[][] data = grid.getData();
170     double scale = 255.0 / (max - min + 1); // first is for 'missing'
171     BufferedImage result = new BufferedImage(grid.getNumLon(), grid.getNumLat(), BufferedImage.TYPE_INT_RGB);
172     for (int i=0; i < data.length; ++i){
173         for (int j=0; j < data[0].length; ++j){
174             int scaled = 0;
175             if ( data[i][j] == grid.getMissing() ){
176                 scaled = 0;
177             } else if ( data[i][j] < min ){
178                 // System.out.println(data[i][j] + " " + scaled);
179                 scaled = 0;
180             } else if ( data[i][j] >= max ){
181                 scaled = 255;
182             } else {
183                 scaled = (int) ( (data[i][j]-min) * scale + 1.5);
184             }
185             result.setRGB(j, i, colormap.getRGB(scaled));
186         }
187     }
188 }
189
190     // write it out
191     ImageIO.write(result, "png", outputFile);
192     System.out.println("Wrote " + outputFile + " by scaling data between " + min + " and " + max);
193 }

```

6.87.3 Member Data Documentation

6.87.3.1 final byte edu.ou.asgbook.io.PngWriter.DEFAULT_TRANSPARENCY = (byte) 200 [static, package]

6.88 edu.ou.asgbook.core.PointObservations Class Reference

A set of observation points.

Public Member Functions

- [PointObservations](#) (ObservationPoint[] points, int missing)
- int [getMaxValue](#) ()
- ObservationPoint[] [getPoints](#) ()
- int [getMissing](#) ()

Classes

- class **ObservationPoint**
An observation point is a value at a given location.

6.88.1 Detailed Description

A set of observation points.

Author:

valliappa.lakshmanan

6.88.2 Constructor & Destructor Documentation

6.88.2.1 edu.ou.asgbook.core.PointObservations.PointObservations (ObservationPoint[] points, int missing)

```

13                                     {
14         this.points = points;
15         this.missing = missing;
16     }
```

6.88.3 Member Function Documentation

6.88.3.1 int edu.ou.asgbook.core.PointObservations.getMaxValue ()

```

18                                     {
19         int result = missing;
```

```
20         for (int i=0; i < points.length; ++i){
21             if ( result == missing || points[i].getValue() > result ){
22                 result = points[i].getValue();
23             }
24         }
25         return result;
26     }
```

6.88.3.2 int edu.ou.asgbook.core.PointObservations.getMissing ()

```
32         {
33             return missing;
34     }
```

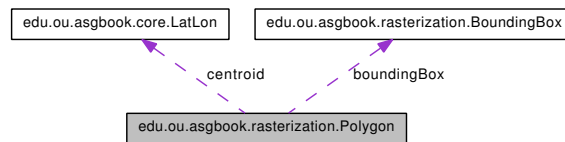
6.88.3.3 ObservationPoint [] edu.ou.asgbook.core.PointObservations.getPoints ()

```
28         {
29             return points;
30     }
```

6.89 edu.ou.asgbook.rasterization.Polygon Class Reference

A polygon consisting of straight edges along the earth's surface.

Collaboration diagram for edu.ou.asgbook.rasterization.Polygon:



Public Member Functions

- [BoundingBox](#) [getBoundingBox](#) ()
- [Polygon](#) ([LatLon](#)[] vertices)
- List< [Line](#) > [getEdges](#) ()
- [LatLon](#) [getCentroid](#) ()
- double [getArea](#) ()

The area is in degrees², so not very useful unless you can convert to km².

- boolean [contains](#) (double x, double y)

Workhorse method: finds out if this point is within this polygon.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.89.1 Detailed Description

A polygon consisting of straight edges along the earth's surface.

Author:

valliappa.lakshmanan

6.89.2 Constructor & Destructor Documentation

6.89.2.1 edu.ou.asgbook.rasterization.Polygon.Polygon ([LatLon\[\]](#) vertices)

```

33         {
34         if (vertices.length < 3){
35             throw new IllegalArgumentException("Need atleast 3 vertices for polygon.");
36         }
37         for (int i=0; i < vertices.length-1; ++i){
38             Line line = new Line( vertices[i].getLat(), vertices[i].getLon(),
39                                 vertices[i+1].getLat(), vertices[i+1].getLon() );
40             edges.add(line);
41         }
42
43         // connect start and end point
44         LatLon last = vertices[vertices.length-1];
45         LatLon first = vertices[0];
46         edges.add(new Line(last.getLat(), last.getLon(), first.getLat(), first.getLon()));
47         computeAreaAndCentroid();
48         boundingBox = new BoundingBox(vertices);
49     }

```

6.89.3 Member Function Documentation

6.89.3.1 boolean edu.ou.asgbook.rasterization.Polygon.contains (double x, double y)

Workhorse method: finds out if this point is within this polygon.

```

99         {
100         // as an optimization, check the bounding box first
101         if (!boundingBox.contains(x,y)){
102             return false;
103         }
104
105         int num_xcrossing = 0;
106         int num_ycrossing = 0;
107         for (int i = 0; i < edges.size(); ++i) {
108             Double x_intercept = edges.get(i).getXIntercept(y);
109             Double y_intercept = edges.get(i).getYIntercept(x);
110             if (y_intercept != null) {
111                 if (y_intercept >= y) {
112                     ++num_ycrossing;
113                 }
114             }
115             if (x_intercept != null) {
116                 if (x_intercept >= x) {
117                     ++num_xcrossing;
118                 }
119             }
120         }
121         // odd number of crossings means inside
122         return ((num_xcrossing % 2 == 1) && (num_ycrossing % 2 == 1));

```

```
123    }
```

6.89.3.2 `double edu.ou.asgbook.rasterization.Polygon.getArea ()`

The area is in degrees², so not very useful unless you can convert to km².

```
63    {
64        return area;
65    }
```

6.89.3.3 `BoundingBox edu.ou.asgbook.rasterization.Polygon.getBoundingBox ()`

```
29    {
30        return boundingBox;
31    }
```

6.89.3.4 `LatLon edu.ou.asgbook.rasterization.Polygon.getCentroid ()`

```
55    {
56        return centroid;
57    }
```

6.89.3.5 `List<Line> edu.ou.asgbook.rasterization.Polygon.getEdges ()`

```
51    {
52        return edges;
53    }
```

6.89.3.6 `static void edu.ou.asgbook.rasterization.Polygon.main (String[] args) throws Exception [static]`

```
126    {
127        File out = OutputDirectory.getDefault("rasterpolygon");
128
129        // made up
130        LatLonGrid grid = new LatLonGrid(500, 500, 0, new LatLon(20, -10), 0.01, 0.01);
131        LatLon[] vertices = new LatLon[]{
132            new LatLon(19,-7),
133            new LatLon(17.5,-6),
134            new LatLon(16.5,-6.8),
135            new LatLon(17.2,-8.5),
136            new LatLon(16,-9.5),
```

```
137         new LatLon(17,-9)
138     };
139     Polygon poly = new Polygon(vertices);
140
141     // draw edges
142     final int EDGE = 10;
143     for (Line line : poly.getEdges()){
144         for (Pixel p : line.getPositionIn(grid)){
145             grid.setValue(p.getRow(), p.getCol(), EDGE);
146         }
147     }
148     KmlWriter.write(grid, out, "edges", PngWriter.createCoolToWarmColormap());
149
150
151     // fill points inside
152     final int POLY = 5;
153     int npix = 0;
154     for (int i=0; i < grid.getNumLat(); ++i){
155         for (int j=0; j < grid.getNumLon(); ++j){
156             LatLon loc = grid.getLocation(i, j);
157             if ( poly.contains(loc.getLat(), loc.getLon())){
158                 grid.setValue(i, j, POLY);
159                 ++npix;
160             }
161         }
162     }
163     KmlWriter.write(grid, out, "polygon", PngWriter.createCoolToWarmColormap());
164
165     System.out.println("Area of polygon: " + poly.getArea() + " num-pixels colored=" + npix);
166     System.out.println("Centroid of polygon: " + poly.getCentroid());
167 }
```

6.90 edu.ou.asgbook.datamining.PrimaryCities Class Reference

Identifies the primary cities in each country.

Static Public Member Functions

- static [LabelResult findPrimaryCities](#) ([LatLonGrid](#) population, [LatLonGrid](#) countries, File out)
- static void [main](#) (String[] args) throws Exception

6.90.1 Detailed Description

Identifies the primary cities in each country.

Author:

valliappa.lakshmanan

6.90.2 Member Function Documentation

6.90.2.1 static [LabelResult](#) edu.ou.asgbook.datamining.PrimaryCities.findPrimaryCities ([LatLonGrid](#) population, [LatLonGrid](#) countries, File out) [static]

```

31
32      // find cities from population data using watershed
33      write(out, population, "pop", PngWriter.createCoolToWarmColormap());
34      EnhancedWatershedSegmenter seg = new EnhancedWatershedSegmenter(10, 1, 600, 10, 5);
35      LabelResult label = seg.label(population);
36      RegionProperty[] popProps = RegionProperty.compute(label, population);
37      write(out, label.label, "allcities", PngWriter.createRandomColormap());
38
39      // initialize primary-cities
40      int ncountries = 1 + new MaxValueFilter().findHighestValued(countries).value;
41      int[] primaryCity = new int[ncountries]; // one for each country
42      for (int i=0; i < ncountries; ++i){
43          primaryCity[i] = -1; // none
44      }
45
46      // go through the cities and assign them to their appropriate country
47      for (int i=1; i < popProps.length; ++i){
48          LatLon centroid = population.getLocation(popProps[i].getCx(), popProps[i].getCy());
49          int country = countries.getValue(centroid);
50          if (country >= 0){
51              if (primaryCity[country] < 0){
52                  primaryCity[country] = i; // first city in country

```



```
53         } else {
54             // the primary city is the one with the greater avg population
55             int previous = primaryCity[country];
56             if (popProps[i].getCval() > popProps[previous].getCval()){
57                 primaryCity[country] = i;
58             }
59         }
60     }
61 }
62
63 // keep only those cities that are primary
64 boolean[] keep = new boolean[popProps.length];
65 for (int i=0; i < ncountries; ++i){
66     if (primaryCity[i] >= 0){
67         int regno = primaryCity[i];
68         keep[regno] = true;
69     }
70 }
71 return RegionProperty.prune(label, keep);
72 }
```

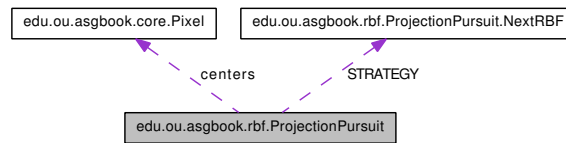
6.90.2.2 static void edu.ou.asgbook.datamining.PrimaryCities.main (String[] args) throws Exception [static]

```
84                                     {
85     // create output directory
86     File out = OutputDirectory.getDefault("primary");
87
88     // read input (crop to cover Spain)
89     LatLonGrid pop = GlobalPopulation.read(GlobalPopulation.WORLD).crop(980, 4080, 220, 350);
90     LatLonGrid countries = CountryPolygons.readGrid(CountryPolygons.WORLD_GRID);
91     LabelResult primary = PrimaryCities.findPrimaryCities(pop, countries, out);
92
93     KmlWriter.write(primary.label, out, "primarycities", PngWriter.createRandomColormap());
94 }
```

6.91 edu.ou.asgbook.rbf.ProjectionPursuit Class Reference

Approximates a spatial grid by a RBF when nothing is known beyond the number of Gaussians desired.

Collaboration diagram for edu.ou.asgbook.rbf.ProjectionPursuit:



Public Member Functions

- [ProjectionPursuit](#) ([LatLonGrid](#) orig, int max_tot_abs_error, int max_number_rbf, File outDir)
- [Pixel](#)[] [getCenters](#) ()
- double[] [getSigmax](#) ()
- double[] [getSigmay](#) ()
- String [toString](#) ()

Static Public Member Functions

- static void [runOnSimulatedInput](#) () throws Exception
- static void [runOnPopDensity](#) (boolean crop) throws Exception
- static void [main](#) (String[] args) throws Exception

Static Public Attributes

- static final [NextRBF](#) [STRATEGY](#) = new LocalMax()

Classes

- class [LocalMax](#)
- interface [NextRBF](#)
- class [SpatialMean](#)

6.91.1 Detailed Description

Approximates a spatial grid by a RBF when nothing is known beyond the number of Gaussians desired.

Author:

v.lakshmanan

6.91.2 Constructor & Destructor Documentation

6.91.2.1 edu.ou.asgbook.rbf.ProjectionPursuit.ProjectionPursuit ([LatLonGrid orig](#), int *max_tot_abs_error*, int *max_number_rbfs*, File *outDir*)

```
34
35     this.MAX_TOT_ABS_ERROR = max_tot_abs_error;
36     this.MAX_NUMBER_RBFS = max_number_rbfs;
37     if ( MAX_NUMBER_RBFS < 10 ){
38         outputInterval = 1;
39     } else if ( MAX_NUMBER_RBFS < 50 ){
40         outputInterval = 5;
41     } else {
42         outputInterval = 10;
43     }
44     this.outDir = outDir;
45     fit(orig, STRATEGY);
46 }
```

6.91.3 Member Function Documentation

6.91.3.1 [Pixel](#) [] edu.ou.asgbook.rbf.ProjectionPursuit.getCenters ()

```
48                                     {
49     return centers;
50 }
```

6.91.3.2 double [] edu.ou.asgbook.rbf.ProjectionPursuit.getSigmax ()

```
52                                     {
53     return sigmax;
54 }
```

6.91.3.3 double [] edu.ou.asgbook.rbf.ProjectionPursuit.getSigmay ()

```
56                                     {
57     return sigmay;
58 }
```

6.91.3.4 static void edu.ou.asgbook.rbf.ProjectionPursuit.main (String[] args) throws Exception [static]

```

257                                     {
258         // runOnSimulatedInput();
259         runOnPopDensity(false); // run with -Xmx1024m otherwise, you'll get out-of-memory
260     }

```

6.91.3.5 static void edu.ou.asgbook.rbf.ProjectionPursuit.runOnPopDensity (boolean crop) throws Exception [static]

```

227                                     {
228         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new G
229         if (crop){
230             popdensity = popdensity.crop(900, 2500, 200, 200);
231         } else {
232             LatLon nwCorner = new LatLon(60, -130);
233             LatLon seCorner = new LatLon(12, -52);
234             popdensity = popdensity.crop(popdensity.getRow(nwCorner),
235                 popdensity.getCol(nwCorner),
236                 popdensity.getRow(seCorner) - popdensity.getRow(nwCorner),
237                 popdensity.getCol(seCorner) - popdensity.getCol(nwCorner));
238         }
239
240         File out = OutputDirectory.getDefault("rbfpopdensity");
241
242         ProjectionPursuit fit = new ProjectionPursuit(popdensity, 1000, 9, out);
243         List<LatLon> locs = new ArrayList<LatLon>();
244         List<String> names = new ArrayList<String>();
245         for (int i=0; i < fit.getCenters().length; ++i){
246             LatLon loc = popdensity.getLocation( fit.getCenters()[i].getX(), fit.getCenters
247             String name = ("RBF#" + i + " ampl=" + fit.getCenters()[i].getValue() + " sign
248             System.out.println(" loc: " + loc + name);
249             if (fit.getCenters()[i].getValue() > 0){
250                 locs.add(loc);
251                 names.add(name);
252             }
253         }
254         KmlWriter.write(locs, names, out, "rbfcities");
255     }

```

6.91.3.6 static void edu.ou.asgbook.rbf.ProjectionPursuit.runOnSimulated-Input () throws Exception [static]

```

209                                     {
210         int nrows = 100;
211         int ncols = 100;
212         Pixel[] centers = new Pixel[]{ new Pixel(nrows/4,ncols/3,20), new Pixel(nrows/3,ncols/3,20) };
213         double[] sigmax = new double[] { nrows/12, ncols/8 };
214         double[] sigmay = new double[] { nrows/8, ncols/12 };
215         LatLonGrid m = DataSimulator.simulateData(centers, sigmax, sigmay, nrows, ncols);

```

```
216         System.out.println("Created data of size " + m.getNumLat() + "x" + m.getNumLon());
217         for (int i=0; i < centers.length; ++i){
218             System.out.println("true RBF#" + i + " center: " + centers[i] + " sigmax=" + sigmax[i] + "
219         }
220
221         File out = OutputDirectory.getDefault("rbf");
222
223         ProjectionPursuit fit = new ProjectionPursuit(m, 100, 4, out);
224         System.out.println(fit);
225     }
```

6.91.3.7 String edu.ou.asgbook.rbf.ProjectionPursuit.toString ()

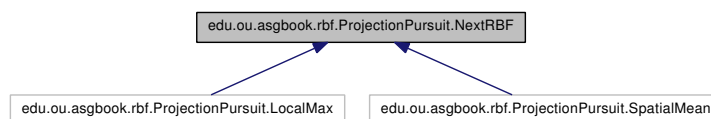
```
65         {
66             StringBuilder sb = new StringBuilder();
67             for (int i=0; i < centers.length; ++i){
68                 sb.append("RBF#" + i + " center: " + centers[i] + " sigmax=" + sigmax[i] + " sigmay=" + sig
69             }
70             return sb.toString();
71         }
```

6.91.4 Member Data Documentation

6.91.4.1 final [NextRBF](#) [edu.ou.asgbook.rbf.ProjectionPursuit.STRATEGY](#) =
new LocalMax() [static]

6.92 edu.ou.asgbook.rbf.ProjectionPursuit.NextRBF Interface Reference

Inheritance diagram for edu.ou.asgbook.rbf.ProjectionPursuit.NextRBF:



Public Member Functions

- double[] [getNewCenterAndSigmas](#) ([LatLonGrid](#) error)

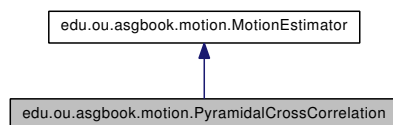
6.92.1 Member Function Documentation

- 6.92.1.1** double [] edu.ou.asgbook.rbf.ProjectionPursuit.NextRBF.getNewCenterAndSigmas ([LatLonGrid](#) *error*)

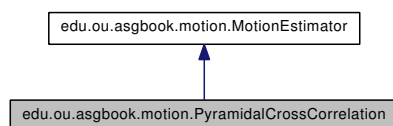
6.93 edu.ou.asgbook.motion.PyramidalCrossCorrelation Class Reference

Cross-correlation at muliple resolutions.

Inheritance diagram for edu.ou.asgbook.motion.PyramidalCrossCorrelation:



Collaboration diagram for edu.ou.asgbook.motion.PyramidalCrossCorrelation:



Public Member Functions

- [PyramidalCrossCorrelation](#) (int maxmotion_x, int maxmotion_y)
Pass in the maximum movement in the two directions.
- Override Pair< [LatLonGrid](#), [LatLonGrid](#) > [compute](#) ([LatLonGrid](#) data0, [LatLonGrid](#) data1, File outdir)
returns motion in the two directions.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.93.1 Detailed Description

Cross-correlation at muliple resolutions.

Author:

v.lakshmanan

6.93.2 Constructor & Destructor Documentation

6.93.2.1 `edu.ou.asgbook.motion.PyramidalCrossCorrelation.PyramidalCrossCorrelation(int maxmotion_x, int maxmotion_y)`

Pass in the maximum movement in the two directions.

```

30                                     {
31             MAX_U = maxmotion_x;
32             MAX_V = maxmotion_y;
33     }
```

6.93.3 Member Function Documentation

6.93.3.1 Override Pair<[LatLonGrid](#), [LatLonGrid](#)> `edu.ou.asgbook.motion.PyramidalCrossCorrelation.compute(LatLonGrid data0, LatLonGrid data1, File outdir)`

returns motion in the two directions.

The first one is north to south and the second one is east to west. The data is aligned to second time frame. The output dir is used for intermediate products and may be null.

Implements [edu.ou.asgbook.motion.MotionEstimator](#).

```

43
44     int numres = (int) Math.round( Math.log( Math.max(MAX_U,MAX_V))/Math.log(2) );
45     System.out.println("Will do computation at " + numres + " resolutions");
46
47     // add missing data to the borders to get them to be divisible by factor
48     int factor = (int) ( Math.pow(2, numres) + 0.01 );
49     data0 = pad( data0, nextMultiple(data0.getNumLat(),factor), nextMultiple(data0.getNumLat(),factor) );
50     data1 = pad( data1, nextMultiple(data0.getNumLat(),factor), nextMultiple(data0.getNumLat(),factor) );
51
52     LatLonGrid aligned_data0 = data0;
53     LatLonGrid u = null;
54     LatLonGrid v = null;
55     for (int res = numres; res >= 0; --res){
56         // shift data0 using u,v
57         if (u != null){
58             aligned_data0 = align(data0, u, v);
59             if (outdir != null){
60                 try {
61                     KmlWriter.write(aligned_data0, outdir, "pxaligned_" + res, PngWriter);
62                 } catch (Exception e) {
63                     e.printStackTrace();
64                 }
65             }
66         }
67
68         // downsample grids to this resolution
69         int smsize = (int) ( Math.pow(2, res) + 0.01 ); // at res=4, this is 2^4 or 16
```



```

70         LatLonGrid grid0 = decreaseSize(aligned_data0, smsize);
71         LatLonGrid grid1 = decreaseSize(data1, smsize);
72         if (outdir != null){
73             try {
74                 KmlWriter.write(grid0, outdir, "pxdata0_" + res, PngWriter.createCoolToWarmColormap());
75                 KmlWriter.write(grid1, outdir, "pxdata1_" + res, PngWriter.createCoolToWarmColormap());
76             } catch (Exception e) {
77                 e.printStackTrace();
78             }
79         }
80
81         // find u,v at this resolution
82         int est_size = numres-res; // 0 at coarsest resolution, +1 with each resolution
83         int search_radius = 1;
84         CrossCorrelation xcorr = new CrossCorrelation(est_size, est_size, search_radius, search_radius);
85         Pair<LatLonGrid,LatLonGrid> motion = xcorr.compute(grid0, grid1, outdir);
86         LatLonGrid thisu = increaseSize(motion.first, smsize);
87         LatLonGrid thisv = increaseSize(motion.second, smsize);
88         thisu = multiply(thisu, smsize); // movement of 1 pixel at res=4 is equal to movement of 16 pixels at res=1
89         thisv = multiply(thisv, smsize);
90         // update the total u,v
91         u = (u != null)? LatLonGrid.add(thisu, u) : thisu;
92         v = (v != null)? LatLonGrid.add(thisv, v) : thisv;
93         if (outdir != null){
94             try {
95                 KmlWriter.write(u, outdir, "pxu_" + res, PngWriter.createCoolToWarmColormap());
96                 KmlWriter.write(v, outdir, "pxv_" + res, PngWriter.createCoolToWarmColormap());
97             } catch (Exception e) {
98                 e.printStackTrace();
99             }
100         }
101     }
102     return new Pair<LatLonGrid,LatLonGrid>(u,v);
103 }

```

6.93.3.2 static void edu.ou.asgbook.motion.PyramidalCrossCorrelation.main (String[] args) throws Exception [static]

```

170                                     {
171         // create output directory
172         File out = OutputDirectory.getDefault("pyramidxcorr");
173
174         // read
175         File f = new File("data/seviri");
176         Pair<LatLonGrid,Date>[] grids = SeviriInfraredTemperature.readAll(f);
177
178         // do alg
179         PyramidalCrossCorrelation alg = new PyramidalCrossCorrelation(20,20);
180         Pair<LatLonGrid,LatLonGrid> motion = alg.compute(grids[0].first, grids[1].first, out);
181
182         // write
183         KmlWriter.write(motion.first, out, "pxfinal_u", PngWriter.createCoolToWarmColormap());
184         KmlWriter.write(motion.second, out, "pxfinal_v", PngWriter.createCoolToWarmColormap());
185     }

```

6.94 edu.ou.asgbook.imgstat.Quantizer Class Reference

Develops a quantization scheme using histogram equalization.

Public Member Functions

- [Quantizer](#) ([Histogram](#) hist, int K)
Pass in a high-resolution histogram i.e.
- int [getBinNumber](#) (int val)
- int [getCenterValue](#) (int bin_no)
- [LatLonGrid](#) band ([LatLonGrid](#) data)
replaces each pixel by the center of its bin
- Override String [toString](#) ()

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.94.1 Detailed Description

Develops a quantization scheme using histogram equalization.

Author:

valliappa.lakshmanan

6.94.2 Constructor & Destructor Documentation

6.94.2.1 edu.ou.asgbook.imgstat.Quantizer.Quantizer ([Histogram](#) hist, int K)

Pass in a high-resolution histogram i.e.

with incr=1, for example

Parameters:

hist

K number of levels

```

30                                     {
31     this.min = hist.getMin();
32     int incr = hist.getIncr();
33     int[] freq = hist.getHist();
34     int N = 0; // number of samples
35     for (int i=0; i < freq.length; ++i){
36         N += freq[i];
37     }
38     double N_per_level = N/(double)K;
39
40     // populate
41     upperBound = new int[K];
42     int level_no=0;
43     int at_this_level = 0;
44     for (int bin_no=0; bin_no < freq.length; ++bin_no){
45         if (at_this_level < N_per_level){
46             at_this_level += freq[bin_no]; // on to next
47         } else {
48             upperBound[level_no] = min + (bin_no * incr);
49             // next level
50             ++level_no;
51             at_this_level = freq[bin_no];
52         }
53     }
54     for (; level_no < K; ++level_no){
55         upperBound[level_no] = min + freq.length * incr;
56     }
57     System.out.println(this);
58 }

```

6.94.3 Member Function Documentation

6.94.3.1 [LatLonGrid](#) edu.ou.asgbook.imgstat.Quantizer.band ([LatLonGrid](#) data)

replaces each pixel by the center of its bin

```

76                                     {
77     LatLonGrid result = LatLonGrid.copyOf(data);
78     int nrows = result.getNumLat();
79     int ncols = result.getNumLon();
80     for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
81         int bin_no = getBinNumber(data.getValue(i,j));
82         int cval = (bin_no < 0)? data.getMissing() : getCenterValue(bin_no);
83         result.setValue(i,j, cval);
84     }
85     return result;
86 }

```

6.94.3.2 int edu.ou.asgbook.imgstat.Quantizer.getBinNumber (int val)

```

60                                     {

```

```

61         for (int i=0; i < upperBound.length; ++i){
62             if (val < upperBound[i]){
63                 return i;
64             }
65         }
66         return -1;
67     }

```

6.94.3.3 `int edu.ou.asgbook.imgstat.Quantizer.getCenterValue (int bin_no)`

```

69         {
70             int lb = (bin_no > 0)? upperBound[bin_no-1] : this.min;
71             int ub = upperBound[bin_no];
72             return (ub+lb)/2;
73         }

```

6.94.3.4 `static void edu.ou.asgbook.imgstat.Quantizer.main (String[] args)` `throws Exception` [static]

```

99         {
100             // create output directory
101             File outdir = OutputDirectory.getDefault("quantizer");
102
103             // read input
104             LatLonGrid conus = SurfaceAlbedo.read(SurfaceAlbedo.CONUS, 100);
105
106             // find histogram in three different ways
107             Histogram full = HistogramBinSelection.createHighestResolution(conus);
108             for (int k=4; k < 32; k *= 2){ // 4, 8, 16
109                 Quantizer quant = new Quantizer(full, k);
110                 LatLonGrid banded = quant.band(conus);
111                 KmlWriter.write(banded, outdir, "quant_" + k, PngWriter.createCoolToWarmColormap);
112
113                 int incr = (int) Math.round( full.getIncr() * full.getHist().length / (double)
114                 Histogram eq = new Histogram(full.getMin(),incr, k);
115                 banded = HistogramBinSelection.band(conus, eq);
116                 KmlWriter.write(banded, outdir, "hist_" + k, PngWriter.createCoolToWarmColormap);
117             }
118         }

```

6.94.3.5 `Override String edu.ou.asgbook.imgstat.Quantizer.toString ()`

```

89         {
90             StringBuilder sb = new StringBuilder();
91             sb.append("Quantizer levels: ");
92             for (int i=0; i < upperBound.length; ++i){
93                 sb.append(upperBound[i]);
94                 sb.append(" ");
95             }

```

```
96         return sb.toString();
97     }
```

6.95 edu.ou.asgbook.filters.QuickSelect Class Reference

From Numerical Recipes, a fast way to find the kth smallest item in a list Useful to implement rank filters.

Static Public Member Functions

- static int [kth_element](#) (int[] arr, int k)
- static int [kth_element](#) (int[] arr, int n, int k)

Finds the kth smallest item in the list.

6.95.1 Detailed Description

From Numerical Recipes, a fast way to find the kth smallest item in a list Useful to implement rank filters.

Author:

v.lakshmanan

6.95.2 Member Function Documentation

6.95.2.1 static int edu.ou.asgbook.filters.QuickSelect.kth_element (int[] arr, int n, int k) [static]

Finds the kth smallest item in the list.

Parameters:

arr list

n number of elements in list, in case the last elements of list are unfilled

k finds kth smallest

Returns:

```

32                                     {
33         if (k > n){
34             throw new IllegalArgumentException("k should be less than n!");
35         }
36         int i, ir, j, low, mid;
```

```

37         int a;
38
39         low = 0;
40         ir = n - 1;
41         for (;;) {
42             if (ir <= low + 1) {
43                 if (ir == low + 1 && arr[ir] < arr[low]) {
44                     SWAP(arr, low, ir);
45                 }
46                 return arr[k];
47             } else {
48                 mid = (low + ir) >> 1;
49                 SWAP(arr, mid, low + 1);
50                 if (arr[low] > arr[ir]) {
51                     SWAP(arr, low, ir);
52                 }
53                 if (arr[low + 1] > arr[ir]) {
54                     SWAP(arr, low + 1, ir);
55                 }
56                 if (arr[low] > arr[low + 1]) {
57                     SWAP(arr, low, low + 1);
58                 }
59                 i = low + 1;
60                 j = ir;
61                 a = arr[low + 1];
62                 for (;;) {
63                     do
64                         i++;
65                     while (arr[i] < a);
66                     do
67                         j--;
68                     while (arr[j] > a);
69                     if (j < i)
70                         break;
71                     SWAP(arr, i, j);
72                 }
73                 arr[low + 1] = arr[j];
74                 arr[j] = a;
75                 if (j >= k)
76                     ir = j - 1;
77                 if (j <= k)
78                     low = i;
79             }
80         }
81     }

```

6.95.2.2 static int edu.ou.asgbook.filters.QuickSelect.kth_element (int[] arr, int k) [static]

```

21         {
22             return kth_element(arr, arr.length, k);
23         }

```

6.96 edu.ou.asgbook.rbf.RadialBasisFunction Class Reference

Finds best fit of a spatial grid to a sum of Gaussians when the centers and sigmas of the Gaussians are known.

Static Public Member Functions

- static double[] [fit](#) ([LatLonGrid](#) data, [Pixel](#)[] center, double[] sigmax, double[] sigmay)

Provide the center locations and this function will fill in the optimal amplitude.

- static void [main](#) (String[] args)

6.96.1 Detailed Description

Finds best fit of a spatial grid to a sum of Gaussians when the centers and sigmas of the Gaussians are known.

Author:

v.lakshmanan

6.96.2 Member Function Documentation

6.96.2.1 static double [] edu.ou.asgbook.rbf.RadialBasisFunction.fit ([LatLonGrid](#) data, [Pixel](#)[] center, double[] sigmax, double[] sigmay) [static]

Provide the center locations and this function will fill in the optimal amplitude.

```

19
20         // inv( transpose(H) * H) * transpose(H) * data
21         int p = data.getNumLat() * data.getNumLon();
22         int m = center.length;
23         Matrix H = new Matrix(p, m);
24         Matrix ycap = new Matrix(p, 1);
25         for (int i=0; i < p; ++i){
26             int x = i / data.getNumLon();
27             int y = i % data.getNumLon();
28             for (int j=0; j < m; ++j){
29                 double xdist = x - center[j].getX();
30                 double ydist = y - center[j].getY();
31                 double xnorm = (xdist*xdist) / (sigmax[j] * sigmax[j]);
32                 double ynorm = (ydist*ydist) / (sigmay[j] * sigmay[j]);

```



```

33         double wt = Math.exp(-(xnorm + ynorm));
34         H.set(i,j, wt);
35     }
36     ycap.set(i, 0, data.getValue(x, y));
37 }
38 // H.print(H.getColumnDimension(), H.getRowDimension());
39
40 Matrix HT = H.transpose();
41 Matrix HTH = HT.times(H);
42 Matrix HTHinv = HTH.inverse();
43 Matrix HTHinvHT = HTHinv.times(HT);
44
45 return HTHinvHT.times(ycap).transpose().toArray()[0];
46 }

```

6.96.2.2 static void edu.ou.asgbook.rbf.RadialBasisFunction.main (String[] args) [static]

```

48                                     {
49     int nrows = 100;
50     int ncols = 100;
51     Pixel[] centers = new Pixel[]{ new Pixel(nrows/4,ncols/3,20), new Pixel(nrows/3,ncols/2,10) };
52     double[] sigmax = new double[] { nrows/12, ncols/8 };
53     double[] sigmay = new double[] { nrows/8, ncols/12 };
54     LatLonGrid m = DataSimulator.simulateData(centers, sigmax, sigmay, nrows, ncols);
55
56     System.out.println("Created data of size " + m.getNumLat() + "x" + m.getNumLon());
57     double[] weights = fit( m, centers, sigmax, sigmay );
58     for (int i=0; i < weights.length; ++i){
59         System.out.println("Actual: " + centers[i].getValue() + " RBF: " + +weights[i]);
60     }
61 }

```

6.97 edu.ou.asgbook.segmentation.RegionGrowing Class Reference

Common object-identification utility.

Static Public Member Functions

- static void [growRegion](#) (int x, int y, [LatLonGrid](#) data, int thresh, [LatLonGrid](#) label, int currLabel)

6.97.1 Detailed Description

Common object-identification utility.

Author:

v.lakshmanan

6.97.2 Member Function Documentation

6.97.2.1 static void edu.ou.asgbook.segmentation.RegionGrowing.growRegion (int x, int y, [LatLonGrid](#) data, int thresh, [LatLonGrid](#) label, int currLabel) [static]

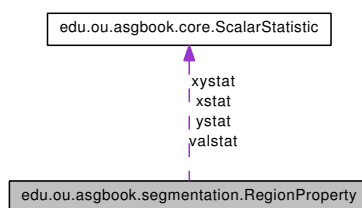
```

19
20     final int junk = 0; // data value not needed for region growing
21     final int UNSET = 0;
22     List<Pixel> stack = new ArrayList<Pixel>();
23     stack.add(new Pixel(x,y,junk));
24     while (stack.size() > 0){
25         Pixel p = stack.remove(stack.size()-1);
26         label.setValue(p.getX(), p.getY(), currLabel);
27         for (int i=p.getX()-1; i <= p.getX()+1; ++i){
28             for (int j=p.getY()-1; j <= p.getY()+1; ++j){
29                 if (data.isValid(i, j) && data.getValue(i,j) > thresh && label.getValue
30                     stack.add(new Pixel(i,j,junk));
31             }
32         }
33     }
34 }
35 
```

6.98 edu.ou.asgbook.segmentation.RegionProperty Class Reference

Properties of a region such as geometric (centroid, area, etc) and physical (based on other grid values).

Collaboration diagram for edu.ou.asgbook.segmentation.RegionProperty:



Public Member Functions

- double [getCx](#) ()
- double [getCy](#) ()
- double [getCval](#) ()
- int [getSize](#) ()
- Ellipse [getEllipseFit](#) ()

Static Public Member Functions

- static [RegionProperty](#)[] [compute](#) ([LabelResult](#) label, [LatLonGrid](#) data)
- static [Pixel](#)[][] [getPixelsInRegions](#) ([LatLonGrid](#) data1, [LabelResult](#) objects1)
All the pixels for each region.
- static [LabelResult](#) [prune](#) ([LabelResult](#) input, boolean[] keep)
Regions for which keep=false are removed.
- static [LabelResult](#) [pruneBySize](#) ([LabelResult](#) input, [LatLonGrid](#) grid, int sizethresh)
Regions smaller than sizethresh are removed.
- static void [main](#) (String[] args) throws Exception

Classes

- class [Ellipse](#)

6.98.1 Detailed Description

Properties of a region such as geometric (centroid, area, etc) and physical (based on other grid values).

Author:

v.lakshmanan

6.98.2 Member Function Documentation

6.98.2.1 static [RegionProperty](#) [] edu.ou.asgbook.segmentation.RegionProperty.compute ([LabelResult](#) label, [LatLonGrid](#) data)

[static]

```

113                                     {
114     RegionProperty[] props = new RegionProperty[label.maxlabel+1];
115     for (int i=1; i < props.length; ++i){
116         props[i] = new RegionProperty();
117     }
118     int nrows = label.label.getNumLat();
119     int ncols = label.label.getNumLon();
120     for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
121         if (label.label.getValue(i,j) > 0){
122             props[ label.label.getValue(i,j) ].update(i, j, data.getValue(i,j));
123         }
124     }
125     return props;
126 }
```

6.98.2.2 double edu.ou.asgbook.segmentation.RegionProperty.getCval ()

```

46                                     {
47     return valstat.getMean();
48 }
```

6.98.2.3 double edu.ou.asgbook.segmentation.RegionProperty.getCx ()

```

40                                     {
41     return xstat.getMean();
42 }
```

6.98.2.4 double edu.ou.asgbook.segmentation.RegionProperty.getCy ()

```

43                                     {
44     return ystat.getMean();
45 }
```

6.98.2.5 Ellipse edu.ou.asgbook.segmentation.RegionProperty.getEllipseFit ()

```

86         {
87             final double cx = xstat.getMean();
88             final double cy = ystat.getMean();
89             final double s11 = xstat.getVariance();
90             final double s22 = ystat.getVariance();
91             final double s12 = xystat.getMean() - cx*cy;
92             double tmp = (s11 - s22) * (s11 - s22) + 4 * s12 * s12;
93             if (tmp >= 0.00001) {
94                 tmp = Math.sqrt(tmp);
95             } else {
96                 tmp = 0;
97             }
98             double eigen1 = (s11 + s22 + tmp) / 2;
99             double eigen2 = (s11 + s22 - tmp) / 2;
100
101             double v1 = s12
102                 / Math.sqrt((eigen1 - s11) * (eigen1 - s11) + s12 * s12);
103             double v2 = (eigen1 - s11)
104                 / Math.sqrt((eigen1 - s11) * (eigen1 - s11) + s12 * s12);
105
106             double a = 2 * Math.sqrt(eigen1);
107             double b = 2 * Math.sqrt(eigen2);
108             double phi = Math.toDegrees(Math.atan2(v2, v1));
109             return new Ellipse(cx, cy, a, b, phi);
110         }
111     }

```

6.98.2.6 static `Pixel` [][] edu.ou.asgbook.segmentation.RegionProperty.getPixelsInRegions (`LatLonGrid data1`, `LabelResult objects1`)
[static]

All the pixels for each region.

The array is organized as pixels[regno][pixelno]

Parameters:

data1

objects1

Returns:

```

135         {
136             @SuppressWarnings("unchecked")
137             List<Pixel>[] regions = new List[objects1.maxlabel+1];
138             for (int reg=1; reg < regions.length; ++reg){
139                 regions[reg] = new ArrayList<Pixel>();
140             }

```

```

141         for (int i=0; i < data1.getNumLat(); ++i) for (int j=0; j < data1.getNumLon(); ++j)
142             int reg = objects1.label.getValue(i, j);
143             if (reg > 0){
144                 regions[reg].add(new Pixel(i, j, data1.getValue(i, j)));
145             }
146         }
147         Pixel[][] result = new Pixel[regions.length][];
148         for (int i=1; i < result.length; ++i){
149             result[i] = regions[i].toArray(new Pixel[0]);
150         }
151         return result;
152     }

```

6.98.2.7 int edu.ou.asgbook.segmentation.RegionProperty.getSize ()

```

49         {
50             return xstat.getNumSamples();
51         }

```

6.98.2.8 static void edu.ou.asgbook.segmentation.RegionProperty.main (String[] args) throws Exception [static]

```

206         {
207             File out = OutputDirectory.getDefault("regionproperty");
208
209             // data
210             LatLonGrid grid = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalP
211             KmlWriter.write(grid, out, "orig", PngWriter.createCoolToWarmColormap());
212
213             // global thresh
214             int thresh = 20;
215             ThresholdSegmenter seg = new ThresholdSegmenter(thresh);
216             LabelResult labelResult = seg.label(grid);
217             labelResult.label.setMissing(-1); // to get background color
218             KmlWriter.write(labelResult.label, out, "allcities_"+thresh, PngWriter.createRandom
219
220             // prune cities less than 5 pixels in size
221             for (int sizethresh = 2; sizethresh <= 5; ++sizethresh){
222                 LabelResult pruned = pruneBySize(labelResult, grid, sizethresh);
223                 pruned.label.setMissing(-1); // to get background color
224                 KmlWriter.write(pruned.label, out, "sizepruned_"+sizethresh, PngWriter.createR
225                 // get geocode
226                 RegionProperty[] prop = RegionProperty.compute(pruned, grid);
227                 for (int i=1; i < prop.length; ++i){
228                     LatLon loc = grid.getLocation(prop[i].getCx(), prop[i].getCy());
229                     UsaZipcode.Entry entry = UsaZipcode.getInstance().getEntryClosestTo(loc);
230                     System.out.println(entry + " " + prop[i].getEllipseFit());
231                 }
232
233                 // there are more efficient ways to paint an ellipse, but this will do
234                 Ellipse[] ellipses = new Ellipse[prop.length];
235                 for (int i=1; i < prop.length; ++i){

```

```

236         ellipses[i] = prop[i].getEllipseFit();
237     }
238     LatLonGrid ellipse = LatLonGrid.copyOf(pruned.label);
239     ellipse.fill(0);
240     int nrows = ellipse.getNumLat();
241     int ncols = ellipse.getNumLon();
242     for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
243         for (int k=1; k < prop.length; ++k){
244             if (ellipses[k].contains(i, j)){
245                 ellipse.setValue(i, j, k); // paint pixel
246             }
247         }
248     }
249     KmlWriter.write(ellipse, out, "ellipse_"+sizethresh, PngWriter.createRandomColormap());
250 }
251 }

```

6.98.2.9 static [LabelResult](#) edu.ou.asgbook.segmentation.RegionProperty.prune([LabelResult](#) input, boolean[] keep) [static]

Regions for which keep=false are removed.

```

157                                     {
158         // find mapping
159         int[] newRegionNo = new int[keep.length]; // init to zero
160         int numRegions = 0;
161         for (int i=1; i < keep.length; ++i){
162             if ( keep[i] ){
163                 ++numRegions;
164                 newRegionNo[i] = numRegions;
165             }
166         }
167
168         // replace old label by new label
169         LatLonGrid newLabel = LatLonGrid.copyOf(input.label);
170         int nrows = newLabel.getNumLat();
171         int ncols = newLabel.getNumLon();
172         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
173             int oldno = input.label.getValue(i, j);
174             newLabel.setValue(i, j, newRegionNo[oldno]);
175         }
176         return new LabelResult(newLabel, numRegions);
177     }

```

6.98.2.10 static [LabelResult](#) edu.ou.asgbook.segmentation.RegionProperty.pruneBySize([LabelResult](#) input, [LatLonGrid](#) grid, int sizethresh) [static]

Regions smaller than sizethresh are removed.

```
182
183     RegionProperty[] prop = RegionProperty.compute(input, grid);
184
185     // find mapping
186     int[] newRegionNo = new int[prop.length]; // init to zero
187     int numRegions = 0;
188     for (int i=1; i < prop.length; ++i){
189         if ( prop[i].getSize() >= sizethresh ){
190             ++numRegions;
191             newRegionNo[i] = numRegions;
192         }
193     }
194
195     // replace old label by new label
196     LatLonGrid newLabel = LatLonGrid.copyOf(input.label);
197     int nrows = newLabel.getNumLat();
198     int ncols = newLabel.getNumLon();
199     for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
200         int oldno = input.label.getValue(i,j);
201         newLabel.setValue(i,j, newRegionNo[oldno]);
202     }
203     return new LabelResult(newLabel, numRegions);
204 }
```


6.99 edu.ou.asgbook.projections.Remapper Class Reference

Utilities to remap one map projection to another.

Static Public Member Functions

- static int [nearestNeighbor](#) (double rowno, double colno, int[][] input, int missing)
- static int [bilinearInterpolation](#) (double rowno, double colno, int[][] input, int missing)

6.99.1 Detailed Description

Utilities to remap one map projection to another.

Author:

valliappa.lakshmanan

6.99.2 Member Function Documentation

6.99.2.1 static int edu.ou.asgbook.projections.Remapper.bilinearInterpolation (double rowno, double colno, int input[][], int missing) [static]

```

26
27         final int row0 = (int) Math.floor( rowno );
28         final int col0 = (int) Math.floor( colno );
29         final int row1 = (int) Math.ceil( rowno );
30         final int col1 = (int) Math.ceil( colno );
31         final int nrows = input.length;
32         final int ncols = (nrows > 0)? input[0].length : 0;
33
34         int npts = 0;
35         double totwt = 0;
36         double totval = 0;
37         for (int row = row0; row <= row1; ++row){
38             for (int col = col0; col <= col1; ++col){
39                 if ( row >= 0 && col >= 0 && row < nrows && col < ncols && input[row][col] != missing )
40                     double rowwt = 1 - Math.abs(rowno-row);
41                     double colwt = 1 - Math.abs(colno-col);
42                     double wt = rowwt * colwt;
43                     npts++;
44                     totwt += wt;
45                     totval += wt * input[row][col];
46             }
47         }

```

```
48         }
49
50         // weighted average
51         if (npts == 0){
52             return missing;
53         } else {
54             return (int) Math.round(totval / totwt);
55         }
56     }
```

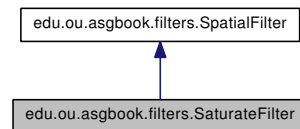
6.99.2.2 static int edu.ou.asgbook.projections.Remapper.nearestNeighbor (double *rowno*, double *colno*, int *input*[[]], int *missing*) [static]

```
14
15     final int row = (int) Math.round( rowno );
16     final int col = (int) Math.round( colno );
17     final int nrows = input.length;
18     final int ncols = (nrows > 0)? input[0].length : 0;
19     if ( row >= 0 && col >= 0 && row < nrows && col < ncols ){
20         return input[row][col];
21     } else {
22         return missing;
23     }
24 }
```

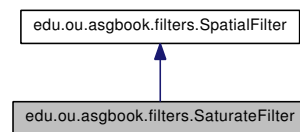
6.100 edu.ou.asgbook.filters.SaturateFilter Class Reference

Sets all values < MIN to MIN and all values > MAX to MAX.

Inheritance diagram for edu.ou.asgbook.filters.SaturateFilter:



Collaboration diagram for edu.ou.asgbook.filters.SaturateFilter:



Public Member Functions

- [SaturateFilter](#) (int min, int max)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)
- [LatLonGrid saturate](#) (final [LatLonGrid](#) input)

6.100.1 Detailed Description

Sets all values < MIN to MIN and all values > MAX to MAX.

Author:

Valliappa.Lakshmanan

6.100.2 Constructor & Destructor Documentation

6.100.2.1 edu.ou.asgbook.filters.SaturateFilter.SaturateFilter (int *min*, int *max*)

```
16                                     {
17         this.min = min;
18         this.max = max;
19     }
```

6.100.3 Member Function Documentation

6.100.3.1 Override [LatLonGrid](#) edu.ou.asgbook.filters.SaturateFilter.filter ([LatLonGrid](#) *input*)

```
22                                     {
23     return saturate(input);
24 }
```

6.100.3.2 [LatLonGrid](#) edu.ou.asgbook.filters.SaturateFilter.saturate (final [LatLonGrid](#) *input*)

```
26                                     {
27     LatLonGrid output = LatLonGrid.copyOf(input);
28     int[][] outData = output.getData();
29     int[][] inData = input.getData();
30     for (int i=0; i < output.getNumLat(); ++i){
31         for (int j=0; j < output.getNumLon(); ++j){
32             int inval = inData[i][j];
33             if ( inval < min || inval == input.getMissing() ){
34                 outData[i][j] = min;
35             } else if ( inval > max ){
36                 outData[i][j] = max;
37             }
38         }
39     }
40     return output;
41 }
```

6.101 edu.ou.asgbook.core.ScalarStatistic Class Reference

A utility class to compute mean, variance of a streaming set of inputs.

Public Member Functions

- void [update](#) (double x)
- void [update](#) (double x, int relwt)
- void [update](#) ([ScalarStatistic](#) other)
- double [getMean](#) ()
- double [getMin](#) ()
- double [getMax](#) ()
- double [getVariance](#) ()
- double [getStdDeviation](#) ()
- Override String [toString](#) ()
- int [getNumSamples](#) ()

6.101.1 Detailed Description

A utility class to compute mean, variance of a streaming set of inputs.

Author:

v.lakshmanan

6.101.2 Member Function Documentation

6.101.2.1 double edu.ou.asgbook.core.ScalarStatistic.getMax ()

```
84                                     {  
85         return max;  
86     }
```

6.101.2.2 double edu.ou.asgbook.core.ScalarStatistic.getMean ()

```
75                                     {  
76         compute();  
77         return mean;  
78     }
```

6.101.2.3 double edu.ou.asgbook.core.ScalarStatistic.getMin ()

```
80         {
81         return min;
82     }
```

6.101.2.4 int edu.ou.asgbook.core.ScalarStatistic.getNumSamples ()

```
103         {
104         return N;
105     }
```

6.101.2.5 double edu.ou.asgbook.core.ScalarStatistic.getStdDeviation ()

```
93         {
94         compute();
95         return stddev;
96     }
```

6.101.2.6 double edu.ou.asgbook.core.ScalarStatistic.getVariance ()

```
88         {
89         compute();
90         return var;
91     }
```

6.101.2.7 Override String edu.ou.asgbook.core.ScalarStatistic.toString ()

```
99         {
100         return "value = " + getMean() + "+/-" + getStdDeviation() + " based on " + N + " s
101     }
```

6.101.2.8 void edu.ou.asgbook.core.ScalarStatistic.update ([ScalarStatistic](#) *other*)

```
47         {
48         sumx += other.sumx;
49         sumx2 += other.sumx2;
50         if ( N == 0 ){
51             min = other.min;
52             max = other.max;
53         } else if (other.N != 0){
54             min = Math.min(min, other.min);
```

```
55         max = Math.max(max, other.max);
56     }
57     N += other.N;
58 }
```

6.101.2.9 void edu.ou.asgbook.core.ScalarStatistic.update (double x, int *relwt*)

```
35                                     {
36     sumx += (x*relwt);
37     sumx2 += (x*x*relwt);
38     if ( N == 0 ){
39         min = max = x;
40     } else {
41         min = Math.min(min, x);
42         max = Math.max(max, x);
43     }
44     N += relwt;
45 }
```

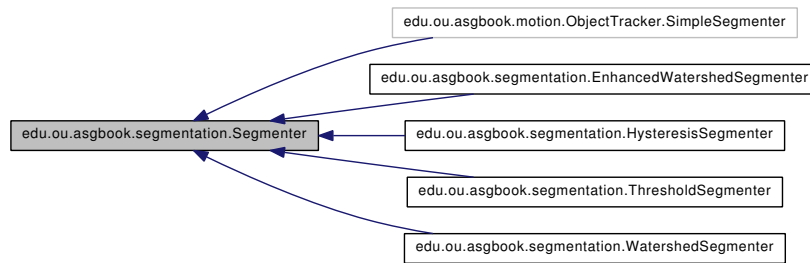
6.101.2.10 void edu.ou.asgbook.core.ScalarStatistic.update (double x)

```
23                                     {
24     sumx += x;
25     sumx2 += x*x;
26     if ( N == 0 ){
27         min = max = x;
28     } else {
29         min = Math.min(min, x);
30         max = Math.max(max, x);
31     }
32     ++N;
33 }
```

6.102 edu.ou.asgbook.segmentation.Segmenter Interface Reference

Object identification technique.

Inheritance diagram for edu.ou.asgbook.segmentation.Segmenter:



Public Member Functions

- abstract [LabelResult](#) label ([LatLonGrid](#) data)
Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

6.102.1 Detailed Description

Object identification technique.

Author:

valliappa.lakshmanan

6.102.2 Member Function Documentation

6.102.2.1 abstract [LabelResult](#) edu.ou.asgbook.segmentation.Segmenter.label ([LatLonGrid](#) data) [pure virtual]

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

.. All pixels > thresh are part of an object.

Implemented in [edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter](#),
[edu.ou.asgbook.segmentation.Hysteresis-](#)

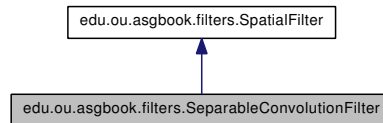
6.102 edu.ou.asgbook.segmentation.Segmenter Interface Reference 333

Segmenter, [edu.ou.asgbook.segmentation.ThresholdSegmenter](#), and
[edu.ou.asgbook.segmentation.WatershedSegmenter](#).

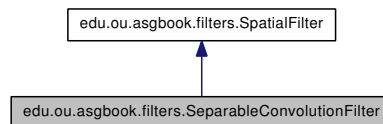
6.103 edu.ou.asgbook.filters.SeparableConvolution-Filter Class Reference

An optimized convolution filter.

Inheritance diagram for edu.ou.asgbook.filters.SeparableConvolutionFilter:



Collaboration diagram for edu.ou.asgbook.filters.SeparableConvolutionFilter:



Public Member Functions

- [SeparableConvolutionFilter](#) (double[] coeffs_x, double[] coeffs_y)
- [SeparableConvolutionFilter](#) (double[] coeffs)
- [LatLonGrid smooth](#) (final [LatLonGrid](#) input)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)

Static Public Member Functions

- static [SeparableConvolutionFilter boxcar](#) (int numx, int numy)
- static [SeparableConvolutionFilter gauss](#) (int numx, int numy)
- static double[] [gauss](#) (int numx)
- static void [main](#) (String[] args) throws Exception

6.103.1 Detailed Description

An optimized convolution filter.

Author:

Valliappa.Lakshmanan

6.103.2 Constructor & Destructor Documentation

6.103.2.1 edu.ou.asgbook.filters.SeparableConvolutionFilter.SeparableConvolutionFilter (double[] *coeffs_x*, double[] *coeffs_y*)

```

24                                     {
25         this.coeffs_x = coeffs_x;
26         if ( coeffs_x.length % 2 == 0 ){
27             throw new IllegalArgumentException("Dimensions of coefficients array needs to be odd");
28         }
29         this.coeffs_y = coeffs_y;
30         if ( coeffs_y.length % 2 == 0 ){
31             throw new IllegalArgumentException("Dimensions of coefficients array needs to be odd");
32         }
33     }

```

6.103.2.2 edu.ou.asgbook.filters.SeparableConvolutionFilter.SeparableConvolutionFilter (double[] *coeffs*)

```

35                                     {
36         this(coeffs,coeffs);
37     }

```

6.103.3 Member Function Documentation

6.103.3.1 static [SeparableConvolutionFilter](#) edu.ou.asgbook.filters.SeparableConvolutionFilter.boxcar (int *numx*, int *numy*) [static]

```

92                                     {
93         double[] coeffs_x = new double[numx];
94         double[] coeffs_y = new double[numy];
95         for (int i=0; i < numx; ++i){
96             coeffs_x[i] = 1.0/numx;
97         }
98         for (int i=0; i < numy; ++i){
99             coeffs_y[i] = 1.0/numy;
100        }
101         return new SeparableConvolutionFilter(coeffs_x, coeffs_y);
102     }

```

6.103.3.2 Override [LatLonGrid](#) edu.ou.asgbook.filters.SeparableConvolutionFilter.filter ([LatLonGrid](#) *input*)

```

142                                     {
143         return smooth(input);
144     }

```

6.103.3.3 static double [] edu.ou.asgbook.filters.SeparableConvolutionFilter.gauss (int numx) [static]

```

108         {
109         double[] coeffs = new double[numx];
110         double sigmax = numx / 6.0; // 3-sigma on either side
111         for (int i=0; i < coeffs.length; ++i){
112             double x = (i - coeffs.length/2.0)/sigmax;
113             coeffs[i] = Math.exp(-(x*x));
114         }
115         return coeffs;
116     }

```

6.103.3.4 static [SeparableConvolutionFilter](#) edu.ou.asgbook.filters.SeparableConvolutionFilter.gauss (int numx, int numy) [static]

```

104         {
105         return new SeparableConvolutionFilter(gauss(numx), gauss(numy));
106     }

```

6.103.3.5 static void edu.ou.asgbook.filters.SeparableConvolutionFilter.main (String[] args) throws Exception [static]

```

118         {
119         // create output directory
120         File out = OutputDirectory.getDefault("separable");
121
122         // read input
123         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new G
124         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
125
126         // boxcar
127         {
128             SeparableConvolutionFilter filter = SeparableConvolutionFilter.boxcar(5, 5);
129             LatLonGrid sm = filter.smooth(popdensity);
130             KmlWriter.write(sm, out, "boxcar", PngWriter.createCoolToWarmColormap());
131         }
132
133         // gauss
134         {
135             SeparableConvolutionFilter filter = SeparableConvolutionFilter.gauss(11, 11);
136             LatLonGrid sm = filter.smooth(popdensity);
137             KmlWriter.write(sm, out, "gauss", PngWriter.createCoolToWarmColormap());
138         }
139     }

```

6.103.3.6 [LatLonGrid](#) edu.ou.asgbook.filters.Separable-ConvolutionFilter.smooth (final [LatLonGrid](#) *input*)

```

39                                     {
40     int[][] inData = input.getData();
41     final int nx = input.getNumLat();
42     final int ny = input.getNumLon();
43
44     // filter the rows
45     final int hx = coeffs_x.length / 2;
46     int[][] rowResult = new int[nx][ny];
47     for (int j=0; j < ny; ++j){
48         for (int i=hx; i < (nx-hx); ++i){
49             double tot = 0;
50             double wt = 0;
51             for (int m=-hx; m <= hx; ++m){
52                 double coeff = coeffs_x[m+hx];
53                 int inval = inData[i+m][j];
54                 if (inval != input.getMissing()){
55                     tot += inval*coeff;
56                     wt += coeff;
57                 }
58             }
59             if ( wt > 0 ){
60                 rowResult[i][j] = (int)( Math.round(tot / wt) );
61             }
62         }
63     }
64
65     // now filter the columns of rowResult
66     inData = rowResult;
67     LatLonGrid output = LatLonGrid.copyOf(input);
68     output.fill(output.getMissing());
69     final int hy = coeffs_y.length / 2;
70     int[][] outData = output.getData();
71     for (int i=0; i < nx; ++i){
72         for (int j=hy; j < (ny-hy); ++j){
73             double tot = 0;
74             double wt = 0;
75             for (int n=-hy; n <= hy; ++n){
76                 double coeff = coeffs_y[n+hy];
77                 int inval = inData[i][j+n];
78                 if (inval != input.getMissing()){
79                     tot += inval*coeff;
80                     wt += coeff;
81                 }
82             }
83             if ( wt > 0 ){
84                 outData[i][j] = (int)( Math.round(tot / wt) );
85             }
86         }
87     }
88
89     return output;
90 }

```

6.104 edu.ou.asgbook.dataset.SeviriInfrared-Temperature Class Reference

To read binary dump output from WDSS-II (<http://www.wdssii.org/>).

Static Public Member Functions

- static Pair< [LatLonGrid](#), Date > [read](#) (File f) throws IOException, ParseException
- static Pair< [LatLonGrid](#), Date >[] [readAll](#) (File dir) throws IOException, ParseException
- static void [main](#) (String[] args) throws Exception

Static Package Functions

- [[static initializer](#)]

6.104.1 Detailed Description

To read binary dump output from WDSS-II (<http://www.wdssii.org/>).

Author:

v.lakshmanan

6.104.2 Member Function Documentation

6.104.2.1]

edu.ou.asgbook.dataset.SeviriInfraredTemperature.[\[static initializer\]](#) () [[static](#), [package](#)]

6.104.2.2 static void edu.ou.asgbook.dataset.SeviriInfraredTemperature.main (String[] args) throws Exception [[static](#)]

```

86                                     {
87     File f = new File("data/seviri");
88     Pair<LatLonGrid,Date>[] grids = readAll(f);
89
90     // create output directory
91     File out = OutputDirectory.getDefault("seviri");
92     // write out as image, for viewing
93     for (int i=0; i < grids.length; ++i){

```

```

94         KmlWriter.write(grid[i].first, out, "ir_"+i, PngWriter.createCoolToWarmColormap());
95     }
96 }

```

6.104.2.3 static Pair<LatLonGrid,Date> edu.ou.asgbook.dataset.SeviriInfraredTemperature.read (File *f*) throws IOException, ParseException [static]

```

32
33 // parse filename: eg: MSG_20050105-150000_556x1111_60.00_-10.00_Channel_09_0.027_0.027.llg
34 String[] pieces = f.getName().replace(".llg", "").split("_");
35 int pieceno = 0;
36 pieceno++; // typeName ignored
37 String date = pieces[pieceno++];
38 String[] dimpieces = pieces[pieceno++].split("x");
39 int numRows = Integer.parseInt(dimpieces[0]);
40 int numcols = Integer.parseInt(dimpieces[1]);
41 float nwlat = Float.parseFloat(pieces[pieceno++]);
42 float nwlon = Float.parseFloat(pieces[pieceno++]);
43 ++pieceno; // subtype ignored
44 int numleft = pieces.length - pieceno - 2;
45 pieceno += numleft; // subtype has an underscore
46 float deltalat = Float.parseFloat(pieces[pieceno++]);
47 float deltalon = Float.parseFloat(pieces[pieceno++]);
48
49 // read in LatLonGrid
50 FileInputStream fis = new FileInputStream(f);
51 byte[] bytes = new byte[numRows*numcols];
52 fis.read(bytes);
53 LatLonGrid grid = new LatLonGrid(numRows, numcols, 0, new LatLon(nwlat,nwlon), deltalat, deltalon);
54 int index = 0;
55 for (int i=0; i < numRows; ++i){
56     for (int j=0; j < numcols; ++j){
57         int value = 256 + bytes[index++]; // 0 to 255
58         int reversed = 256 - value; // assign higher value to colder pixels
59         grid.setValue(i, j, reversed);
60     }
61 }
62
63 // format date
64 SimpleDateFormat df = new SimpleDateFormat("yyyyMMdd-HH:mm:ss");
65 Date gridTime = df.parse(date);
66
67 System.out.println("Read grid at " + gridTime);
68 return new Pair<LatLonGrid,Date>(grid,gridTime);
69 }

```

6.104.2.4 static Pair<LatLonGrid,Date> [] edu.ou.asgbook.dataset.SeviriInfraredTemperature.readAll (File *dir*) throws IOException, ParseException [static]

```

71

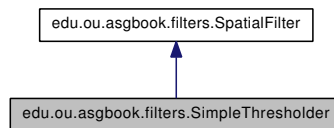
```

```
72         File[] files = dir.listFiles(new FileFilter(){
73             @Override
74             public boolean accept(File f) {
75                 return f.getName().endsWith(".llg");
76             }
77         });
78         @SuppressWarnings("unchecked")
79         Pair<LatLonGrid,Date>[] result = new Pair[files.length];
80         for (int i=0; i < result.length; ++i){
81             result[i] = read(files[i]);
82         }
83         return result;
84     }
```

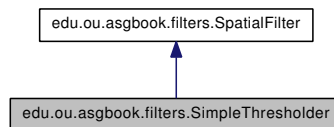

6.105 edu.ou.asgbook.filters.SimpleThresholder Class Reference

Replace pixel values with 1 or 0 depending on whether they are above or below a single threshold.

Inheritance diagram for edu.ou.asgbook.filters.SimpleThresholder:



Collaboration diagram for edu.ou.asgbook.filters.SimpleThresholder:



Public Member Functions

- [SimpleThresholder](#) (int thresh)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)
- [LatLonGrid threshold](#) (final [LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.105.1 Detailed Description

Replace pixel values with 1 or 0 depending on whether they are above or below a single threshold.

Author:

Valliappa.Lakshmanan

6.105.2 Constructor & Destructor Documentation

6.105.2.1 `edu.ou.asgbook.filters.SimpleThresholder.SimpleThresholder (int thresh)`

```

24                                     {
25         this.thresh = thresh;
26     }
```

6.105.3 Member Function Documentation

6.105.3.1 Override [LatLonGrid](#) `edu.ou.asgbook.filters.SimpleThresholder.filter (LatLonGrid input)`

```

29                                     {
30         return threshold(input);
31     }
```

6.105.3.2 `static void edu.ou.asgbook.filters.SimpleThresholder.main (String[] args) throws Exception [static]`

```

45                                     {
46         // create output directory
47         File out = OutputDirectory.getDefault("threshold");
48
49         // read input
50         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA);
51         popdensity.setMissing(0); // transparent
52         KmlWriter.write(popdensity, out, "popdensity", PngWriter.createCoolToWarmColormap());
53
54         // threshold
55         SimpleThresholder filter = new SimpleThresholder(100);
56         LatLonGrid thresh = filter.threshold(popdensity);
57         KmlWriter.write(thresh, out, "highdensity", PngWriter.createCoolToWarmColormap());
58     }
```

6.105.3.3 [LatLonGrid](#) `edu.ou.asgbook.filters.SimpleThresholder.threshold (final LatLonGrid input)`

```

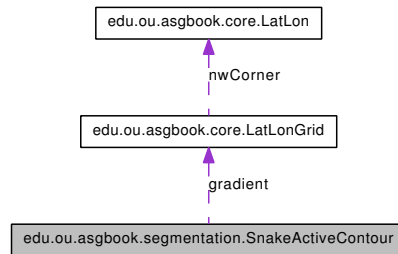
33                                     {
34         LatLonGrid output = LatLonGrid.copyOf(input);
35         int[][] outData = output.getData();
36         int[][] inData = input.getData();
37         for (int i=0; i < output.getNumLat(); ++i){
38             for (int j=0; j < output.getNumLon(); ++j){
39                 outData[i][j] = (inData[i][j] >= thresh)? 1 : 0;
40             }
41         }
```

```
42         return output;
43     }
```

6.106 edu.ou.asgbook.segmentation.SnakeActiveContour Class Reference

Active contour method of identifying objects.

Collaboration diagram for edu.ou.asgbook.segmentation.SnakeActiveContour:



Public Member Functions

- [SnakeActiveContour](#) ([LatLonGrid](#) gradient)
- [Snake pruneAndResample](#) ([Snake](#) inputSnake)
- [Snake resampleNodes](#) ([Snake](#) inputSnake)
- [Snake resample](#) ([Snake](#) inputSnake)
- [SnakeNode\[\] moveSnake](#) ([SnakeNode\[\]](#) pixels, int numIter)

Provide an initial guess of points.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

Classes

- class [Snake](#)
- class [SnakeNode](#)

6.106.1 Detailed Description

Active contour method of identifying objects.

Author:

v.lakshmanan

6.106.2 Constructor & Destructor Documentation

6.106.2.1 edu.ou.asgbook.segmentation.SnakeActiveContour.SnakeActiveContour (LatLonGrid gradient)

```

37                                     {
38     // Normalize the gradient grid to lie between 0 and 100
39     this.gradient = LatLonGrid.copyOf(gradient);
40     int maxgradient = 0;
41     for (int i=0; i < gradient.getNumLat(); ++i) for (int j=0; j < gradient.getNumLon(); ++j){
42         if ( gradient.getValue(i, j) > maxgradient ){
43             maxgradient = gradient.getValue(i, j);
44         }
45     }
46     for (int i=0; i < gradient.getNumLat(); ++i) for (int j=0; j < gradient.getNumLon(); ++j){
47         if ( gradient.getValue(i, j) != gradient.getMissing() ){
48             this.gradient.setValue(i, j, (gradient.getValue(i, j) * GRADIENT_SCALE) / maxgradient);
49         } else {
50             this.gradient.setValue(i, j, 0);
51         }
52     }
53 }
```

6.106.3 Member Function Documentation

6.106.3.1 static void edu.ou.asgbook.segmentation.SnakeActiveContour.main (String[] args) throws Exception [static]

Parameters:

args

```

351                                     {
352     File out = OutputDirectory.getDefault("snake");
353
354     // data
355     LatLonGrid grid = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulation.LatLonGrid());
356     KmlWriter.write(grid, out, "orig", PngWriter.createCoolToWarmColormap());
357
358     // find cities > 10 px and more than 20,000 residents/km^2
359     int thresh = 20;
360     Segmenter seg = new HysteresisSegmenter(thresh, thresh-5);
361     LabelResult labelResult = seg.label(grid);
362     KmlWriter.write(labelResult.label, out, "allcities_"+thresh, PngWriter.createRandomColormap());
363     int sizethresh = 10;
364     LabelResult pruned = RegionProperty.pruneBySize(labelResult, grid, sizethresh);
365     RegionProperty[] prop = RegionProperty.compute(pruned, grid);
366     KmlWriter.write(pruned.label, out, "largecities_"+sizethresh, PngWriter.createRandomColormap());
367
368     // threshold image and compute gradient of thresholded image
369     LatLonGrid binaryImage = new SimpleThresholder(1).threshold(pruned.label);
370     binaryImage = new DilationFilter(1).filter(binaryImage);
371     KmlWriter.write(binaryImage, out, "thresh", PngWriter.createCoolToWarmColormap());
```

```

372     LatLonGrid gradient = new LoGEdgeFilter(2,1).edgeFilter(binaryImage);
373     KmlWriter.write(gradient, out, "gradient", PngWriter.createCoolToWarmColormap());
374
375     // for each city, initialize a snake
376     SnakeActiveContour alg = new SnakeActiveContour(gradient);
377     int numiter = 30;
378     for (int i=1; i < prop.length; ++i){
379         double cx = prop[i].getCx();
380         double cy = prop[i].getCy();
381         // square box enclosing the center point that is larger than core area
382         double initsize = 3 * Math.sqrt( prop[i].getSize() );
383         SnakeNode[] snakepts = new SnakeNode[4];
384         snakepts[0] = new SnakeNode(cx + initsize, cy - initsize, grid.getNumLat(), gr
385         snakepts[1] = new SnakeNode(cx + initsize, cy + initsize, grid.getNumLat(), gr
386         snakepts[2] = new SnakeNode(cx - initsize, cy + initsize, grid.getNumLat(), gr
387         snakepts[3] = new SnakeNode(cx - initsize, cy - initsize, grid.getNumLat(), gr
388
389         SnakeNode[] snake = alg.moveSnake(snakepts, numiter);
390         // mark snake points on grid
391         for (int k=0; k < snake.length; ++k){
392             grid.setValue(snake[k].getX(), snake[k].getY(), 1000);
393         }
394     }
395
396     // write out original grid with snake points marked
397     KmlWriter.write(grid, out, "snakes", PngWriter.createCoolToWarmColormap());
398 }

```

6.106.3.2 SnakeNode [] edu.ou.asgbook.segmentation.SnakeActiveContour.moveSnake (SnakeNode[] *pixels*, int *numIter*)

Provide an initial guess of points.

```

189                                     {
190         Snake snake = new Snake(pixels);
191         snake = resample(snake); // get it to desired length, then start moving it
192         snake = moveSnake(snake, numIter);
193         return complete(snake.pts);
194     }

```

6.106.3.3 Snake edu.ou.asgbook.segmentation.SnakeActiveContour.pruneAndResample (Snake *inputSnake*)

```

140                                     {
141         List<SnakeNode> nodes = new ArrayList<SnakeNode>(Arrays.asList(inputSnake.pts));
142         int numNodes = nodes.size();
143         if (numNodes <= 3) return inputSnake;
144
145         int dist_thresh = SNAKE_DIST_BETWEEN_PTS * SNAKE_DIST_BETWEEN_PTS;
146

```

```

147         for (int i = 0; i < numNodes; i++) {
148             SnakeNode curPt = nodes.get(i);
149             int next = i + 1; if (next == numNodes) next = 0;
150             SnakeNode nextPt = nodes.get(next);
151
152             int distsq = (nextPt.getX()-curPt.getX())*(nextPt.getX()-curPt.getX())+(nextPt.getY()-curPt.getY())*(nextPt.getY()-curPt.getY());
153             boolean currNotOnGradient = gradient.getData()[curPt.getX()][curPt.getY()] < 30;
154             boolean nextNotOnGradient = gradient.getData()[nextPt.getX()][nextPt.getY()] < 30;
155
156             boolean remove = numNodes > SNAKE_LENGTH && ( (distsq < 20) || (distsq < 80 && nextNotOnGradient) );
157             if (remove) {
158                 nodes.remove(next);
159                 --numNodes;
160                 --i; // retry this node
161             } else if (distsq > dist_thresh && (currNotOnGradient || nextNotOnGradient)) {
162                 SnakeNode newPt = new SnakeNode((curPt.getX()+nextPt.getX())/2, (curPt.getY()+nextPt.getY())/2);
163                 nodes.add(i+1, newPt);
164                 numNodes++;
165             }
166         }
167         return new Snake(nodes.toArray(new SnakeNode[0]));
168     }

```

6.106.3.4 [Snake](#) edu.ou.asgbook.segmentation.SnakeActiveContour.resample ([Snake](#) *inputSnake*)

```

182         {
183             return pruneAndResample(inputSnake);
184         }

```

6.106.3.5 [Snake](#) edu.ou.asgbook.segmentation.SnakeActiveContour.resample- Nodes ([Snake](#) *inputSnake*)

```

170         {
171             SnakeNode[] full = complete(inputSnake.pts);
172             if ( full.length <= SNAKE_LENGTH ) return inputSnake;
173             System.out.println("Resampling " + full.length + " to " + SNAKE_LENGTH);
174             int sampleInterval = full.length / SNAKE_LENGTH;
175             SnakeNode[] sampled = new SnakeNode[SNAKE_LENGTH];
176             for (int i=0; i < sampled.length; ++i){
177                 sampled[i] = full[i*sampleInterval];
178             }
179             return new Snake(sampled);
180         }

```

6.107 edu.ou.asgbook.segmentation.SnakeActive-Contour.Snake Class Reference

Public Member Functions

- [Snake](#) (SnakeNode[] pts)
- SnakeNode [get](#) (int k)
the snake is a closed curve, so does modulo to get points
- SnakeNode[] [getNodes](#) ()
- double [computeEnergy](#) (int candx, int candy, SnakeNode current, SnakeNode previous, SnakeNode next)

6.107.1 Constructor & Destructor Documentation

6.107.1.1 edu.ou.asgbook.segmentation.SnakeActiveContour.Snake.Snake (SnakeNode[] pts)

```

103                                     {
104         this.pts = pts;
105         meanDistBetweenPts = 0;
106         if ( this.pts.length == 0 ) return;
107
108         // compute mean dist
109         for (int i=0; i < pts.length; ++i){
110             SnakeNode curr = pts[i];
111             SnakeNode next = pts[ (i+1)%(pts.length) ];
112             meanDistBetweenPts += Math.sqrt( curr.getDistanceSquared(next) );
113         }
114         meanDistBetweenPts /= pts.length;
115     }

```

6.107.2 Member Function Documentation

6.107.2.1 double edu.ou.asgbook.segmentation.SnakeActive-Contour.Snake.computeEnergy (int candx, int candy, SnakeNode current, SnakeNode previous, SnakeNode next)

```

130
131         double E_total, E_edgestrength, E_smoothness, E_continuity ;
132         E_edgestrength = gradient.getData()[candx][candy];
133         E_smoothness = Math.pow(previous.getX() - 2 * candx + next.getX(), 2) + Math.p
134         E_continuity = Math.abs( Math.sqrt(previous.getDistanceSquared(candx,candy)) -
135         E_total = current.alpha * E_continuity + current.beta * E_smoothness - current
136         return E_total;
137     }

```


6.107.2.2 SnakeNode edu.ou.asgbook.segmentation.SnakeActiveContour.Snake.get (int *k*)

the snake is a closed curve, so does modulo to get points

```
118         {
119             int len = pts.length;
120             while ( k < 0 ){
121                 k += len;
122             }
123             return pts[k%len];
124         }
```

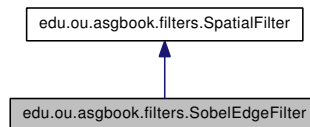
6.107.2.3 SnakeNode [] edu.ou.asgbook.segmentation.SnakeActiveContour.Snake.getNodes ()

```
126         {
127             return pts;
128         }
```

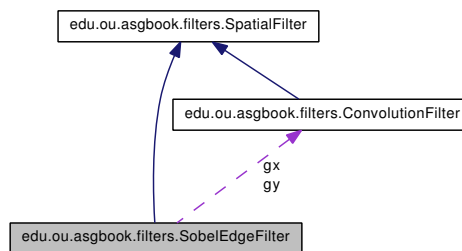
6.108 edu.ou.asgbook.filters.SobelEdgeFilter Class Reference

Find edges in a grid.

Inheritance diagram for edu.ou.asgbook.filters.SobelEdgeFilter:



Collaboration diagram for edu.ou.asgbook.filters.SobelEdgeFilter:



Public Member Functions

- [SobelEdgeFilter](#) ()
- [LatLonGrid saturate](#) (final [LatLonGrid](#) input, int thresh)
treat values > thresh as equal to thresh
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)
- [LatLonGrid edgeFilter](#) (final [LatLonGrid](#) input)
- [LatLonGrid edgeFilter](#) (final [LatLonGrid](#) input, File out)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.108.1 Detailed Description

Find edges in a grid.

Author:

valliappa.lakshmanan

6.108.2 Constructor & Destructor Documentation**6.108.2.1 edu.ou.asgbook.filters.SobelEdgeFilter.SobelEdgeFilter ()**

```

22         {
23         double[][] xc = new double[3][3];
24         double[][] yc = new double[3][3];
25         xc[0][0] = xc[0][2] = yc[0][0] = yc[2][0] = -1;
26         xc[0][1] = yc[1][0] = -2;
27         xc[2][0] = xc[2][2] = yc[0][2] = yc[2][2] = 1;
28         xc[2][1] = yc[1][2] = 2;
29         gx = new ConvolutionFilter(xc);
30         gy = new ConvolutionFilter(yc);
31     }

```

6.108.3 Member Function Documentation**6.108.3.1 [LatLonGrid](#) edu.ou.asgbook.filters.SobelEdgeFilter.edgeFilter (final [LatLonGrid](#) *input*, *File out*)**

```

57         {
58         LatLonGrid g1 = gx.convolve(input);
59         KmlWriter.debugWrite(g1, out, "gx");
60         LatLonGrid g2 = gy.convolve(input);
61         KmlWriter.debugWrite(g2, out, "gy");
62         for (int i=0; i < g1.getNumLat(); ++i){
63             for (int j=0; j < g1.getNumLon(); ++j){
64                 if (g1.getValue(i,j) != g1.getMissing() && g2.getValue(i,j) != g2.getMissing()){
65                     int gradient = Math.abs( g1.getValue(i, j) ) + Math.abs( g2.getValue(i,j) );
66                     g1.setValue(i, j, gradient);
67                 } else {
68                     g1.setValue(i,j, g1.getMissing());
69                 }
70             }
71         }
72         return g1;
73     }

```

6.108.3.2 [LatLonGrid](#) edu.ou.asgbook.filters.SobelEdgeFilter.edgeFilter (final [LatLonGrid](#) *input*)

```

53         {
54         return edgeFilter(input, null);
55     }

```

6.108.3.3 Override [LatLonGrid](#) edu.ou.asgbook.filters.SobelEdgeFilter.filter ([LatLonGrid](#) *input*)

```

49                                     {
50         return edgeFilter(input);
51     }

```

6.108.3.4 static void edu.ou.asgbook.filters.SobelEdgeFilter.main (String[] *args*) throws Exception [static]

```

75                                     {
76         // create output directory
77         File out = OutputDirectory.getDefault("sobel");
78
79         // read input
80         DataTransform t = new GlobalPopulation.LogScaling();
81         // DataTransform t = new GlobalPopulation.LinearScaling();
82         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, t).crop();
83         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
84
85         SobelEdgeFilter filter = new SobelEdgeFilter();
86         LatLonGrid edges = filter.edgeFilter(popdensity, out);
87         KmlWriter.write(edges, out, "edge", PngWriter.createCoolToWarmColormap());
88
89         ConvolutionFilter sm = new ConvolutionFilter(ConvolutionFilter.gauss(11, 11));
90         popdensity = sm.smooth(popdensity);
91         edges = filter.edgeFilter(popdensity, null);
92         KmlWriter.write(edges, out, "smoothedge", PngWriter.createCoolToWarmColormap());
93
94         LatLonGrid saturated = filter.saturate(edges, 5000);
95         KmlWriter.write(saturated, out, "saturated", PngWriter.createCoolToWarmColormap());
96     }

```

6.108.3.5 [LatLonGrid](#) edu.ou.asgbook.filters.SobelEdgeFilter.saturate (final [LatLonGrid](#) *input*, int *thresh*)

treat values > thresh as equal to thresh

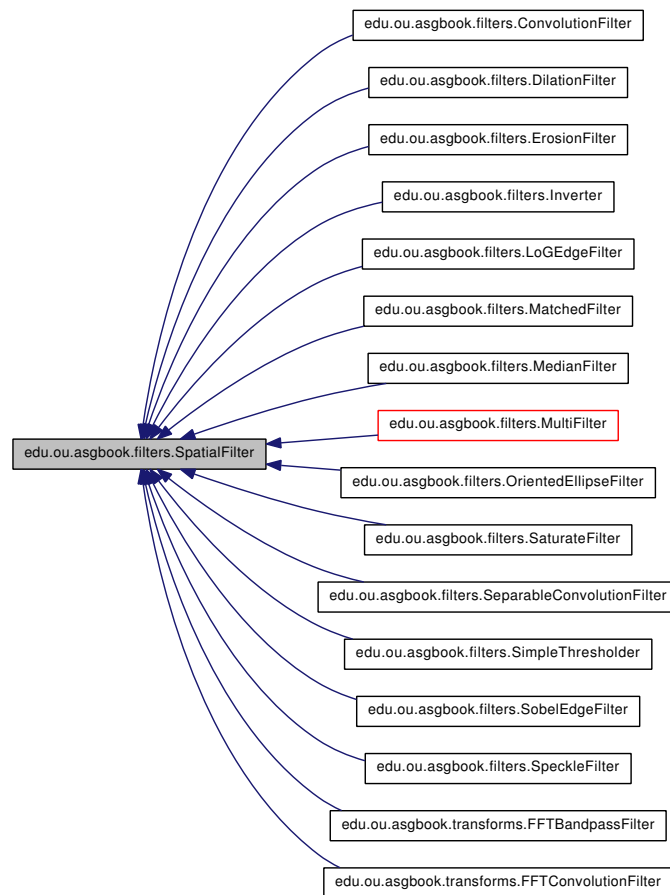
```

36                                     {
37         LatLonGrid result = LatLonGrid.copyOf(input);
38         for (int i=0; i < input.getNumLat(); ++i){
39             for (int j=0; j < input.getNumLon(); ++j){
40                 if ( result.getValue(i, j) > thresh ){
41                     result.setValue(i,j, thresh);
42                 }
43             }
44         }
45         return result;
46     }

```

6.109 edu.ou.asgbook.filters.SpatialFilter Interface Reference

Inheritance diagram for edu.ou.asgbook.filters.SpatialFilter:



Public Member Functions

- [LatLonGrid filter](#) (final [LatLonGrid](#) input)

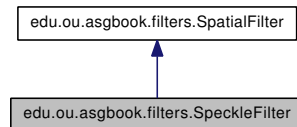
6.109.1 Member Function Documentation

6.109.1.1 [LatLonGrid](#) edu.ou.asgbook.filters.SpatialFilter.filter (final
[LatLonGrid](#) *input*)

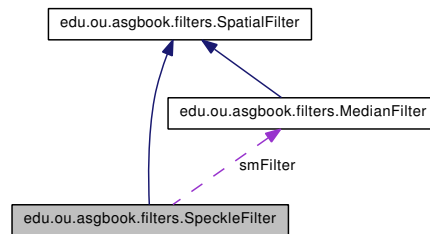
6.110 edu.ou.asgbook.filters.SpeckleFilter Class Reference

Denoising filter that removes speckle.

Inheritance diagram for edu.ou.asgbook.filters.SpeckleFilter:



Collaboration diagram for edu.ou.asgbook.filters.SpeckleFilter:



Public Member Functions

- [SpeckleFilter](#) (int halfSize, int maxChange)
- Override [LatLonGrid filter](#) ([LatLonGrid](#) input)
- [LatLonGrid speckleFilter](#) (final [LatLonGrid](#) input)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.110.1 Detailed Description

Denoising filter that removes speckle.

Author:

Valliappa.Lakshmanan

6.110.2 Constructor & Destructor Documentation

6.110.2.1 `edu.ou.asgbook.filters.SpeckleFilter.SpeckleFilter (int halfSize, int maxChange)`

```

25                                     {
26         this.smFilter = new MedianFilter(halfSize);
27         this.maxChange = maxChange;
28     }
```

6.110.3 Member Function Documentation

6.110.3.1 Override [LatLonGrid](#) `edu.ou.asgbook.filters.SpeckleFilter.filter (LatLonGrid input)`

```

31                                     {
32         return speckleFilter(input);
33     }
```

6.110.3.2 `static void edu.ou.asgbook.filters.SpeckleFilter.main (String[] args) throws Exception [static]`

```

56                                     {
57         // create output directory
58         File out = OutputDirectory.getDefault("speckle");
59
60         // read input
61         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new Gl
62         KmlWriter.write(popdensity, out, "orig", PngWriter.createCoolToWarmColormap());
63
64         // add noise
65         Random rand = new Random();
66         final int NOISE = 200;
67         for (int i=0; i < 1000; ++i){
68             int x = rand.nextInt(popdensity.getNumLat());
69             int y = rand.nextInt(popdensity.getNumLon());
70             int add = NOISE + rand.nextInt(NOISE/2);
71             popdensity.setValue(x, y, popdensity.getValue(x,y) + add);
72         }
73         KmlWriter.write(popdensity, out, "noisy", PngWriter.createCoolToWarmColormap());
74
75         // dilate
76         LatLonGrid dilate1 = new SpeckleFilter(3,NOISE).filter(popdensity);
77         KmlWriter.write(dilate1, out, "speckle_3", PngWriter.createCoolToWarmColormap());
78     }
```


6.110.3.3 [LatLonGrid](#) edu.ou.asgbook.filters.SpeckleFilter.speckleFilter (final [LatLonGrid](#) *input*)

```
35                                     {
36     LatLonGrid smoothed = smFilter.filter(input);
37     LatLonGrid output = LatLonGrid.copyOf(input);
38     int[][] inData = input.getData();
39     int[][] smData = smoothed.getData();
40     int nx = inData.length;
41     int ny = inData[0].length;
42     for (int i=0; i < nx; ++i){
43         for (int j=0; j < ny; ++j){
44             if (inData[i][j] != input.getMissing() &&
45                 smData[i][j] != smoothed.getMissing()){
46                 int diff = Math.abs(inData[i][j] - smData[i][j]);
47                 if (diff > maxChange){ // noise
48                     output.setValue(i, j, smData[i][j]);
49                 }
50             }
51         }
52     }
53     return output;
54 }
```

6.111 edu.ou.asgbook.usage.Sprawl Class Reference

Solution to a classroom assignment to identify regions of urban sprawl from the population density data.

Static Public Member Functions

- static void [runOnPopDensity](#) (boolean crop) throws Exception
- static void [findSprawl](#) ([LatLonGrid](#) grid1, [LatLonGrid](#) grid2, File out) throws Exception
- static void [main](#) (String[] args) throws Exception

6.111.1 Detailed Description

Solution to a classroom assignment to identify regions of urban sprawl from the population density data.

Author:

v.lakshmanan

6.111.2 Member Function Documentation

6.111.2.1 static void edu.ou.asgbook.usage.Sprawl.findSprawl ([LatLonGrid](#) *grid1*, [LatLonGrid](#) *grid2*, File *out*) throws Exception [static]

```

59
60         // write out input grids
61         KmlWriter.write(grid1, out, "pop_1990", PngWriter.createCoolToWarmColormap());
62         KmlWriter.write(grid2, out, "pop_2010", PngWriter.createCoolToWarmColormap());
63         System.out.println("Resolution of image = " + grid1.getLatRes() + "x" + grid1.getLonRes());
64         // Find optimal threshold on 2010 data using Otsu's method
65         final int MIN = 0;
66         final int MAX = 400;
67         final int incr = 1;
68         Histogram hist = new Histogram(MIN, incr, (MAX-MIN)/incr );
69         hist.update(grid2);
70         OtsuThresholdSelector threshold = new OtsuThresholdSelector(hist);
71         int thresh1 = threshold.getOptimalThreshold();
72         System.out.println("Optimal threshold=" + thresh1);
73
74         // A city consists of point with this threshold and contiguous points > some reason
75         int thresh2 = thresh1 * 2;
76         LabelResult label1990 = new HysteresisSegmenter(thresh1, thresh2).label(grid1);
77         KmlWriter.write(label1990.label, out, "label_1990", PngWriter.createRandomColormap());
78
79         LabelResult label2010 = new HysteresisSegmenter(thresh1, thresh2).label(grid2);
80         KmlWriter.write(label2010.label, out, "label_2010", PngWriter.createRandomColormap());

```

```

81
82 // grow regions and find region properties
83 RegionProperty[] props1 = RegionProperty.compute(label1990, grid1);
84 RegionProperty[] props2 = RegionProperty.compute(label2010, grid2);
85
86 // create a new grid that consists of city sizes
87 LatLonGrid citysize1990 = getCitySize(label1990, props1);
88 LatLonGrid citysize2010 = getCitySize(label2010, props2);
89 KmlWriter.write(citysize1990, out, "citysize_1990", PngWriter.createCoolToWarmColormap());
90 KmlWriter.write(citysize2010, out, "citysize_2010", PngWriter.createCoolToWarmColormap());
91
92 // compute and write out difference in size for every 2010 city
93 LatLonGrid changeInSize = LatLonGrid.copyOf(citysize2010);
94 Pixel[] sizechange = new Pixel[props2.length];
95 for (int i=0; i < sizechange.length; ++i){
96     sizechange[i] = new Pixel(0, 0,0);
97 }
98 for (int i=0; i < citysize2010.getNumLat(); ++i){
99     for (int j=0; j < citysize2010.getNumLon(); ++j){
100         int sz1 = citysize1990.getValue(i,j);
101         int sz2 = citysize2010.getValue(i,j);
102         int percentChange = 0;
103         if (sz1 > 5 && sz2 > 5){ // reasonably big?
104             percentChange = (100 * (sz2 - sz1)) / sz1;
105             sizechange[ label2010.label.getValue(i,j) ] = new Pixel( i,j, percentChange);
106         }
107         changeInSize.setValue(i,j, percentChange);
108     }
109 }
110 KmlWriter.write(changeInSize, out, "sprawl_1990_2010", PngWriter.createCoolToWarmColormap());
111
112 // Print out the top cities
113 Arrays.sort(sizechange, new Comparator<Pixel>(){
114     @Override
115     public int compare(Pixel arg0, Pixel arg1) {
116         return arg0.getValue() - arg1.getValue();
117     }
118 });
119 for (int i=Math.max(0,sizechange.length-20); i < sizechange.length; ++i){
120     LatLon loc = citysize2010.getLocation(sizechange[i].getX(), sizechange[i].getY());
121     System.out.println( loc +
122         " : " + citysize1990.getValue(sizechange[i]) +
123         " to " + citysize2010.getValue(sizechange[i]) +
124         " " + getCityNear(loc)
125     );
126 }
127
128 // Shrunk?
129 System.out.println("Cities that have exhibited the least spatial growth:");
130 for (int i=0; i < Math.min(20,sizechange.length); ++i){
131     LatLon loc = citysize2010.getLocation(sizechange[i].getX(), sizechange[i].getY());
132     System.out.println( loc +
133         " : " + citysize1990.getValue(sizechange[i]) +
134         " to " + citysize2010.getValue(sizechange[i]) +
135         " " + getCityNear(loc)
136     );
137 }

```

```
138     }
```

6.111.2.2 static void edu.ou.asgbook.usage.Sprawl.main (String[] args) throws Exception [static]

```
179     {
180         boolean crop = false; // if false, on USA; if true, on NYC area
181         runOnPopDensity(crop);
182     }
```

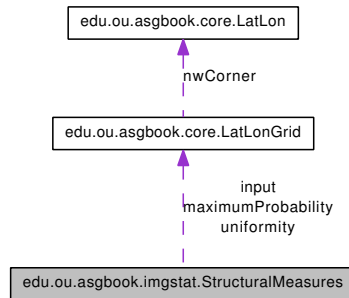
6.111.2.3 static void edu.ou.asgbook.usage.Sprawl.runOnPopDensity (boolean crop) throws Exception [static]

```
33     {
34         LatLonGrid popdensity1 = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA1990, new G
35         LatLonGrid popdensity2 = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new G
36
37         int x1, x2, x3, x4;
38         if (crop){
39             x1 = 900;
40             x2 = 2500;
41             x3 = 200;
42             x4 = 200;
43         } else {
44             LatLon nwCorner = new LatLon(60, -130);
45             LatLon seCorner = new LatLon(12, -52);
46             x1 = popdensity1.getRow(nwCorner);
47             x2 = popdensity1.getCol(nwCorner);
48             x3 = popdensity1.getRow(seCorner) - popdensity1.getRow(nwCorner);
49             x4 = popdensity1.getCol(seCorner) - popdensity1.getCol(nwCorner);
50         }
51         popdensity1 = popdensity1.crop(x1, x2, x3, x4);
52         popdensity2 = popdensity2.crop(x1, x2, x3, x4);
53
54         File out = OutputDirectory.getDefault("sprawl");
55
56         findSprawl(popdensity1, popdensity2, out);
57     }
```

6.112 edu.ou.asgbook.imgstat.StructuralMeasures Class Reference

Statistics computed in the neighborhood of a pixel.

Collaboration diagram for edu.ou.asgbook.imgstat.StructuralMeasures:



Public Member Functions

- [StructuralMeasures](#) ([LatLonGrid](#) input, int Nx, int Ny, int min, int incr, int bins)
- [LatLonGrid\[\]](#) [getUniformity](#) ()
- [LatLonGrid\[\]](#) [getMaximumProbability](#) ()

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.112.1 Detailed Description

Statistics computed in the neighborhood of a pixel.

Author:

valliappa.lakshmanan

6.112.2 Constructor & Destructor Documentation

- ##### 6.112.2.1 edu.ou.asgbook.imgstat.StructuralMeasures.StructuralMeasures
- ([LatLonGrid](#) input, int Nx, int Ny, int min, int incr, int bins)

```

28         this.hx = Nx/2;
29         this.hy = Ny/2;
30         this.input = input;
31         this.min = min;
32         this.incr = incr;
33         this.bins = bins;
34         for (int i=0; i < uniformity.length; ++i){
35             this.uniformity[i] = LatLonGrid.copyOf(input);
36             this.maximumProbability[i] = LatLonGrid.copyOf(input);
37         }
38         this.compute();
39     }

```

6.112.3 Member Function Documentation

6.112.3.1 [LatLonGrid](#) [] edu.ou.asgbook.imgstat.StructuralMeasures.getMaximumProbability ()

```

59                                     {
60         return maximumProbability;
61     }

```

6.112.3.2 [LatLonGrid](#) [] edu.ou.asgbook.imgstat.StructuralMeasures.getUniformity ()

```

55                                     {
56         return uniformity;
57     }

```

6.112.3.3 static void edu.ou.asgbook.imgstat.StructuralMeasures.main (String[] args) throws Exception [static]

```

63                                     {
64         // log-scaled population density
65         LatLonGrid popdensity = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GLCMStat());
66         popdensity = popdensity.crop(900, 2500, 200, 200);
67         File out = OutputDirectory.getDefault("glcmstat");
68
69         KmlWriter.write(popdensity, out, "popdensity", PngWriter.createCoolToWarmColormap());
70         StructuralMeasures stat = new StructuralMeasures(popdensity, 11, 11, 100, 500, 50);
71         for (int i=0; i < Direction.values().length; ++i){
72             KmlWriter.write(stat.getUniformity()[i], out, "uniformity_" + Direction.values()[i].name());
73         }
74         for (int i=0; i < Direction.values().length; ++i){
75             KmlWriter.write(stat.getMaximumProbability()[i], out, "maxprob_" + Direction.values()[i].name());
76         }
77     }

```

6.113 edu.ou.asgbook.dataset.SurfaceAlbedo Class Reference

Reads lambert-conformal ascii grid.

Static Public Member Functions

- static [LatLonGrid read](#) (Reader inputFile, int scaling)
- static [LatLonGrid read](#) (File file, int scaling) throws IOException
reads data from a File.
- static void [main](#) (String[] args) throws Exception

Static Public Attributes

- static File [CONUS](#) = new File("data/sfcalbedo/sfcalbedo.txt.gz")

6.113.1 Detailed Description

Reads lambert-conformal ascii grid.

Author:

v.lakshmanan

6.113.2 Member Function Documentation

6.113.2.1 static void edu.ou.asgbook.dataset.SurfaceAlbedo.main (String[] args) throws Exception [static]

```
138                                     {
139     // create output directory
140     File out = OutputDirectory.getDefault("sfcalbedo");
141
142     // read input
143     LatLonGrid conus = SurfaceAlbedo.read(SurfaceAlbedo.CONUS, 100);
144
145     // write out as image, for viewing
146     KmlWriter.write(conus, out, "sfcalbedo", PngWriter.createCoolToWarmColormap());
147 }
```

6.113.2.2 static [LatLonGrid](#) edu.ou.asgbook.dataset.SurfaceAlbedo.read (File *file*, int *scaling*) throws IOException [static]

reads data from a File.

The File can be gzipped or uncompressed.

```

126                                                     {
127         Reader f = null;
128         System.out.println("Reading " + file.getAbsolutePath());
129         if (file.getAbsolutePath().endsWith(".gz")) {
130             f = new InputStreamReader(new GZIPInputStream(new FileInputStream(
131                 file)));
132         } else {
133             f = new FileReader(file);
134         }
135         return read(f, scaling);
136     }

```

6.113.2.3 static [LatLonGrid](#) edu.ou.asgbook.dataset.SurfaceAlbedo.read (Reader *inputFile*, int *scaling*) [static]

```

34                                                     {
35         Scanner s = null;
36         try {
37             s = new Scanner(inputFile);
38
39             // read header
40             @SuppressWarnings("unused")
41             String junk;
42             junk = s.next(); String ellipsoid = s.next();
43             junk = s.next(); String projection = s.next();
44             if (! (ellipsoid.equals("WGS-84") && projection.equals("LAMBERT2SP"))){
45                 throw new IllegalArgumentException("Expect data to be in LAMBERT2SP and WGS-
46             }
47             junk = s.next(); double lat1 = s.nextDouble();
48             junk = s.next(); double lat2 = s.nextDouble();
49             junk = s.next(); double center_lat = s.nextDouble();
50             junk = s.next(); double center_lon = s.nextDouble();
51             junk = s.next(); double eastres = s.nextDouble(); // meters
52             junk = s.next(); double northres = s.nextDouble();
53             junk = s.next(); int nrows = s.nextInt();
54             junk = s.next(); int ncols = s.nextInt();
55
56             double center_northing = - nrows * 0.5 * northres;
57             double center_easting = ncols * 0.5 * eastres;
58
59             int missing = -999; // doesn't exist in the data
60
61             // read in data (in Lambert projection)
62             int[][] lamdata = new int[nrows][ncols];
63             for (int i=0; i < nrows; ++i){
64                 for (int j=0; j < ncols; ++j){

```



```

65         try {
66             double value = s.nextDouble();
67             lamdata[i][j] = (int)(0.5 + value * scaling);
68         } catch (Exception e){
69             lamdata[i][j] = missing;
70         }
71     }
72 }
73
74 // Find grid extent
75 LambertConformal2SP proj = new LambertConformal2SP(Ellipsoid.WGS84(), new LatLon(center_lat, center_lon));
76 double minlat = 180; double maxlat = -180;
77 double minlon = 180; double maxlon = -180;
78 for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
79     LambertConformal2SP.Coord lam = new LambertConformal2SP.Coord(-(i+0.5)*northres, (j+0.5)*lonres);
80     LatLon ll = proj.getLatLon(lam);
81     if ( ll.getLat() < minlat ) minlat = ll.getLat();
82     if ( ll.getLat() > maxlat ) maxlat = ll.getLat();
83     if ( ll.getLon() < minlon ) minlon = ll.getLon();
84     if ( ll.getLon() > maxlon ) maxlon = ll.getLon();
85 }
86 System.out.println("Grid extent: " + minlat + " " + minlon + " " + maxlat + " " + maxlon);
87
88 // best latres, lonres
89 int outrows = nrows;
90 int outcols = ncols;
91 double latres = (maxlat - minlat)/outrows;
92 double lonres = (maxlon - minlon)/outcols;
93 // lookup nearest neighbor in lat-lon space
94 int[][] lldata = new int[outrows][outcols];
95 for (int i=0; i < outrows; ++i){
96     double lat = maxlat - i * latres;
97     for (int j=0; j < outcols; ++j){
98         double lon = minlon + j * lonres;
99         LambertConformal2SP.Coord lam = proj.getLambert( new LatLon(lat,lon) );
100         double rowno = (0 - lam.northing)/northres;
101         rowno = nrows - rowno - 1; // row=0 is southmost row
102         double colno = (lam.easting - 0)/eastres;
103         // lldata[i][j] = Remapper.nearestNeighbor(rowno, colno, lamdata, missing);
104         lldata[i][j] = Remapper.bilinearInterpolation(rowno, colno, lamdata, missing);
105     }
106 }
107
108 return new LatLonGrid(lldata, missing, new LatLon(maxlat,minlon), latres, lonres);
109 } catch (Exception e){
110     System.err.println("Error reading file: " + e);
111     throw new IllegalArgumentException(e);
112 } finally {
113     if (s != null) {
114         try{
115             s.close();
116         } catch (Exception e){
117             // okay
118         }
119     }
120 }
121 }

```

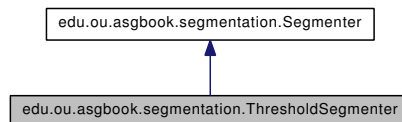
6.113.3 Member Data Documentation

6.113.3.1 File [edu.ou.asgbook.dataset.SurfaceAlbedo.CONUS](#) = new
File("data/sfcalbedo/sfcalbedo.txt.gz") [static]

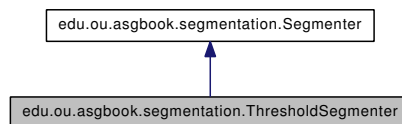
6.114 edu.ou.asgbook.segmentation.ThresholdSegmenter Class Reference

Simple object identification based on a single threshold.

Inheritance diagram for edu.ou.asgbook.segmentation.ThresholdSegmenter:



Collaboration diagram for edu.ou.asgbook.segmentation.ThresholdSegmenter:



Public Member Functions

- [ThresholdSegmenter](#) (int thresh)
- [LabelResult label](#) (LatLonGrid data)

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.114.1 Detailed Description

Simple object identification based on a single threshold.

Author:

v.lakshmanan

6.114.2 Constructor & Destructor Documentation

6.114.2.1 `edu.ou.asgbook.segmentation.ThresholdSegmenter.ThresholdSegmenter(int thresh)`

```

24                                     {
25         super();
26         this.thresh = thresh;
27     }
```

6.114.3 Member Function Documentation

6.114.3.1 `LabelResult edu.ou.asgbook.segmentation.ThresholdSegmenter.label(LatLonGrid data)` [virtual]

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

.. All pixels > thresh are part of an object.

Implements [edu.ou.asgbook.segmentation.Segmenter](#).

```

34                                     {
35         final int UNSET = 0;
36         int nrows = data.getNumLat();
37         int ncols = data.getNumLon();
38         LatLonGrid label = new LatLonGrid(nrows,ncols,0,data.getNwCorner(),data.getLatRes(),data.getLonRes());
39         // label.fill(UNSET); java default is to zero-out arrays
40         int regno = 0;
41         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
42             if ( data.getValue(i, j) > thresh && label.getValue(i, j) == UNSET ){
43                 ++regno;
44                 RegionGrowing.growRegion(i,j, data, thresh, label, regno);
45             }
46         }
47         System.out.println("Found " + (regno+1) + " objects");
48         return new LabelResult(label, regno);
49     }
```

6.114.3.2 `static void edu.ou.asgbook.segmentation.ThresholdSegmenter.main(String[] args)` throws Exception [static]

```

51                                     {
52         File out = OutputDirectory.getDefault("regiongrowing");
53
54         // data
55         LatLonGrid grid = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulation.Reader());
56         KmlWriter.write(grid, out, "orig", PngWriter.createCoolToWarmColormap());
57         KmlWriter.write(GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalPopulation.Reader()), out, "seg");
58     }
```

```
59         // global thresh
60         for (int thresh = 10; thresh <= 30; thresh += 10){
61             KmlWriter.write(new SimpleThresholder(thresh).threshold(grid), out, "thresh_"+thresh, PngWr
62             ThresholdSegmenter seg = new ThresholdSegmenter(thresh);
63             LatLonGrid label = seg.label(grid).label;
64             // label.setMissing(-1); // so background is present
65             KmlWriter.write(label, out, "cities_"+thresh, PngWriter.createRandomColormap());
66         }
67     }
```

6.115 edu.ou.asgbook.geocode.UsaZipcode Class Reference

Find the city for each zipcode in the USA.

Collaboration diagram for edu.ou.asgbook.geocode.UsaZipcode:



Public Member Functions

- Entry [getEntryClosestTo](#) ([LatLon](#) loc)

Static Public Member Functions

- static [UsaZipcode](#) [getInstance](#) ()
- static void [main](#) (String[] args)

Classes

- class [Entry](#)

6.115.1 Detailed Description

Find the city for each zipcode in the USA.

Author:

v.lakshmanan

6.115.2 Member Function Documentation

6.115.2.1 Entry edu.ou.asgbook.geocode.UsaZipcode.getEntryClosestTo ([LatLon](#) loc)

```

67                                     {
68         double mindistsq = 0.5*0.5; // within 50 km
69         Entry best = null;
70         for (Entry e : entries){
71             double dist_lat = e.location.getLat() - loc.getLat();
72             double dist_lon = e.location.getLon() - loc.getLon();
73             double dist_sq = dist_lat*dist_lat + dist_lon*dist_lon;
```

```
74         if ( dist_sq < mindistsq ){
75             mindistsq = dist_sq;
76             best = e;
77         }
78     }
79     return best;
80 }
```

6.115.2.2 static [UsaZipcode](#) edu.ou.asgbook.geocode.UsaZipcode.getInstance () [static]

```
82                                     {
83     return instance;
84 }
```

6.115.2.3 static void edu.ou.asgbook.geocode.UsaZipcode.main (String[] args) [static]

```
86                                     {
87     Entry e = UsaZipcode.getInstance().getEntryClosestTo(new LatLon(18.31,-66.06));
88     System.out.println(e);
89
90     e = UsaZipcode.getInstance().getEntryClosestTo(new LatLon(35.2,-97.4));
91     System.out.println(e);
92 }
```

6.116 edu.ou.asgbook.imgstat.VectorQuantizer Class Reference

Develops a quantization scheme using vector quantization.

Public Member Functions

- [VectorQuantizer](#) ([LatLonGrid](#)[] params, int K)
- int [getBinNumber](#) ([LatLonGrid](#)[] params, int x, int y)
- Vector [getCenterValue](#) (int bin_no)
- [LatLonGrid](#) [band](#) ([LatLonGrid](#)[] params)
replaces each pixel by the bin number it belongs to.
- Override String [toString](#) ()

Static Public Member Functions

- static [LatLonGrid](#)[] [normalize](#) ([LatLonGrid](#)[] params)
- static [LatLonGrid](#) [normalize](#) ([LatLonGrid](#) data)
The output grid ranges from 0 to 100.
- static void [main](#) (String[] args) throws Exception

Classes

- class **Vector**

6.116.1 Detailed Description

Develops a quantization scheme using vector quantization.

Author:

valliappa.lakshmanan

6.116.2 Constructor & Destructor Documentation

6.116.2.1 edu.ou.asgbook.imgstat.VectorQuantizer.VectorQuantizer ([LatLonGrid](#)[] *params*, int *K*)

Parameters:

params where to get the vectors from. These grids should be normalized.

K

```

58                                     {
59     int nrows = params[0].getNumLat();
60     int ncols = params[0].getNumLon();
61     // 1. initialize centroid with mean
62     centroids = new Vector[1];
63     centroids[0] = new Vector(params.length); // zero
64     for (int p=0; p < params.length; ++p){
65         int N = 0;
66         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
67             int val = params[p].getValue(i,j);
68             if ( val != params[p].getMissing() ){
69                 centroids[0].values[p] += val;
70                 ++N;
71             }
72         }
73         if (N > 0){
74             centroids[0].values[p] /= N;
75         }
76     }
77     System.out.println(this);
78     while ( centroids.length < K ){
79         // 2. split the centroids
80         final double epsilon = 0.1;
81         centroids = split(centroids, epsilon);
82         // System.out.println(this);
83         // 3. update centroid
84         centroids = computeCentroids(params);
85         // System.out.println(this);
86     }
87
88     System.out.println(this);
89 }
```

6.116.3 Member Function Documentation

6.116.3.1 [LatLonGrid](#) edu.ou.asgbook.imgstat.VectorQuantizer.band ([LatLonGrid](#)[] *params*)

replaces each pixel by the bin number it belongs to.

```

202                                     {
203     LatLonGrid result = LatLonGrid.copyOf(params[0]);
```

```

204         result.setMissing(0);
205         int nrows = result.getNumLat();
206         int ncols = result.getNumLon();
207         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
208             int bin_no = getBinNumber(params, i, j);
209             result.setValue(i, j, bin_no+1);
210         }
211         return result;
212     }

```

6.116.3.2 `int edu.ou.asgbook.imgstat.VectorQuantizer.getBinNumber` ([LatLonGrid\[\]](#) *params*, *int x*, *int y*)

```

183                                                     {
184         // closest centroid wins
185         int best = 0;
186         double mindist = centroids[0].computeDist(params, x, y);
187         for (int p=1; p < centroids.length; ++p){
188             double dist = centroids[p].computeDist(params, x, y);
189             if (dist < mindist){
190                 mindist = dist;
191                 best = p;
192             }
193         }
194         return best;
195     }

```

6.116.3.3 `Vector edu.ou.asgbook.imgstat.VectorQuantizer.getCenterValue` (*int bin_no*)

```

197                                                     {
198         return centroids[bin_no];
199     }

```

6.116.3.4 `static void edu.ou.asgbook.imgstat.VectorQuantizer.main` (*String[] args*) *throws Exception* [*static*]

```

223                                                     {
224         // create output directory
225         File outdir = OutputDirectory.getDefault("quantizer");
226
227         // read input
228         LatLonGrid conus = SurfaceAlbedo.read(SurfaceAlbedo.CONUS, 100);
229
230         // compute a few local and texture measures
231         LocalMeasures local = new LocalMeasures(conus, 11, 11);
232         LatLonGrid mean = local.getMean();
233         // LatLonGrid stdev = local.getStdDeviation();
234         Histogram hist = HistogramBinSelection.createBasedOnRange(conus);

```

```

235     StructuralMeasures texture = new StructuralMeasures(conus, 11, 11, hist.getMin(), hist.getMax());
236     LatLonGrid uniformity = texture.getUniformity()[0];
237
238     LatLonGrid[] params = new LatLonGrid[]{ conus, mean, uniformity };
239     params = normalize(params);
240
241     for (int k=4; k < 32; k *= 2){ // 4, 8, 16
242         VectorQuantizer quant = new VectorQuantizer(params, k);
243         LatLonGrid banded = quant.band(params);
244         KmlWriter.write(banded, outdir, "vecquant_" + k, PngWriter.createCoolToWarmColormap());
245     }
246 }

```

6.116.3.5 static [LatLonGrid](#) edu.ou.asgbook.imgstat.VectorQuantizer.normalize ([LatLonGrid data](#)) [static]

The output grid ranges from 0 to 100.

Parameters:

input

Returns:

```

104     {
105         LatLonGrid result = LatLonGrid.copyOf(data);
106         result.setMissing(-1);
107         // find range
108         int min = data.getMissing();
109         int max = data.getMissing();
110         int nrows = data.getNumLat();
111         int ncols = data.getNumLon();
112         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
113             int val = data.getValue(i,j);
114             if ( val != data.getMissing() ){
115                 if (min == data.getMissing() || val < min){
116                     min = val;
117                 }
118                 if (max == data.getMissing() || val > max){
119                     max = val;
120                 }
121             }
122         }
123         for (int i=0; i < nrows; ++i) for (int j=0; j < ncols; ++j){
124             int val = data.getValue(i,j);
125             if ( val != data.getMissing() ){
126                 result.setValue(i, j, (int)Math.round((100.0*(val-min))/(max-min) ));
127             } else {
128                 result.setValue(i, j, result.getMissing() );
129             }
130         }

```

```
131         return result;
132     }
```

6.116.3.6 static `LatLonGrid []` `edu.ou.asgbook.imgstat.VectorQuantizer.normalize (LatLonGrid[] params)` [static]

```
91
92     LatLonGrid[] norm = new LatLonGrid[params.length];
93     for (int i=0; i < params.length; ++i){
94         norm[i] = normalize(params[i]);
95     }
96     return norm;
97 }
```

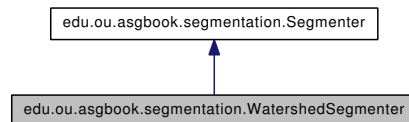
6.116.3.7 Override String `edu.ou.asgbook.imgstat.VectorQuantizer.toString ()`

```
215         {
216             StringBuilder sb = new StringBuilder("Centroids: ");
217             for (int p=0; p < centroids.length; ++p){
218                 sb.append(centroids[p]);
219             }
220             return sb.toString();
221         }
```

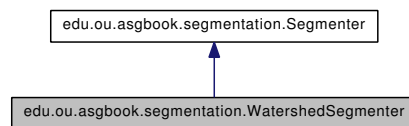
6.117 edu.ou.asgbook.segmentation.WatershedSegmenter Class Reference

Watershed approach of object identification.

Inheritance diagram for edu.ou.asgbook.segmentation.WatershedSegmenter:



Collaboration diagram for edu.ou.asgbook.segmentation.WatershedSegmenter:



Public Member Functions

- [WatershedSegmenter](#) (int thresh)
- [LabelResult label](#) (LatLonGrid data)

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.117.1 Detailed Description

Watershed approach of object identification.

Author:

valliappa.lakshmanan

6.117.2 Constructor & Destructor Documentation

6.117.2.1 `edu.ou.asgbook.segmentation.WatershedSegmenter.WatershedSegmenter(int thresh)`

```

30                                     {
31         super();
32         this.thresh = thresh;
33     }
```

6.117.3 Member Function Documentation

6.117.3.1 `LabelResult edu.ou.asgbook.segmentation.WatershedSegmenter.label(LatLonGrid data)` [virtual]

Creates a labeled grid where background pixels are set to 0 and labels for objects go 1,2,3.

.. All pixels > thresh are part of an object.

Implements [edu.ou.asgbook.segmentation.Segmenter](#).

```

39                                     {
40         return vincent_segment(data);
41     }
```

6.117.3.2 `static void edu.ou.asgbook.segmentation.WatershedSegmenter.main(String[] args)` throws Exception [static]

```

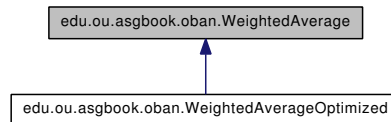
255                                     {
256         File out = OutputDirectory.getDefault("wshed");
257
258         // data
259         LatLonGrid grid = GlobalPopulation.read(GlobalPopulation.NORTHAMERICA, new GlobalP
260         KmlWriter.write(grid, out, "orig", PngWriter.createCoolToWarmColormap());
261
262         // int min_thresh = 0; int max_thresh = 400; int incr_thresh = 200; // log scaling
263         int min_thresh = 0; int max_thresh = 20; int incr_thresh = 10; // linear scaling
264
265         for (int thresh=min_thresh; thresh <= max_thresh; thresh += incr_thresh){
266             WatershedSegmenter seg = new WatershedSegmenter(thresh);
267             LatLonGrid label = seg.label(grid).label;
268             KmlWriter.write(label, out, "wsheds_"+thresh, PngWriter.createRandomColormap());
269         }
270
271         grid = new ConvolutionFilter(ConvolutionFilter.gauss(9, 9)).smooth(grid);
272         for (int thresh=min_thresh; thresh <= max_thresh; thresh += incr_thresh){
273             WatershedSegmenter seg = new WatershedSegmenter(thresh);
274             LatLonGrid label = seg.label(grid).label;
275             KmlWriter.write(label, out, "urbanareas_"+thresh, PngWriter.createRandomColorm
```

```
276         }  
277     }
```

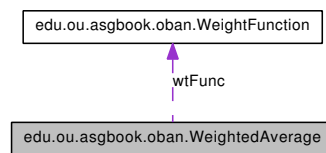
6.118 edu.ou.asgbook.oban.WeightedAverage Class Reference

Interpolation methods for point observations.

Inheritance diagram for edu.ou.asgbook.oban.WeightedAverage:



Collaboration diagram for edu.ou.asgbook.oban.WeightedAverage:



Public Member Functions

- [LatLonGrid analyze](#) ([PointObservations](#) data)
- [LatLonGrid analyze](#) ([PointObservations](#) data, int numPasses, int physicalMin, int physicalMax)
- [WeightedAverage](#) ([WeightFunction](#) wtFunc, double latres, double lonres, int minPoints)

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

Protected Attributes

- final [WeightFunction](#) wtFunc
- final double latres
- final double lonres
- final int minPoints

6.118.1 Detailed Description

Interpolation methods for point observations.

This is here only for illustration; you should use the [WeightedAverageOptimized](#)

Author:

Valliappa.Lakshmanan

6.118.2 Constructor & Destructor Documentation

6.118.2.1 `edu.ou.asgbook.oban.WeightedAverage.WeightedAverage` ([WeightFunction](#) *wtFunc*, double *latres*, double *lonres*, int *minPoints*)

```

96                                     {
97         this.wtFunc = wtFunc;
98         this.latres = latres;
99         this.lonres = lonres;
100         this.minPoints = minPoints;
101     }
```

6.118.3 Member Function Documentation

6.118.3.1 [LatLonGrid](#) `edu.ou.asgbook.oban.WeightedAverage.analyze` ([PointObservations](#) *data*, int *numPasses*, int *physicalMin*, int *physicalMax*)

```

59                                     {
60         LatLonGrid result = analyze(data); // pass #1
61         final PointObservations.ObservationPoint[] points = data.getPoints();
62         for (int pass=1; pass < numPasses; ++pass){
63             // find error at each point
64             PointObservations.ObservationPoint[] errors = new PointObservations.ObservationPoint[points.length];
65             for (int k=0; k < points.length; ++k){
66                 int a = points[k].getValue();
67                 int b = result.getValue(points[k]);
68                 int error = 0;
69                 if ( a != data.getMissing() && b != result.getMissing() ){
70                     error = a - b;
71                 }
72                 errors[k] = new PointObservations.ObservationPoint(points[k].getLat(), points[k].getLon(), error);
73             }
74             // create a grid of errors and add this to the original grid
75             LatLonGrid errGrid = analyze(new PointObservations(errors,data.getMissing()));
76             add( result, errGrid , physicalMin, physicalMax);
77         }
78         return result;
79     }
```

6.118.3.2 [LatLonGrid](#) [edu.ou.asgbook.oban.WeightedAverage.analyze](#) ([PointObservations](#) *data*)

Reimplemented in [edu.ou.asgbook.oban.WeightedAverageOptimized](#).

```

30                                     {
31     LatLonGrid grid = ObjectiveAnalysisUtils.createBoundingGrid(data, latres, lonres);
32     PointObservations.ObservationPoint[] points = data.getPoints();
33     for (int i=0; i < grid.getNumLat(); ++i){
34         for (int j=0; j < grid.getNumLon(); ++j){
35             LatLon gridpt = grid.getLocation(i, j);
36             double sum = 0;
37             double sumwt = 0;
38             int n = 0;
39             for (int k=0; k < points.length; ++k){
40                 if ( points[k].getValue() != data.getMissing() ){
41                     double wt = wtFunc.computeWt( points[k].getLat() - gridpt.getLat(),
42                     if ( wt > 0 ){
43                         sum += wt * points[k].getValue();
44                         sumwt += wt;
45                         ++n;
46                     }
47                 }
48             }
49             if ( n >= minPoints ){
50                 grid.setValue(i, j, (int) Math.round(sum/sumwt));
51             } else {
52                 grid.setValue(i, j, grid.getMissing());
53             }
54         }
55     }
56     return grid;
57 }
```

6.118.3.3 `static void` [edu.ou.asgbook.oban.WeightedAverage.main](#) ([String\[\]](#) *args*) throws [Exception](#) [`static`]

Reimplemented in [edu.ou.asgbook.oban.WeightedAverageOptimized](#).

```

103                                     {
104     PointObservations data = DailyRainfall.read(DailyRainfall.TN_Oct2010);
105
106     double meansep = ObjectiveAnalysisUtils.computeMeanDistance(data);
107     System.out.println("Objectively analyzing " + data.getPoints().length + " pts with
108     WeightFunction wtFunc = new CressmanWeighting(3*meansep);
109     WeightedAverage analyzer = new WeightedAverage(wtFunc, 0.01, 0.01, 1);
110
111     long startTime = System.nanoTime();
112     final int numPasses = 2;
113     LatLonGrid grid = analyzer.analyze(data, numPasses, 0, data.getMaxValue());
114     System.out.println("Took " + (System.nanoTime() - startTime)/(1000*1000.0*1000) + "
115 }
```

```
116         // write output
117         File out = OutputDirectory.getDefault("oban");
118         KmlWriter.write(grid, out, "Precip24H", PngWriter.createCoolToWarmColormap());
119     }
```

6.118.4 Member Data Documentation

6.118.4.1 final double [edu.ou.asgbook.oban.WeightedAverage.latres](#)
[protected]

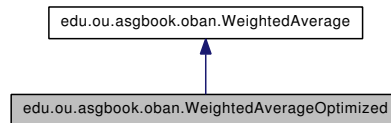
6.118.4.2 final double [edu.ou.asgbook.oban.WeightedAverage.lonres](#)
[protected]

6.118.4.3 final int [edu.ou.asgbook.oban.WeightedAverage.minPoints](#)
[protected]

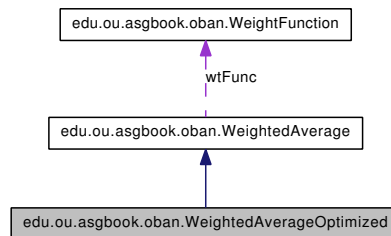
6.118.4.4 final [WeightFunction](#) [edu.ou.asgbook.oban.WeightedAverage.wtFunc](#)
[protected]

6.119 edu.ou.asgbook.oban.WeightedAverageOptimized Class Reference

Inheritance diagram for edu.ou.asgbook.oban.WeightedAverageOptimized:



Collaboration diagram for edu.ou.asgbook.oban.WeightedAverageOptimized:



Public Member Functions

- Override [LatLonGrid analyze](#) ([PointObservations](#) data)
- [WeightedAverageOptimized](#) ([WeightFunction wtFunc](#), double [latres](#), double [lonres](#), int [minPoints](#))

Static Public Member Functions

- static void [main](#) (String[] args) throws Exception

6.119.1 Detailed Description

Author:

Valliappa.Lakshmanan

6.119.2 Constructor & Destructor Documentation

6.119.2.1 edu.ou.asgbook.oban.WeightedAverageOptimized.WeightedAverageOptimized ([WeightFunction](#) wtFunc, double latres, double lonres, int minPoints)

```

92
93     super(wtFunc, latres, lonres, minPoints);
94     this.wtKernel = computeWeightKernel(wtFunc, latres, lonres);
95 }
```

6.119.3 Member Function Documentation

6.119.3.1 Override [LatLonGrid](#) edu.ou.asgbook.oban.WeightedAverageOptimized.analyze ([PointObservations](#) data)

Reimplemented from [edu.ou.asgbook.oban.WeightedAverage](#).

```

24                                     {
25     LatLonGrid grid = ObjectiveAnalysisUtils.createBoundingGrid(data, latres, lonres);
26     double[][] sum = new double[grid.getNumLat()][grid.getNumLon()];
27     double[][] sumwt = new double[grid.getNumLat()][grid.getNumLon()];
28     int[][] numpts = new int[grid.getNumLat()][grid.getNumLon()];
29     PointObservations.ObservationPoint[] points = data.getPoints();
30
31     final int half_rows = wtKernel.length / 2;
32     final int half_cols = wtKernel.length / 2;
33     for (int k=0; k < points.length; ++k){
34         final int row = grid.getRow(points[k]);
35         final int col = grid.getCol(points[k]);
36         if ( points[k].getValue() != data.getMissing() ){
37             for (int m=-half_rows; m <= half_rows; ++m){
38                 for (int n=-half_cols; n <= half_cols; ++n){
39                     final int i = row + m;
40                     final int j = col + n;
41                     final double wt = wtKernel[m+half_rows][n+half_cols];
42                     if( wt > 0 && grid.isValid(i, j)){
43                         sum[i][j] += points[k].getValue() * wt;
44                         sumwt[i][j] += wt;
45                         numpts[i][j] ++;
46                     }
47                 }
48             }
49         }
50     }
51
52     for (int i=0; i < grid.getNumLat(); ++i){
53         for (int j=0; j < grid.getNumLon(); ++j){
54             if ( numpts[i][j] >= minPoints ){
55                 grid.setValue(i, j, (int) Math.round(sum[i][j]/sumwt[i][j]));
56             } else {
```

```
57             grid.setValue(i, j, grid.getMissing());
58         }
59     }
60 }
61     return grid;
62 }
```

6.119.3.2 static void edu.ou.asgbook.oban.WeightedAverageOptimized.main (String[] args) throws Exception [static]

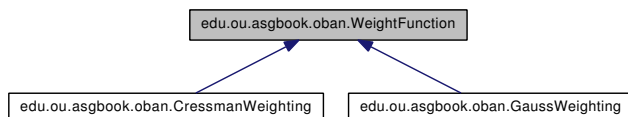
Reimplemented from [edu.ou.asgbook.oban.WeightedAverage](#).

```
97     {
98         File out = OutputDirectory.getDefault("obanopt");
99         run(DailyRainfall.read(DailyRainfall.TN_Oct2010), out, "Precip24H", 2);
100        run(MadisTemperature.read(MadisTemperature.TN_Oct2010), out, "Temperature_pass1", 1);
101        run(MadisTemperature.read(MadisTemperature.TN_Oct2010), out, "Temperature_pass2", 2);
102        run(MadisTemperature.read(MadisTemperature.TN_Oct2010), out, "Temperature_pass3", 3);
103        run(MadisTemperature.read(MadisTemperature.TN_Oct2010), out, "Temperature_pass10", 10);
104    }
```

6.120 edu.ou.asgbook.oban.WeightFunction Interface Reference

Used by [WeightedAverage](#).

Inheritance diagram for edu.ou.asgbook.oban.WeightFunction:



Public Member Functions

- abstract double [computeWt](#) (double latdist, double londist)
Subclasses implement a weighting function.

Static Public Attributes

- static final double [INVALID_WEIGHT](#) = -100.0

6.120.1 Detailed Description

Used by [WeightedAverage](#).

Author:

Valliappa.Lakshmanan

6.120.2 Member Function Documentation

6.120.2.1 abstract double edu.ou.asgbook.oban.WeightFunction.computeWt (double *latdist*, double *londist*) [pure virtual]

Subclasses implement a weighting function.

If -ve value is returned, then the point will be considered too far away and not used in weighting.

Implemented in [edu.ou.asgbook.oban.CressmanWeighting](#), and [edu.ou.asgbook.oban.GaussWeighting](#).

6.120.3 Member Data Documentation

6.120.3.1 final double [edu.ou.asgbook.oban.WeightFunction.INVALID_](#)
[WEIGHT](#) = -100.0 [static]

6.121 edu.ou.asgbook.dataset.WorldBankGDI Class Reference

Reads country-by-country Global development index from World Bank.

Public Types

- [LowIncome](#)
- enum [DevelopmentCategory](#) {
 [LowIncome](#), [LowerMiddleIncome](#), [UpperMiddleIncome](#), [HighIncomeNon-OECD](#),
 [HighIncomeOECD](#), [text](#) }

Static Public Member Functions

- static [CountryDI](#)[] [read](#) (File file) throws IOException
reads data from a CSV File.
- static [LatLonGrid](#) [readGrid](#) (File file) throws IOException
reads data from a ESRI grid file.
- static [DevelopmentLookup](#) [readAsMap](#) (File f) throws Exception
- static void [main](#) (String[] args) throws Exception

Static Public Attributes

- static File [WORLD_TABULAR](#) = new File("data/development/WDI_GDF_Country.csv")
- static File [WORLD_GRID](#) = new File("data/development/globaldevelopmentindex.txt.gz")

Classes

- class [CountryDI](#)
- class [DevelopmentLookup](#)

6.121.1 Detailed Description

Reads country-by-country Global development index from World Bank.

Author:

v.lakshmanan

6.121.2 Member Enumeration Documentation

6.121.2.1 enum [edu.ou.asgbook.dataset.WorldBankGDI.Development-Category](#)

Enumerator:

LowIncome
LowerMiddleIncome
UpperMiddleIncome
HighIncomeNonOECD
HighIncomeOECD
text

```

36                                     {
37         LowIncome, LowerMiddleIncome, UpperMiddleIncome, HighIncomeNonOECD, HighIncomeOECD,
38
39         public static DevelopmentCategory getInstance(String text){
40             if (text.equals("Low income")){
41                 return LowIncome;
42             } else if (text.equals("Lower middle income")){
43                 return LowerMiddleIncome;
44             } else if (text.equals("Upper middle income")){
45                 return UpperMiddleIncome;
46             } else if (text.equals("High income: nonOECD")){
47                 return HighIncomeNonOECD;
48             } else if (text.equals("High income: OECD")){
49                 return HighIncomeOECD;
50             }
51             throw new IllegalArgumentException("Unknown category: {" + text + "}");
52         }
53     }

```

6.121.3 Member Function Documentation

6.121.3.1 static void [edu.ou.asgbook.dataset.WorldBankGDI.main](#) (String[] args) throws Exception [static]

```

163                                     {
164         WorldBankGDI.CountryDI[] countries = WorldBankGDI.read(WorldBankGDI.WORLD_TABULAR);
165         for (WorldBankGDI.CountryDI c : countries){
166             System.out.println(c);
167         }
168
169         // some basic stats
170         System.out.println("Distribution of income levels is as follows:");
171         int[] count = new int[ DevelopmentCategory.values().length ];
172         for (WorldBankGDI.CountryDI c : countries){
173             count[ c.category.ordinal() ] ++;
174         }

```

```

175         for (int i=0; i < count.length; ++i){
176             WorldBankGDI.DevelopmentCategory cat = WorldBankGDI.DevelopmentCategory.values()[i];
177             System.out.println( cat + " " + count[i]);
178         }
179
180         // now combine with country polygons
181         DevelopmentLookup lookup = WorldBankGDI.readAsMap(WorldBankGDI.WORLD_TABULAR);
182         LatLonGrid countryGrid = CountryPolygons.readGrid(CountryPolygons.WORLD_GRID);
183         DevelopmentCategory[] categories = lookup.getDevelopmentCategories(CountryPolygons.readKml(Cou
184         for (int i=0; i < countryGrid.getNumLat(); ++i){
185             for (int j=0; j < countryGrid.getNumLon(); ++j){
186                 int countryIndex = countryGrid.getValue(i, j);
187                 if ( countryIndex >= 0 ){
188                     int devCategory = categories[countryIndex].ordinal();
189                     countryGrid.setValue(i, j, devCategory);
190                 }
191             }
192         }
193         File out = OutputDirectory.getDefault("countries");
194         KmlWriter.write(countryGrid, out, "gdi", PngWriter.createCoolToWarmColormap());
195         EsriGrid.write(countryGrid, out, "globaldevelopmentindex.txt.gz");
196         EsriGrid.write(countryGrid, WorldBankGDI.WORLD_GRID);
197     }

```

6.121.3.2 static CountryDI [] edu.ou.asgbook.dataset.WorldBankGDI.read (File file) throws IOException [static]

reads data from a CSV File.

The File can be gzipped or uncompressed.

```

71                                     {
72         Reader f = null;
73         System.out.println("Reading " + file.getAbsolutePath());
74         if (file.getAbsolutePath().endsWith(".gz")) {
75             f = new InputStreamReader(new GZIPInputStream(new FileInputStream(
76                 file)));
77         } else {
78             f = new FileReader(file);
79         }
80         return read(f);
81     }

```

6.121.3.3 static DevelopmentLookup edu.ou.asgbook.dataset.World- BankGDI.readAsMap (File f) throws Exception [static]

```

155                                     {
156         DevelopmentLookup result = new DevelopmentLookup();
157         for (CountryDI c : read(f)){
158             result.add(c);

```

```
159         }
160         return result;
161     }
```

6.121.3.4 static [LatLonGrid](#) edu.ou.asgbook.dataset.WorldBankGDI.readGrid
(File *file*) throws IOException [static]

reads data from a ESRI grid file.

The File can be gzipped or uncompressed.

```
86
87         return EsriGrid.read(file, new LinearScaling(1));
88     }
```

6.121.4 Member Data Documentation

6.121.4.1 File [edu.ou.asgbook.dataset.WorldBankGDI.WORLD_GRID](#) =
new File("data/development/globaldevelopmentindex.txt.gz")
[static]

6.121.4.2 File [edu.ou.asgbook.dataset.WorldBankGDI.WORLD_TABULAR](#) =
new File("data/development/WDI_GDF_Country.csv") [static]

Chapter 7

Automating Analysis of Spatial Grids File Documentation

7.1 AlignAndDifference.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.AlignAndDifference](#)
Aligns two grids and then computes their difference.

7.2 AlignmentEstimator.java File Reference

Namespaces

- namespace [edu.ou.asgbook.transforms](#)

Classes

- class [edu.ou.asgbook.transforms.AlignmentEstimator](#)
Estimate the degree of spatial displacement between two similar grids.

7.3 Assignment4.java File Reference

Namespaces

- namespace [edu.ou.asgbook.usage](#)

Classes

- class [edu.ou.asgbook.usage.Assignment4](#)
(1) Find optimal threshold on log(pop) Find distance of every grid point to a point < thresh Find optimal threshold of distance values Threshold image to keep only values < threshold

7.4 BoundingBox.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rasterization](#)

Classes

- class [edu.ou.asgbook.rasterization.BoundingBox](#)

A rectangular bounding box of a polygon.

7.5 CatmullRom.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rasterization](#)

Classes

- class [edu.ou.asgbook.rasterization.CatmullRom](#)
A Catmull-Rom spline, a local spline.
- class [edu.ou.asgbook.rasterization.CatmullRom.XtoY](#)

7.6 CityCategories.java File Reference

Namespaces

- namespace [edu.ou.asgbook.datamining](#)

Classes

- class [edu.ou.asgbook.datamining.CityCategories](#)
Obtains city data for clustering.

7.7 CityGdiModels.java File Reference

Namespaces

- namespace [edu.ou.asgbook.datamining](#)

Classes

- class [edu.ou.asgbook.datamining.CityGdiModels](#)
Applies different data mining models to each city.

7.8 ContiguityEnhancedKMeansSegmenter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.ContiguityEnhancedKMeansSegmenter](#)
Objects consist of pixels that are grown from initial centers using K-means.
- class **edu.ou.asgbook.segmentation.ContiguityEnhancedKMeansSegmenter.Cluster**

7.9 ConvolutionFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.ConvolutionFilter](#)
Convolve an image by a window.

7.10 CountryPolygons.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

Classes

- class [edu.ou.asgbook.dataset.CountryPolygons](#)
Reads country-by-country coordinates from a KML placemarks file.
- class **edu.ou.asgbook.dataset.CountryPolygons.Country**

7.11 CressmanWeighting.java File Reference

Namespaces

- namespace [edu.ou.asgbook.oban](#)

Classes

- class [edu.ou.asgbook.oban.CressmanWeighting](#)
An interpolation method that uses $1/r^2$.

7.12 CrossCorrelation.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.CrossCorrelation](#)
Estimates motion using cross-correlation.

7.13 CumulativeDistributionFunction.java File Reference

Namespaces

- namespace [edu.ou.asgbook.histogram](#)

Classes

- class [edu.ou.asgbook.histogram.CumulativeDistributionFunction](#)
Forms a CDF from a [Histogram](#).

7.14 DailyRainfall.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

Classes

- class [edu.ou.asgbook.dataset.DailyRainfall](#)
Reads the ASCII precipitation data available at <http://madis-data.noaa.gov/public/hydrodumpguest.html>.

7.15 DataSimulator.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rbf](#)

Classes

- class [edu.ou.asgbook.rbf.DataSimulator](#)
Simulates RBF data to be fit.

7.16 DataTransform.java File Reference

Namespaces

- namespace [edu.ou.asgbook.linearity](#)

Classes

- class [edu.ou.asgbook.linearity.DataTransform](#)
Transform pixel values, usually to meet linearity requirements.

7.17 Differencer.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.Differencer](#)
Just computes a pixel-by-pixel difference.

7.18 DilateErodeFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.DilateErodeFilter](#)
Carries out paired dilation followed by erosion for filling in holes.

7.19 DilationFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.DilationFilter](#)
Expands entities by taking a local maximum.

7.20 EdgeBased.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.EdgeBased](#)
Estimates motion based on the displacement of edges.

7.21 Ellipsoid.java File Reference

Namespaces

- namespace [edu.ou.asgbook.projections](#)

Classes

- class [edu.ou.asgbook.projections.Ellipsoid](#)
An ellipsoidal approximation to the earth.

7.22 EnhancedWatershedSegmenter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter](#)
Enhanced watershed segmentation following Lakshmanan, Hondl and Rabin.
- class **edu.ou.asgbook.segmentation.EnhancedWatershedSegmenter.Glob**

7.23 Entropy.java File Reference

Namespaces

- namespace [edu.ou.asgbook.histogram](#)

Classes

- class [edu.ou.asgbook.histogram.Entropy](#)
Compute entropy from a histogram.

7.24 ErodeDilateFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.ErodeDilateFilter](#)
Carries out paired erosion followed by dilation for denoising.

7.25 ErosionFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.ErosionFilter](#)
Reduces the size of entities by taking a local minimum.

7.26 EsriGrid.java File Reference

Namespaces

- namespace [edu.ou.asgbook.io](#)

Classes

- class [edu.ou.asgbook.io.EsriGrid](#)

Read an ESRI grid.

7.27 EuclideanDT.java File Reference

Namespaces

- namespace [edu.ou.asgbook.distance](#)

Classes

- interface [edu.ou.asgbook.distance.EuclideanDT](#)

7.28 EuclideanDTPropagation.java File Reference

Namespaces

- namespace [edu.ou.asgbook.distance](#)

Classes

- class [edu.ou.asgbook.distance.EuclideanDTPropagation](#)
Implementation of Euclidean distance that updates the distance instead of computing it afresh each time.

7.29 EuclideanDTRecursivePropagation.java File Reference

Namespaces

- namespace [edu.ou.asgbook.distance](#)

Classes

- class [edu.ou.asgbook.distance.EuclideanDTRecursivePropagation](#)

Note that this class is only for illustrative purposes.

7.30 EuclideanDTSaito.java File Reference

Namespaces

- namespace [edu.ou.asgbook.distance](#)

Classes

- class [edu.ou.asgbook.distance.EuclideanDTSaito](#)

The Saito technique of computing the distance transform by calculating in the two directions separately.

7.31 FFT.java File Reference

Namespaces

- namespace [edu.ou.asgbook.transforms](#)

Classes

- class [edu.ou.asgbook.transforms.FFT](#)
FFT based on Sedgewick and Wayne.
- class [edu.ou.asgbook.transforms.FFT.Complex](#)

7.32 FFT2D.java File Reference

Namespaces

- namespace [edu.ou.asgbook.transforms](#)

Classes

- class [edu.ou.asgbook.transforms.FFT2D](#)
Two-dimensional [FFT](#).

7.33 FFTBandpassFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.transforms](#)

Classes

- class [edu.ou.asgbook.transforms.FFTBandpassFilter](#)
Removes noise (high frequencies) and the gross signal (low frequencies).

7.34 FFTConvolutionFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.transforms](#)

Classes

- class [edu.ou.asgbook.transforms.FFTConvolutionFilter](#)

An optimization for convolution using FFTs.

7.35 FuzzyCandidateMarket.java File Reference

Namespaces

- namespace [edu.ou.asgbook.datamining](#)

Classes

- class [edu.ou.asgbook.datamining.FuzzyCandidateMarket](#)

Uses heuristic rules to choose the next market to enter.

7.36 FuzzyLogic.java File Reference

Namespaces

- namespace [edu.ou.asgbook.datamining](#)

Classes

- class [edu.ou.asgbook.datamining.FuzzyLogic](#)
A simple fuzzy logic engine.
- class **edu.ou.asgbook.datamining.FuzzyLogic.Fuzzy**
- interface [edu.ou.asgbook.datamining.FuzzyLogic.Rule](#)
- class **edu.ou.asgbook.datamining.FuzzyLogic.IsHigh**
- class **edu.ou.asgbook.datamining.FuzzyLogic.IsLow**
- class **edu.ou.asgbook.datamining.FuzzyLogic.IsAbout**
- class **edu.ou.asgbook.datamining.FuzzyLogic.Aggregate**

Simply applies an equal weighting to all of the values.

7.37 GaussianComponent.java File Reference

Namespaces

- namespace [edu.ou.asgbook.gmm](#)

Classes

- class [edu.ou.asgbook.gmm.GaussianComponent](#)
Component of a Gaussian Mixture Model.
- class **edu.ou.asgbook.gmm.GaussianComponent.Expectation**

7.38 GaussianMixtureModel.java File Reference

Namespaces

- namespace [edu.ou.asgbook.gmm](#)

Classes

- class [edu.ou.asgbook.gmm.GaussianMixtureModel](#)
A parametric approximation of a spatial grid as a sum of Gaussians.

7.39 GaussWeighting.java File Reference

Namespaces

- namespace [edu.ou.asgbook.oban](#)

Classes

- class [edu.ou.asgbook.oban.GaussWeighting](#)
An interpolation method that uses $\exp(-1/r^2)$.

7.40 GdiPattern.java File Reference

Namespaces

- namespace [edu.ou.asgbook.datamining](#)

Classes

- class [edu.ou.asgbook.datamining.GdiPattern](#)

The training pattern for each city.

7.41 GlobalPopulation.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

Classes

- class [edu.ou.asgbook.dataset.GlobalPopulation](#)
Reads the ASCII population data available at <http://sedac.ciesin.columbia.edu/gpw>.
- class **edu.ou.asgbook.dataset.GlobalPopulation.LogScaling**
- class **edu.ou.asgbook.dataset.GlobalPopulation.LinearScaling**

7.42 GraylevelCooccurrenceMatrix.java File Reference

Namespaces

- namespace [edu.ou.asgbook.imgstat](#)

Classes

- class [edu.ou.asgbook.imgstat.GraylevelCooccurrenceMatrix](#)
Computes texture properties from a GLCM.

7.43 HilditchSkeletonization.java File Reference

Namespaces

- namespace [edu.ou.asgbook.thinning](#)

Classes

- class [edu.ou.asgbook.thinning.HilditchSkeletonization](#)
Hilditch method of skeletonizing a grid.
- class **edu.ou.asgbook.thinning.HilditchSkeletonization.State**

7.44 Histogram.java File Reference

Namespaces

- namespace [edu.ou.asgbook.histogram](#)

Classes

- class [edu.ou.asgbook.histogram.Histogram](#)
A histogram is an empirical probability distribution.

7.45 HistogramBinSelection.java File Reference

Namespaces

- namespace [edu.ou.asgbook.histogram](#)

Classes

- class [edu.ou.asgbook.histogram.HistogramBinSelection](#)
Tries out different values for the number of bins and replaces each pixel value by the center of its bin.

7.46 HornSchunk.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.HornSchunk](#)
Horn-Schunk optical flow method of motion estimation.

7.47 HoughTransform.java File Reference

Namespaces

- namespace [edu.ou.asgbook.transforms](#)

Classes

- class [edu.ou.asgbook.transforms.HoughTransform](#)
Finds lines in image.
- class [edu.ou.asgbook.transforms.HoughTransform.Line](#)

7.48 HungarianAssigner.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.HungarianAssigner](#)
Optimal assignment algorithm.
- class **edu.ou.asgbook.motion.HungarianAssigner.HungarianMatch**

7.49 HybridTracker.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.HybridTracker](#)
Estimates motion by finding cross-correlation of objects in one frame to the pixels in the previous frame.
- class **edu.ou.asgbook.motion.HybridTracker.Centroid**

7.50 HysteresisSegmenter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.HysteresisSegmenter](#)

Objects consist of pixels that are $> thresh2$ but have at least one pixel $> thresh1$.

7.51 Inverter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.Inverter](#)
at every pixel, replaces its value (val) by (A - val)

7.52 KalmanFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.KalmanFilter](#)
For the time smoothing of motion vectors.

7.53 KmlWriter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.io](#)

Classes

- class [edu.ou.asgbook.io.KmlWriter](#)

Writes data out in KML form, for display in Google Earth or similar program.

7.54 LabelResult.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.LabelResult](#)
Result of segmentation.

7.55 LambertConformal2SP.java File Reference

Namespaces

- namespace [edu.ou.asgbook.projections](#)

Classes

- class [edu.ou.asgbook.projections.LambertConformal2SP](#)
Lambert Conformation 2 Standard Parallels map projection.
- class **edu.ou.asgbook.projections.LambertConformal2SP.Coord**

7.56 LatLon.java File Reference

Namespaces

- namespace [edu.ou.asgbook.core](#)

Classes

- class [edu.ou.asgbook.core.LatLon](#)

A point on the earth's surface typically in WGS84.

7.57 LatLonGrid.java File Reference

Namespaces

- namespace [edu.ou.asgbook.core](#)

Classes

- class [edu.ou.asgbook.core.LatLonGrid](#)
A geospatial grid of data in equilat equilon coordinates typically in WGS84 ellipsoid.

7.58 LevelSet.java File Reference

Namespaces

- namespace [edu.ou.asgbook.core](#)

Classes

- class [edu.ou.asgbook.core.LevelSet](#)

A representation of a spatial grid as a set of levels.

7.59 Line.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rasterization](#)

Classes

- class [edu.ou.asgbook.rasterization.Line](#)
A line that connects two points on the earth's surface.

7.60 LinearityVerifier.java File Reference

Namespaces

- namespace [edu.ou.asgbook.linearity](#)

Classes

- class [edu.ou.asgbook.linearity.LinearityVerifier](#)
Given a 2D array of points, reports error measures of assuming linearity.
- interface **edu.ou.asgbook.linearity.LinearityVerifier.DataSelector**
- class **edu.ou.asgbook.linearity.LinearityVerifier.NotMissing**
- class **edu.ou.asgbook.linearity.LinearityVerifier.InRange**

7.61 LinearScaling.java File Reference

Namespaces

- namespace [edu.ou.asgbook.linearity](#)

Classes

- class [edu.ou.asgbook.linearity.LinearScaling](#)
Scales pixel values as Ax .

7.62 LocalMeasures.java File Reference

Namespaces

- namespace [edu.ou.asgbook.imgstat](#)

Classes

- class [edu.ou.asgbook.imgstat.LocalMeasures](#)
Statistics computed in the neighborhood of a pixel.

7.63 LoGEdgeFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.LoGEdgeFilter](#)
Laplacian of a Gaussian edge filter.

7.64 LogScaling.java File Reference

Namespaces

- namespace [edu.ou.asgbook.linearity](#)

Classes

- class [edu.ou.asgbook.linearity.LogScaling](#)

Transforms pixel values as $\log(x)$.

7.65 MadisTemperature.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

Classes

- class [edu.ou.asgbook.dataset.MadisTemperature](#)
Reads the ASCII temperature data available at <http://madis-data.noaa.gov/public/sfcdumpguest.html>.

7.66 MatchedFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.MatchedFilter](#)

Convolve an image by a window that is akin to the features we want to extract.

7.67 MaxValueFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.MaxValueFilter](#)
Finds the highest value pixel in the image.
- class **edu.ou.asgbook.filters.MaxValueFilter.Result**

7.68 MedialAxisSkeletonization.java File Reference

Namespaces

- namespace [edu.ou.asgbook.thinning](#)

Classes

- class [edu.ou.asgbook.thinning.MedialAxisSkeletonization](#)

The MAT method of skeletonizing a grid.

7.69 MedianFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.MedianFilter](#)
A smoothing operation that involves replacing a pixel by the local median.

7.70 MotionEstimator.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- interface [edu.ou.asgbook.motion.MotionEstimator](#)

7.71 MultiFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.MultiFilter](#)
Carries out multiple operations.

7.72 MultiscaleKMeansSegmenter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter](#)
Quantizes image into K levels, then does multiscale segmentation Does not implement the pruning techniques discussed in the paper.
- class **edu.ou.asgbook.segmentation.MultiscaleKMeansSegmenter.Cluster**

7.73 NHighest.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.NHighest](#)
Finds the N highest valued-pixels in image.

7.74 NHighestLevelSetImpl.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.NHighestLevelSetImpl](#)
Finds the N highest valued-pixels in image using a levelset implementation.

7.75 NighttimeLights.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

Classes

- class [edu.ou.asgbook.dataset.NighttimeLights](#)
Reads night-time lights data in ESRI grid format.

7.76 ObjectiveAnalysisUtils.java File Reference

Namespaces

- namespace [edu.ou.asgbook.oban](#)

Classes

- class [edu.ou.asgbook.oban.ObjectiveAnalysisUtils](#)
Utility functions for objective analysis.

7.77 ObjectTracker.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.ObjectTracker](#)
Estimates motion based on assigning objects in one frame to objects in the previous frame.
- class **edu.ou.asgbook.motion.ObjectTracker.CentroidDistance**
- class **edu.ou.asgbook.motion.ObjectTracker.GreedyAssignment**
- class **edu.ou.asgbook.motion.ObjectTracker.SimpleSegmenter**
- interface [edu.ou.asgbook.motion.ObjectTracker.CostEstimator](#)
- interface [edu.ou.asgbook.motion.ObjectTracker.Assigner](#)

7.78 **OrientedEllipseFilter.java** File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.OrientedEllipseFilter](#)

A non-isotropic smoothing filter.

7.79 OtsuThresholdSelector.java File Reference

Namespaces

- namespace [edu.ou.asgbook.histogram](#)

Classes

- class [edu.ou.asgbook.histogram.OtsuThresholdSelector](#)

Uses Otsu (1979) to select optimal threshold.

7.80 OutputDirectory.java File Reference

Namespaces

- namespace [edu.ou.asgbook.io](#)

Classes

- class [edu.ou.asgbook.io.OutputDirectory](#)

Change this to change the output directory that is used by all the main().

7.81 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.core](#)

7.82 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.datamining](#)

7.83 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

7.84 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.distance](#)

7.85 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

7.86 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.geocode](#)

7.87 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.gmm](#)

7.88 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.histogram](#)

7.89 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.imgstat](#)

7.90 package-info.java File Reference

Namespaces

- namespace edu.ou.asgbook.io

7.91 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.linearity](#)

7.92 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

7.93 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.oban](#)

7.94 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.projections](#)

7.95 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rasterization](#)

7.96 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rbf](#)

7.97 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

7.98 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.thinning](#)

7.99 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.transforms](#)

7.100 package-info.java File Reference

Namespaces

- namespace [edu.ou.asgbook.usage](#)

7.101 Pair.java File Reference

Namespaces

- namespace [edu.ou.asgbook.core](#)

Classes

- class [edu.ou.asgbook.core.Pair< X, Y >](#)
An utility class so that methods can return two objects.

7.102 PhaseCorrelation.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.PhaseCorrelation](#)
Estimate motion based on FFT.

7.103 Pixel.java File Reference

Namespaces

- namespace [edu.ou.asgbook.core](#)

Classes

- class [edu.ou.asgbook.core.Pixel](#)
A grid point in a spatial grid consists of a location and value.
- class **edu.ou.asgbook.core.Pixel.CompareLocation**
- class **edu.ou.asgbook.core.Pixel.CompareValue**

7.104 PngWriter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.io](#)

Classes

- class [edu.ou.asgbook.io.PngWriter](#)
Writes a spatial grid out as PNG file.

7.105 PointObservations.java File Reference

Namespaces

- namespace [edu.ou.asgbook.core](#)

Classes

- class [edu.ou.asgbook.core.PointObservations](#)
A set of observation points.
- class **edu.ou.asgbook.core.PointObservations.ObservationPoint**
An observation point is a value at a given location.

7.106 Polygon.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rasterization](#)

Classes

- class [edu.ou.asgbook.rasterization.Polygon](#)
A polygon consisting of straight edges along the earth's surface.

7.107 PrimaryCities.java File Reference

Namespaces

- namespace [edu.ou.asgbook.datamining](#)

Classes

- class [edu.ou.asgbook.datamining.PrimaryCities](#)
Identifies the primary cities in each country.

7.108 ProjectionPursuit.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rbf](#)

Classes

- class [edu.ou.asgbook.rbf.ProjectionPursuit](#)
Approximates a spatial grid by a RBF when nothing is known beyond the number of Gaussians desired.
- interface [edu.ou.asgbook.rbf.ProjectionPursuit.NextRBF](#)
- class **edu.ou.asgbook.rbf.ProjectionPursuit.SpatialMean**
- class **edu.ou.asgbook.rbf.ProjectionPursuit.LocalMax**

7.109 PyramidalCrossCorrelation.java File Reference

Namespaces

- namespace [edu.ou.asgbook.motion](#)

Classes

- class [edu.ou.asgbook.motion.PyramidalCrossCorrelation](#)
Cross-correlation at multiple resolutions.

7.110 Quantizer.java File Reference

Namespaces

- namespace [edu.ou.asgbook.imgstat](#)

Classes

- class [edu.ou.asgbook.imgstat.Quantizer](#)

Develops a quantization scheme using histogram equalization.

7.111 QuickSelect.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.QuickSelect](#)

From Numerical Recipes, a fast way to find the k th smallest item in a list Useful to implement rank filters.

7.112 RadialBasisFunction.java File Reference

Namespaces

- namespace [edu.ou.asgbook.rbf](#)

Classes

- class [edu.ou.asgbook.rbf.RadialBasisFunction](#)

Finds best fit of a spatial grid to a sum of Gaussians when the centers and sigmas of the Gaussians are known.

7.113 RegionGrowing.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.RegionGrowing](#)
Common object-identification utility.

7.114 RegionProperty.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.RegionProperty](#)
Properties of a region such as geometric (centroid, area, etc) and physical (based on other grid values).
- class **edu.ou.asgbook.segmentation.RegionProperty.Ellipse**

7.115 Remapper.java File Reference

Namespaces

- namespace [edu.ou.asgbook.projections](#)

Classes

- class [edu.ou.asgbook.projections.Remapper](#)
Utilities to remap one map projection to another.

7.116 SaturateFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.SaturateFilter](#)

Sets all values $< MIN$ to MIN and all values $> MAX$ to MAX .

7.117 ScalarStatistic.java File Reference

Namespaces

- namespace [edu.ou.asgbook.core](#)

Classes

- class [edu.ou.asgbook.core.ScalarStatistic](#)
A utility class to compute mean, variance of a streaming set of inputs.

7.118 Segmenter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- interface [edu.ou.asgbook.segmentation.Segmenter](#)
Object identification technique.

7.119 SeparableConvolutionFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.SeparableConvolutionFilter](#)
An optimized convolution filter.

7.120 SeviriInfraredTemperature.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

Classes

- class [edu.ou.asgbook.dataset.SeviriInfraredTemperature](#)

To read binary dump output from WDSS-II (<http://www.wdssii.org/>).

7.121 SimpleThresholder.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.SimpleThresholder](#)
Replace pixel values with 1 or 0 depending on whether they are above or below a single threshold.

7.122 SnakeActiveContour.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.SnakeActiveContour](#)
Active contour method of identifying objects.
- class **[edu.ou.asgbook.segmentation.SnakeActiveContour.SnakeNode](#)**
- class [edu.ou.asgbook.segmentation.SnakeActiveContour.Snake](#)

7.123 SobelEdgeFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.SobelEdgeFilter](#)
Find edges in a grid.

7.124 SpatialFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- interface [edu.ou.asgbook.filters.SpatialFilter](#)

7.125 SpeckleFilter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.filters](#)

Classes

- class [edu.ou.asgbook.filters.SpeckleFilter](#)
Denoising filter that removes speckle.

7.126 Sprawl.java File Reference

Namespaces

- namespace [edu.ou.asgbook.usage](#)

Classes

- class [edu.ou.asgbook.usage.Sprawl](#)

Solution to a classroom assignment to identify regions of urban sprawl from the population density data.

7.127 StructuralMeasures.java File Reference

Namespaces

- namespace [edu.ou.asgbook.imgstat](#)

Classes

- class [edu.ou.asgbook.imgstat.StructuralMeasures](#)
Statistics computed in the neighborhood of a pixel.

7.128 SurfaceAlbedo.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

Classes

- class [edu.ou.asgbook.dataset.SurfaceAlbedo](#)
Reads lambert-conformal ascii grid.

7.129 ThresholdSegmenter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.ThresholdSegmenter](#)
Simple object identification based on a single threshold.

7.130 UsaZipcode.java File Reference

Namespaces

- namespace [edu.ou.asgbook.geocode](#)

Classes

- class [edu.ou.asgbook.geocode.UsaZipcode](#)
Find the city for each zipcode in the USA.
- class **edu.ou.asgbook.geocode.UsaZipcode.Entry**

7.131 VectorQuantizer.java File Reference

Namespaces

- namespace [edu.ou.asgbook.imgstat](#)

Classes

- class [edu.ou.asgbook.imgstat.VectorQuantizer](#)
Develops a quantization scheme using vector quantization.
- class **edu.ou.asgbook.imgstat.VectorQuantizer.Vector**

7.132 WatershedSegmenter.java File Reference

Namespaces

- namespace [edu.ou.asgbook.segmentation](#)

Classes

- class [edu.ou.asgbook.segmentation.WatershedSegmenter](#)
Watershed approach of object identification.

7.133 WeightedAverage.java File Reference

Namespaces

- namespace [edu.ou.asgbook.oban](#)

Classes

- class [edu.ou.asgbook.oban.WeightedAverage](#)
Interpolation methods for point observations.

7.134 WeightedAverageOptimized.java File Reference

Namespaces

- namespace [edu.ou.asgbook.oban](#)

Classes

- class [edu.ou.asgbook.oban.WeightedAverageOptimized](#)

7.135 WeightFunction.java File Reference

Namespaces

- namespace [edu.ou.asgbook.oban](#)

Classes

- interface [edu.ou.asgbook.oban.WeightFunction](#)
Used by [WeightedAverage](#).

7.136 WorldBankGDI.java File Reference

Namespaces

- namespace [edu.ou.asgbook.dataset](#)

Classes

- class [edu.ou.asgbook.dataset.WorldBankGDI](#)
Reads country-by-country Global development index from World Bank.
- class **edu.ou.asgbook.dataset.WorldBankGDI.CountryDI**
- class **edu.ou.asgbook.dataset.WorldBankGDI.DevelopmentLookup**

Index

- [static initializer]
 - edu::ou::asgbook::dataset::Seviri-InfraredTemperature, [338](#)
- add
 - edu::ou::asgbook::core::LatLonGrid, [211](#)
- AlignAndDifference.java, [393](#)
- AlignmentEstimator
 - edu::ou::asgbook::transforms::AlignmentEstimator, [47](#)
- AlignmentEstimator.java, [394](#)
- analyze
 - edu::ou::asgbook::oban::WeightedAverage, [381](#)
 - edu::ou::asgbook::oban::WeightedAverageOptimized, [385](#)
- apply
 - edu::ou::asgbook::datamining::FuzzyLogic::Rule, [141](#)
- applyDecisionTree
 - edu::ou::asgbook::datamining::CityGdiModels, [61](#)
- applyLinearModel
 - edu::ou::asgbook::datamining::CityGdiModels, [62](#)
- applyNeuralNetwork
 - edu::ou::asgbook::datamining::CityGdiModels, [62](#)
- asLatLonGrid
 - edu::ou::asgbook::dataset::CountryPolygons, [72](#)
- asPixels
 - edu::ou::asgbook::core::LatLonGrid, [212](#)
- Assignment4.java, [395](#)
- band
 - edu::ou::asgbook::histogram::HistogramBinSelection, [169](#)
 - edu::ou::asgbook::imgstat::Quantizer, [311](#)
 - edu::ou::asgbook::imgstat::VectorQuantizer, [373](#)
- bilinearInterpolation
 - edu::ou::asgbook::projections::Remapper, [325](#)
- BoundingBox
 - edu::ou::asgbook::rasterization::BoundingBox, [53](#)
- BoundingBox.java, [396](#)
- boxcar
 - edu::ou::asgbook::filters::ConvolutionFilter, [68](#)
 - edu::ou::asgbook::filters::SeparableConvolutionFilter, [335](#)
- calcProb
 - edu::ou::asgbook::histogram::Histogram, [166](#)
- CatmullRom.java, [397](#)
- CityCategories.java, [398](#)
- CityGdiModels.java, [399](#)
- compareTo
 - edu::ou::asgbook::core::Pixel, [286](#)
- compute
 - edu::ou::asgbook::motion::AlignAndDifference, [46](#)
 - edu::ou::asgbook::motion::CrossCorrelation, [80](#)
 - edu::ou::asgbook::motion::Differencer, [92](#)
 - edu::ou::asgbook::motion::EdgeBased, [100](#)

- edu::ou::asgbook::motion::Horn-Schunk, [175](#)
- edu::ou::asgbook::motion::HybridTracker, [185](#)
- edu::ou::asgbook::motion::MotionEstimator, [250](#)
- edu::ou::asgbook::motion::ObjectTracker, [267](#)
- edu::ou::asgbook::motion::PhaseCorrelation, [282](#)
- edu::ou::asgbook::motion::PyramidalCrossCorrelation, [308](#)
- edu::ou::asgbook::segmentation::RegionProperty, [320](#)
- computeCentroid
 - edu::ou::asgbook::motion::PhaseCorrelation, [283](#)
 - edu::ou::asgbook::transforms::AlignmentEstimator, [49](#)
- computeCost
 - edu::ou::asgbook::motion::ObjectTracker::CostEstimator, [271](#)
- computeDifferenceMoment
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, [160](#)
- computeEnergy
 - edu::ou::asgbook::segmentation::SnakeActiveContour::Snake, [348](#)
- computeEntropy
 - edu::ou::asgbook::histogram::Entropy, [107](#)
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, [160](#)
- computeMaximumProbability
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, [160](#)
- computeMeanDistance
 - edu::ou::asgbook::oban::ObjectiveAnalysisUtils, [264](#)
- computeProbabilityDensityAt
 - edu::ou::asgbook::gmm::GaussianComponent, [144](#), [145](#)
- computeUniformity
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, [160](#)
- computeWt
 - edu::ou::asgbook::oban::CressmanWeighting, [77](#)
 - edu::ou::asgbook::oban::GaussWeighting, [151](#)
 - edu::ou::asgbook::oban::WeightFunction, [387](#)
- contains
 - edu::ou::asgbook::rasterization::BoundingBox, [54](#)
 - edu::ou::asgbook::rasterization::Polygon, [297](#)
- ContiguityEnhancedKMeansSegmenter
 - edu::ou::asgbook::segmentation::ContiguityEnhancedKMeansSegmenter, [64](#)
 - ContiguityEnhancedKMeansSegmenter.java, [400](#)
- CORONUS
 - edu::ou::asgbook::dataset::SurfaceAlbedo, [366](#)
- ConvolutionFilter
 - edu::ou::asgbook::filters::ConvolutionFilter, [68](#)
 - ConvolutionFilter.java, [401](#)
- convolve
 - edu::ou::asgbook::filters::ConvolutionFilter, [68](#)
 - edu::ou::asgbook::transforms::FTTBandpassFilter, [132](#)
 - edu::ou::asgbook::transforms::FFTConvolutionFilter, [135](#)
- copyOf
 - edu::ou::asgbook::core::LatLonGrid, [212](#)
 - edu::ou::asgbook::rasterization::BoundingBox, [54](#)
- CountryPolygons.java, [402](#)
- createBasedOnNumSamples
 - edu::ou::asgbook::histogram::HistogramBinSelection, [170](#)
- createBasedOnRange
 - edu::ou::asgbook::histogram::HistogramBinSelection, [170](#)
- createBasedOnStdDev
 - edu::ou::asgbook::histogram::HistogramBinSelection, [171](#)

- createBoundingBox
 - edu::ou::asgbook::oban::Objective-AnalysisUtils, [265](#)
- createCoolToWarmColormap
 - edu::ou::asgbook::io::PngWriter, [290](#)
- createGrayScaleColormap
 - edu::ou::asgbook::io::PngWriter, [290](#)
- createHighestResolution
 - edu::ou::asgbook::histogram::HistogramBinSelection, [172](#)
- createHotColormap
 - edu::ou::asgbook::io::PngWriter, [291](#)
- createRandomColormap
 - edu::ou::asgbook::io::PngWriter, [291](#)
- CressmanWeighting
 - edu::ou::asgbook::oban::CressmanWeighting, [77](#)
- CressmanWeighting.java, [403](#)
- crop
 - edu::ou::asgbook::core::LatLonGrid, [212](#)
- CrossCorrelation
 - edu::ou::asgbook::motion::CrossCorrelation, [80](#)
- CrossCorrelation.java, [404](#)
- CumulativeDistributionFunction
 - edu::ou::asgbook::histogram::CumulativeDistributionFunction, [83](#)
- CumulativeDistributionFunction.java, [405](#)
- DailyRainfall.java, [406](#)
- DataSimulator.java, [407](#)
- DataTransform.java, [408](#)
- debugWrite
 - edu::ou::asgbook::io::KmlWriter, [197](#)
- DEFAULT_TRANSPARENCY
 - edu::ou::asgbook::io::PngWriter, [293](#)
- DevelopmentCategory
 - edu::ou::asgbook::dataset::WorldBankGDI, [390](#)
- Differencer.java, [409](#)
- dilate
 - edu::ou::asgbook::filters::DilationFilter, [97](#)
- DilateErodeFilter
 - edu::ou::asgbook::filters::DilateErodeFilter, [95](#)
- DilateErodeFilter.java, [410](#)
- DilationFilter
 - edu::ou::asgbook::filters::DilationFilter, [97](#)
- DilationFilter.java, [411](#)
- Direction
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, [159](#)
- distanceInKms
 - edu::ou::asgbook::core::LatLon, [207](#)
- EASTWARD
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, [159](#)
- eccsq
 - edu::ou::asgbook::projections::Ellipsoid, [103](#)
- EdgeBased.java, [412](#)
- edgeFilter
 - edu::ou::asgbook::filters::LoGEdgeFilter, [233](#), [234](#)
 - edu::ou::asgbook::filters::SobelEdgeFilter, [351](#)
- edu, [17](#)
- edu.ou, [18](#)
- edu.ou.asgbook, [19](#)
- edu.ou.asgbook.core, [21](#)
- edu.ou.asgbook.datamining, [22](#)
- edu.ou.asgbook.dataset, [23](#)
- edu.ou.asgbook.distance, [24](#)
- edu.ou.asgbook.filters, [25](#)
- edu.ou.asgbook.geocode, [27](#)
- edu.ou.asgbook.gmm, [28](#)
- edu.ou.asgbook.histogram, [29](#)
- edu.ou.asgbook.imgstat, [30](#)
- edu.ou.asgbook.io, [31](#)
- edu.ou.asgbook.linearity, [32](#)

- edu.ou.asgbook.motion, 33
- edu.ou.asgbook.oban, 35
- edu.ou.asgbook.projections, 36
- edu.ou.asgbook.rasterization, 37
- edu.ou.asgbook.rbf, 38
- edu.ou.asgbook.segmentation, 39
- edu.ou.asgbook.thinning, 41
- edu.ou.asgbook.transforms, 42
- edu.ou.asgbook.usage, 43
- edu::ou::asgbook::core::LatLon, 206
- edu::ou::asgbook::core::LatLon
 - distanceInKms, 207
 - getLat, 207
 - getLon, 207
 - LatLon, 206
 - main, 207
 - toString, 207
- edu::ou::asgbook::core::LatLonGrid, 209
- edu::ou::asgbook::core::LatLonGrid
 - add, 211
 - asPixels, 212
 - copyOf, 212
 - crop, 212
 - fill, 213
 - getCol, 213
 - getData, 213
 - getLatRes, 213
 - getLocation, 213, 214
 - getLonRes, 214
 - getMissing, 214
 - getNumLat, 214
 - getNumLon, 214
 - getNwCorner, 215
 - getPixel, 215
 - getRow, 215
 - getSeCorner, 215
 - getValue, 215, 216
 - isValid, 216
 - LatLonGrid, 210, 211
 - longitudewrap, 216
 - remapTo, 217
 - replace, 217
 - setMissing, 217
 - setValue, 217
- edu::ou::asgbook::core::LevelSet, 219
- edu::ou::asgbook::core::Pair< X, Y >, 280
 - first, 280
 - Pair, 280
 - second, 280
- edu::ou::asgbook::core::Pixel, 285
 - compareTo, 286
 - equals, 286
 - getCol, 286
 - getDistanceSquared, 287
 - getRow, 287
 - getValue, 287
 - getX, 287
 - getY, 287
 - main, 288
 - Pixel, 286
 - toString, 288
- edu::ou::asgbook::core::PointObservations, 294
- edu::ou::asgbook::core::Point-
 - Observations
 - getMaxValue, 294
 - getMissing, 295
 - getPoints, 295
 - PointObservations, 294
- edu::ou::asgbook::core::ScalarStatistic, 329
- edu::ou::asgbook::core::ScalarStatistic
 - getMax, 329
 - getMean, 329
 - getMin, 329
 - getNumSamples, 330
 - getStdDeviation, 330
 - getVariance, 330
 - toString, 330
 - update, 330, 331
- edu::ou::asgbook::datamining::CityCategories, 59
- edu::ou::asgbook::datamining::City-
 - Categories
 - main, 59
- edu::ou::asgbook::datamining::CityGdiModels, 61
- edu::ou::asgbook::datamining::CityGdi-
 - Models
 - applyDecisionTree, 61

- applyLinearModel, 62
 - applyNeuralNetwork, 62
 - findPatterns, 62
 - main, 63
- edu::ou::asgbook::datamining::FuzzyCandidateMarket
 - 137
- edu::ou::asgbook::datamining::Fuzzy-
 - CandidateMarket
 - FuzzyCandidateMarket, 137
 - isGoodCandidate, 138
 - main, 138
- edu::ou::asgbook::datamining::FuzzyLogic,
 - 140
- edu::ou::asgbook::datamining::FuzzyLogic::Rule,
 - 141
- edu::ou::asgbook::datamining::Fuzzy-
 - Logic::Rule
 - apply, 141
- edu::ou::asgbook::datamining::GdiPattern,
 - 152
- edu::ou::asgbook::datamining::Gdi-
 - Pattern
 - findTrainingPattern, 152
 - main, 152
 - toString, 153
 - write, 154
- edu::ou::asgbook::datamining::PrimaryCities,
 - 300
- edu::ou::asgbook::datamining::Primary-
 - Cities
 - findPrimaryCities, 300
 - main, 301
- edu::ou::asgbook::dataset::CountryPolygons,
 - 72
- edu::ou::asgbook::dataset::Country-
 - Polygons
 - asLatLonGrid, 72
 - main, 73
 - readGrid, 74
 - readKml, 74
 - WORLD_GRID, 75
 - WORLD_KML, 75
- edu::ou::asgbook::dataset::DailyRainfall,
 - 86
- edu::ou::asgbook::dataset::DailyRainfall
 - main, 86
- read, 86
 - SuppressWarnings, 87
 - TN_Oct2010, 87
- edu::ou::asgbook::dataset::GlobalPopulation,
 - 155
- edu::ou::asgbook::dataset::Global-
 - Population
 - main, 155
 - NORTHAMERICA, 157
 - NORTHAMERICA1990, 157
 - read, 156
 - WORLD, 157
- edu::ou::asgbook::dataset::MadisTemperature,
 - 237
- edu::ou::asgbook::dataset::Madis-
 - Temperature
 - main, 237
 - read, 237
 - SuppressWarnings, 238
 - TN_Oct2010, 238
- edu::ou::asgbook::dataset::NighttimeLights,
 - 262
- edu::ou::asgbook::dataset::Nighttime-
 - Lights
 - main, 262
 - read, 262
 - WORLD, 263
- edu::ou::asgbook::dataset::SeviriInfraredTemperature,
 - 338
- edu::ou::asgbook::dataset::SeviriInfrared-
 - Temperature
 - [static initializer], 338
 - main, 338
 - read, 339
 - readAll, 339
- edu::ou::asgbook::dataset::SurfaceAlbedo,
 - 363
- edu::ou::asgbook::dataset::Surface-
 - Albedo
 - CONUS, 366
 - main, 363
 - read, 363, 364
- edu::ou::asgbook::dataset::WorldBankGDI,
 - 389
- HighIncomeNonOECD, 390
 - HighIncomeOECD, 390

- LowerMiddleIncome, 390
- LowIncome, 390
- text, 390
- UpperMiddleIncome, 390
- edu::ou::asgbook::dataset::WorldBank-
GDI
- DevelopmentCategory, 390
- main, 390
- read, 391
- readAsMap, 391
- readGrid, 392
- WORLD_GRID, 392
- WORLD_TABULAR, 392
- edu::ou::asgbook::distance::EuclideanDT, 117
- edu::ou::asgbook::distance::EuclideanDT
getDistanceTransform, 117
- edu::ou::asgbook::distance::EuclideanDTPropagation, 118
- edu::ou::asgbook::distance::Euclidean-
DTPropagation
- getDistanceTransform, 119
- main, 119
- edu::ou::asgbook::distance::EuclideanDTRecursivePropagation, 120
- edu::ou::asgbook::distance::Euclidean-
DTRecursivePropagation
- getDistanceTransform, 121
- main, 121
- edu::ou::asgbook::distance::EuclideanDTSaito, 122
- edu::ou::asgbook::distance::Euclidean-
DTSaito
- getDistanceTransform, 123
- main, 123
- edu::ou::asgbook::filters::ConvolutionFilter, 67
- edu::ou::asgbook::filters::Convolution-
Filter
- boxcar, 68
- ConvolutionFilter, 68
- convolve, 68
- filter, 69
- gauss, 69
- getFilterNumCols, 70
- getFilterNumRows, 70
- main, 70
- smooth, 71
- edu::ou::asgbook::filters::DilateErodeFilter, 94
- edu::ou::asgbook::filters::DilateErode-
Filter
- DilateErodeFilter, 95
- main, 95
- edu::ou::asgbook::filters::DilationFilter, 96
- edu::ou::asgbook::filters::DilationFilter
dilate, 97
- DilationFilter, 97
- filter, 97
- main, 97
- edu::ou::asgbook::filters::ErodeDilateFilter, 109
- edu::ou::asgbook::filters::ErodeDilate-
Filter
- ErodeDilateFilter, 110
- main, 110
- edu::ou::asgbook::filters::ErosionFilter, 111
- edu::ou::asgbook::filters::ErosionFilter
erode, 112
- ErosionFilter, 112
- filter, 112
- main, 112
- edu::ou::asgbook::filters::Inverter, 191
- filter, 192
- invert, 192
- Inverter, 191
- edu::ou::asgbook::filters::LoEdgeFilter, 232
- edu::ou::asgbook::filters::LoEdgeFilter
edgeFilter, 233, 234
- filter, 234
- LoEdgeFilter, 233
- main, 234
- edu::ou::asgbook::filters::MatchedFilter, 239
- edu::ou::asgbook::filters::MatchedFilter
filter, 240
- main, 240
- match, 241
- MatchedFilter, 240

- edu::ou::asgbook::filters::MaxValueFilter, 243
- edu::ou::asgbook::filters::MaxValueFilter
 - findHighestValued, 243
 - main, 244
- edu::ou::asgbook::filters::MedianFilter, 247
- edu::ou::asgbook::filters::MedianFilter
 - filter, 248
 - main, 248
 - MedianFilter, 248
 - smooth, 248
- edu::ou::asgbook::filters::MultiFilter, 251
- edu::ou::asgbook::filters::MultiFilter
 - filter, 252
 - MultiFilter, 251
- edu::ou::asgbook::filters::NHighest, 257
 - findHighestValued, 257
 - main, 258
 - NHighest, 257
- edu::ou::asgbook::filters::NHighestLevelSetImpl, 260
- edu::ou::asgbook::filters::NHighestLevelSetImpl
 - findHighestValued, 260
 - main, 261
 - NHighestLevelSetImpl, 260
- edu::ou::asgbook::filters::OrientedEllipseFilter, 272
- edu::ou::asgbook::filters::OrientedEllipseFilter
 - filter, 273
 - main, 273
 - OrientedEllipseFilter, 273
 - smooth, 274
- edu::ou::asgbook::filters::QuickSelect, 314
- edu::ou::asgbook::filters::QuickSelect
 - kth_element, 314, 315
- edu::ou::asgbook::filters::SaturateFilter, 327
- edu::ou::asgbook::filters::SaturateFilter
 - filter, 328
 - saturate, 328
 - SaturateFilter, 327
- edu::ou::asgbook::filters::SeparableConvolutionFilter, 334
- edu::ou::asgbook::filters::SeparableConvolutionFilter
 - boxcar, 335
 - filter, 335
 - gauss, 335, 336
 - main, 336
 - SeparableConvolutionFilter, 335
 - smooth, 336
- edu::ou::asgbook::filters::SimpleThresholder, 341
- edu::ou::asgbook::filters::SimpleThresholder
 - filter, 342
 - main, 342
 - SimpleThresholder, 342
 - threshold, 342
- edu::ou::asgbook::filters::SobelEdgeFilter, 350
- edu::ou::asgbook::filters::SobelEdgeFilter
 - edgeFilter, 351
 - filter, 351
 - main, 352
 - saturate, 352
 - SobelEdgeFilter, 351
- edu::ou::asgbook::filters::SpatialFilter, 353
- edu::ou::asgbook::filters::SpatialFilter
 - filter, 354
- edu::ou::asgbook::filters::SpeckleFilter, 355
- edu::ou::asgbook::filters::SpeckleFilter
 - filter, 356
 - main, 356
 - SpeckleFilter, 356
 - speckleFilter, 356
- edu::ou::asgbook::geocode::UsaZipcode, 370
- edu::ou::asgbook::geocode::UsaZipcode
 - getEntryClosestTo, 370
 - getInstance, 371
 - main, 371
- edu::ou::asgbook::gmm::GaussianComponent, 142

- edu::ou::asgbook::gmm::Gaussian-Component
 - computeProbabilityDensityAt, [144](#), [145](#)
 - GaussianComponent, [143](#)
 - getCx, [145](#)
 - getCy, [145](#)
 - getSigmax, [145](#)
 - getSigmaxy, [145](#)
 - getSigmay, [145](#)
 - getWeight, [145](#)
 - isValid, [146](#)
 - toString, [146](#)
- edu::ou::asgbook::gmm::GaussianMixtureModel, [147](#)
- edu::ou::asgbook::gmm::Gaussian-MixtureModel
 - GaussianMixtureModel, [147](#)
 - getMixture, [148](#)
 - main, [148](#)
 - setValueInGrid, [148](#)
- edu::ou::asgbook::histogram::CumulativeDistributionFunction, [83](#)
- edu::ou::asgbook::histogram::Cumulative-DistributionFunction
 - CumulativeDistributionFunction, [83](#)
 - getIncr, [84](#)
 - getMin, [84](#)
 - getProb, [84](#)
 - main, [84](#)
 - toString, [85](#)
- edu::ou::asgbook::histogram::Entropy, [107](#)
 - computeEntropy, [107](#)
 - main, [107](#)
- edu::ou::asgbook::histogram::Histogram, [165](#)
 - calcProb, [166](#)
 - getBinNumber, [166](#)
 - getCenterValue, [166](#)
 - getHist, [167](#)
 - getIncr, [167](#)
 - getMin, [167](#)
 - getNumBins, [167](#)
 - Histogram, [165](#)
 - main, [167](#)
 - toString, [168](#)
 - update, [168](#)
- edu::ou::asgbook::histogram::HistogramBinSelection, [169](#)
- edu::ou::asgbook::histogram::Histogram-BinSelection
 - band, [169](#)
 - createBasedOnNumSamples, [170](#)
 - createBasedOnRange, [170](#)
 - createBasedOnStdDev, [171](#)
 - createHighestResolution, [172](#)
 - main, [172](#)
- edu::ou::asgbook::histogram::OtsuThresholdSelector, [275](#)
- edu::ou::asgbook::histogram::Otsu-ThresholdSelector
 - getOptimalThreshold, [276](#)
 - main, [276](#)
 - OtsuThresholdSelector, [275](#)
- edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, [158](#)
- edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix
 - EASTWARD, [159](#)
 - NORTHEAST, [159](#)
 - SOUTHEAST, [159](#)
 - SOUTHWARD, [159](#)
 - xadd, [159](#)
 - yadd, [159](#)
- edu::ou::asgbook::imgstat::Graylevel-CooccurrenceMatrix
 - computeDifferenceMoment, [160](#)
 - computeEntropy, [160](#)
 - computeMaximumProbability, [160](#)
 - computeUniformity, [160](#)
 - Direction, [159](#)
 - findBin, [161](#)
 - GraylevelCooccurrenceMatrix, [159](#)
- edu::ou::asgbook::imgstat::LocalMeasures, [229](#)
- edu::ou::asgbook::imgstat::Local-Measures
 - getMax, [230](#)
 - getMean, [230](#)
 - getMin, [230](#)
 - getStdDeviation, [230](#)
 - LocalMeasures, [230](#)
 - main, [230](#)

- edu::ou::asgbook::imgstat::Quantizer,
 - 310
 - band, 311
 - getBinNumber, 311
 - getCenterValue, 312
 - main, 312
 - Quantizer, 310
 - toString, 312
- edu::ou::asgbook::imgstat::StructuralMeasures,
 - 361
- edu::ou::asgbook::imgstat::Structural-
 Measures
 - getMaximumProbability, 362
 - getUniformity, 362
 - main, 362
 - StructuralMeasures, 361
- edu::ou::asgbook::imgstat::VectorQuantizer,
 - 372
- edu::ou::asgbook::imgstat::Vector-
 Quantizer
 - band, 373
 - getBinNumber, 374
 - getCenterValue, 374
 - main, 374
 - normalize, 375, 376
 - toString, 376
 - VectorQuantizer, 373
- edu::ou::asgbook::io::EsriGrid, 114
- edu::ou::asgbook::io::EsriGrid
 - read, 114, 115
 - write, 116
- edu::ou::asgbook::io::KmlWriter, 197
- edu::ou::asgbook::io::KmlWriter
 - debugWrite, 197
 - main, 197
 - write, 198, 199
- edu::ou::asgbook::io::OutputDirectory,
 - 278
- edu::ou::asgbook::io::OutputDirectory
 - getDefault, 278
 - relative, 278
 - temporary, 278
- edu::ou::asgbook::io::PngWriter, 289
- edu::ou::asgbook::io::PngWriter
 - createCoolToWarmColormap, 290
 - createGrayScaleColormap, 290
 - createHotColormap, 291
 - createRandomColormap, 291
 - DEFAULT_TRANSPARENCY, 293
 - main, 292
 - writeAutoScaled, 292
 - writeScaled, 293
- edu::ou::asgbook::linearity::DataTransform,
 - 90
- edu::ou::asgbook::linearity::Data-
 Transform
 - inverse, 90
 - transform, 90
 - transformAndRoundoff, 90
- edu::ou::asgbook::linearity::LinearityVerifier,
 - 224
- edu::ou::asgbook::linearity::Linearity-
 Verifier
 - main, 224
 - verify, 225
- edu::ou::asgbook::linearity::LinearScaling,
 - 227
- edu::ou::asgbook::linearity::Linear-
 Scaling
 - inverse, 228
 - LinearScaling, 228
 - transform, 228
- edu::ou::asgbook::linearity::LogScaling,
 - 235
- edu::ou::asgbook::linearity::LogScaling
 - inverse, 236
 - LogScaling, 235
 - transform, 236
- edu::ou::asgbook::motion::AlignAndDifference,
 - 45
- edu::ou::asgbook::motion::AlignAnd-
 Difference
 - compute, 46
- edu::ou::asgbook::motion::CrossCorrelation,
 - 79
- edu::ou::asgbook::motion::Cross-
 Correlation
 - compute, 80
 - CrossCorrelation, 80
 - main, 81
 - test, 82

- edu::ou::asgbook::motion::Differencer,
 - 92
 - compute, 92
 - main, 92
- edu::ou::asgbook::motion::EdgeBased,
 - 99
- edu::ou::asgbook::motion::EdgeBased
 - compute, 100
 - main, 100
- edu::ou::asgbook::motion::HornSchunk,
 - 174
- edu::ou::asgbook::motion::HornSchunk
 - compute, 175
 - main, 176
 - MOT_SCALE, 177
- edu::ou::asgbook::motion::HungarianAssigner,
 - 181
- edu::ou::asgbook::motion::Hungarian-Assigner
 - getAssignments, 182
 - main, 183
- edu::ou::asgbook::motion::HybridTracker,
 - 184
- edu::ou::asgbook::motion::HybridTracker
 - compute, 185
 - HybridTracker, 185
 - main, 187
- edu::ou::asgbook::motion::KalmanFilter,
 - 193
- edu::ou::asgbook::motion::KalmanFilter
 - getRateOfChange, 194
 - getValue, 194
 - init, 194
 - KalmanFilter, 193
 - main, 194
 - update, 195
 - updated, 196
- edu::ou::asgbook::motion::MotionEstimator,
 - 250
- edu::ou::asgbook::motion::Motion-Estimator
 - compute, 250
- edu::ou::asgbook::motion::ObjectTracker,
 - 266
- edu::ou::asgbook::motion::ObjectTracker
 - compute, 267
 - main, 268
 - ObjectTracker, 267
- edu::ou::asgbook::motion::ObjectTracker::Assigner,
 - 270
- edu::ou::asgbook::motion::Object-Tracker::Assigner
 - getAssignments, 270
- edu::ou::asgbook::motion::ObjectTracker::CostEstimator,
 - 271
- edu::ou::asgbook::motion::Object-Tracker::CostEstimator
 - computeCost, 271
 - getMaxCost, 271
- edu::ou::asgbook::motion::PhaseCorrelation,
 - 281
- edu::ou::asgbook::motion::Phase-Correlation
 - compute, 282
 - computeCentroid, 283
 - main, 283
 - MAXU, 284
 - MAXV, 284
 - PhaseCorrelation, 281
 - test, 283
- edu::ou::asgbook::motion::PyramidalCrossCorrelation,
 - 307
- edu::ou::asgbook::motion::Pyramidal-CrossCorrelation
 - compute, 308
 - main, 309
 - PyramidalCrossCorrelation, 308
- edu::ou::asgbook::oban::CressmanWeighting,
 - 76
- edu::ou::asgbook::oban::Cressman-Weighting
 - computeWt, 77
 - CressmanWeighting, 77
 - main, 77
- edu::ou::asgbook::oban::GaussWeighting,
 - 150
- edu::ou::asgbook::oban::GaussWeighting
 - computeWt, 151
 - GaussWeighting, 151
 - main, 151
- edu::ou::asgbook::oban::ObjectiveAnalysisUtils,
 - 264

- edu::ou::asgbook::oban::Objective-AnalysisUtils
 - computeMeanDistance, [264](#)
 - createBoundingBox, [265](#)
- edu::ou::asgbook::oban::WeightedAverage, [380](#)
- edu::ou::asgbook::oban::Weighted-Average
 - analyze, [381](#)
 - latres, [383](#)
 - lonres, [383](#)
 - main, [382](#)
 - minPoints, [383](#)
 - WeightedAverage, [381](#)
 - wtFunc, [383](#)
- edu::ou::asgbook::oban::WeightedAverageOptimized, [384](#)
- edu::ou::asgbook::oban::Weighted-AverageOptimized
 - analyze, [385](#)
 - main, [386](#)
 - WeightedAverageOptimized, [385](#)
- edu::ou::asgbook::oban::WeightFunction, [387](#)
- edu::ou::asgbook::oban::WeightFunction
 - computeWt, [387](#)
 - INVALID_WEIGHT, [388](#)
- edu::ou::asgbook::projections::Ellipsoid, [102](#)
 - eccsq, [103](#)
 - Ellipsoid, [102](#)
 - eqr, [103](#)
 - NAD27, [103](#)
 - WGS84, [103](#)
- edu::ou::asgbook::projections::LambertConformal2SP, [203](#)
- edu::ou::asgbook::projections::Lambert-Conformal2SP
 - getLambert, [204](#)
 - getLatLon, [204](#)
 - LambertConformal2SP, [204](#)
 - main, [205](#)
- edu::ou::asgbook::projections::Remapper, [325](#)
 - bilinearInterpolation, [325](#)
 - nearestNeighbor, [326](#)
- edu::ou::asgbook::rasterization::BoundingBox, [53](#)
- edu::ou::asgbook::rasterization::Bounding-Box
 - BoundingBox, [53](#)
 - contains, [54](#)
 - copyOf, [54](#)
 - update, [54](#)
- edu::ou::asgbook::rasterization::CatmullRom, [55](#)
- edu::ou::asgbook::rasterization::Catmull-Rom
 - getPositionIn, [55](#)
 - interpolate, [56](#)
 - main, [57](#)
 - interpolate_and_interpolate, [57](#)
- edu::ou::asgbook::rasterization::Line, [220](#)
 - getLat0, [221](#)
 - getLat1, [221](#)
 - getLon0, [221](#)
 - getLon1, [221](#)
 - getPositionIn, [221](#)
 - getXIntercept, [222](#)
 - getYIntercept, [223](#)
 - Line, [220](#), [221](#)
 - main, [223](#)
- edu::ou::asgbook::rasterization::Polygon, [296](#)
 - contains, [297](#)
 - getArea, [298](#)
 - getBoundingBox, [298](#)
 - getCentroid, [298](#)
 - getEdges, [298](#)
 - main, [298](#)
 - Polygon, [297](#)
- edu::ou::asgbook::rbf::DataSimulator, [88](#)
- edu::ou::asgbook::rbf::DataSimulator
 - main, [88](#)
 - simulateData, [88](#), [89](#)
- edu::ou::asgbook::rbf::ProjectionPursuit, [302](#)
- edu::ou::asgbook::rbf::ProjectionPursuit
 - getCenters, [303](#)
 - getSigmax, [303](#)
 - getSigmay, [303](#)
 - main, [303](#)

- ProjectionPursuit, 303
- runOnPopDensity, 304
- runOnSimulatedInput, 304
- STRATEGY, 305
- toString, 305
- edu::ou::asgbook::rbf::ProjectionPursuit::NextRBF, 306
- edu::ou::asgbook::rbf::ProjectionPursuit::NextRBF
 - getNewCenterAndSigmas, 306
- edu::ou::asgbook::rbf::RadialBasisFunction, 316
- edu::ou::asgbook::rbf::RadialBasisFunction
 - fit, 316
 - main, 317
- edu::ou::asgbook::segmentation::ContiguityEnhancedKMeansSegmenter, 64
- edu::ou::asgbook::segmentation::ContiguityEnhancedKMeansSegmenter
 - ContiguityEnhancedKMeansSegmenter, 64
 - label, 65
 - main, 66
- edu::ou::asgbook::segmentation::EnhancedWatershedSegmenter, 104
- edu::ou::asgbook::segmentation::EnhancedWatershedSegmenter
 - EnhancedWatershedSegmenter, 105
 - label, 105
 - main, 106
 - SuppressWarnings, 106
- edu::ou::asgbook::segmentation::HysteresisSegmenter, 188
- edu::ou::asgbook::segmentation::HysteresisSegmenter
 - HysteresisSegmenter, 189
 - label, 189
 - main, 189
- edu::ou::asgbook::segmentation::LabelResult, 201
- edu::ou::asgbook::segmentation::LabelResult
 - label, 202
 - LabelResult, 201
 - maxlabel, 202
- edu::ou::asgbook::segmentation::MultiscaleKMeansSegmenter, 253
- edu::ou::asgbook::segmentation::MultiscaleKMeansSegmenter
 - label, 254
 - main, 254
 - MultiscaleKMeansSegmenter, 254
 - quantize, 255
- edu::ou::asgbook::segmentation::RegionGrowing, 318
- edu::ou::asgbook::segmentation::RegionGrowing
 - growRegion, 318
- edu::ou::asgbook::segmentation::RegionProperty, 319
- edu::ou::asgbook::segmentation::RegionProperty
 - compute, 320
 - getCval, 320
 - getCx, 320
 - getCy, 320
 - getEllipseFit, 321
 - getPixelsInRegions, 321
 - getSize, 322
- edu::ou::asgbook::segmentation::SnakeActiveContour, 344
- edu::ou::asgbook::segmentation::SnakeActiveContour
 - main, 345
 - moveSnake, 346
 - pruneAndResample, 346
 - resample, 347
 - resampleNodes, 347
 - SnakeActiveContour, 345
- edu::ou::asgbook::segmentation::SnakeActiveContour::Snake, 348
- edu::ou::asgbook::segmentation::SnakeActiveContour::Snake
 - computeEnergy, 348
 - get, 348

- getNode, 349
 - Snake, 348
- edu::ou::asgbook::segmentation::ThresholdSegmenter
 - 367
 - edu::ou::asgbook::segmentation::Threshold-Segmenter
 - label, 368
 - main, 368
 - ThresholdSegmenter, 368
- edu::ou::asgbook::segmentation::WatershedSegmenter
 - 377
 - edu::ou::asgbook::segmentation::Watershed-Segmenter
 - label, 378
 - main, 378
 - WatershedSegmenter, 378
- edu::ou::asgbook::thinning::HilditchSkeletonization
 - 162
 - edu::ou::asgbook::thinning::Hilditch-Skeletonization
 - findSkeleton, 162
 - main, 163
- edu::ou::asgbook::thinning::MedialAxisSkeletonization
 - 245
 - edu::ou::asgbook::thinning::MedialAxis-Skeletonization
 - findSkeleton, 245
 - main, 246
- edu::ou::asgbook::transforms::AlignmentEstimator
 - 47
 - edu::ou::asgbook::transforms::Alignment-Estimator
 - AlignmentEstimator, 47
 - computeCentroid, 49
 - main, 49
 - MAXU, 50
 - MAXV, 50
 - motEW, 50
 - motNS, 50
- edu::ou::asgbook::transforms::FFT, 124
 - fft, 124
 - ifft, 125
 - main, 125
- edu::ou::asgbook::transforms::FFT2D,
 - 127
 - fft, 127
 - ifft, 128
 - main, 128
- edu::ou::asgbook::transforms::FFTBandpassFilter,
 - 131
 - edu::ou::asgbook::transforms::FFTBandpass-Filter
 - convolve, 132
 - FFTBandpassFilter, 132
- edu::ou::asgbook::transforms::FFTConvolutionFilter,
 - 134
 - edu::ou::asgbook::transforms::FFTConvolution-Filter
 - convolve, 135
 - FFTConvolutionFilter, 135
- edu::ou::asgbook::transforms::HoughTransform,
 - 178
 - edu::ou::asgbook::transforms::Hough-Transform
 - findLines, 178
 - main, 179
- edu::ou::asgbook::usage::Assignment4,
 - 51
 - main, 51
- edu::ou::asgbook::usage::Sprawl, 358
 - findSprawl, 358
 - main, 360
 - runOnPopDensity, 360
- Ellipsoid
 - edu::ou::asgbook::projections::Ellipsoid, 102
 - Ellipsoid.java, 413
- EnhancedWatershedSegmenter
 - edu::ou::asgbook::segmentation::Enhanced-WatershedSegmenter, 105
 - EnhancedWatershedSegmenter.java, 414
- Entropy.java, 415
- eqr
 - edu::ou::asgbook::projections::Ellipsoid, 103
- equals
 - edu::ou::asgbook::core::Pixel, 286

- erode
 - edu::ou::asgbook::filters::Erosion-Filter, [112](#)
- ErodeDilateFilter
 - edu::ou::asgbook::filters::ErodeDilateFilter, [110](#)
- ErodeDilateFilter.java, [416](#)
- ErosionFilter
 - edu::ou::asgbook::filters::Erosion-Filter, [112](#)
- ErosionFilter.java, [417](#)
- EsriGrid.java, [418](#)
- EuclideanDT.java, [419](#)
- EuclideanDTPropagation.java, [420](#)
- EuclideanDTRRecursivePropagation.java, [421](#)
- EuclideanDTSaito.java, [422](#)
- fft
 - edu::ou::asgbook::transforms::FFT, [124](#)
 - edu::ou::asgbook::transforms::FFT2D, [127](#)
- FFT.java, [423](#)
- FFT2D.java, [424](#)
- FFTBandpassFilter
 - edu::ou::asgbook::transforms::FFTBandpassFilter, [132](#)
- FFTBandpassFilter.java, [425](#)
- FFTConvolutionFilter
 - edu::ou::asgbook::transforms::FFTConvolution-Filter, [135](#)
- FFTConvolutionFilter.java, [426](#)
- fill
 - edu::ou::asgbook::core::LatLonGrid, [213](#)
- filter
 - edu::ou::asgbook::filters::Convolution-Filter, [69](#)
 - edu::ou::asgbook::filters::Dilation-Filter, [97](#)
 - edu::ou::asgbook::filters::Erosion-Filter, [112](#)
 - edu::ou::asgbook::filters::Inverter, [192](#)
 - edu::ou::asgbook::filters::LoGEdge-Filter, [234](#)
 - edu::ou::asgbook::filters::Matched-Filter, [240](#)
 - edu::ou::asgbook::filters::Median-Filter, [248](#)
 - edu::ou::asgbook::filters::Multi-Filter, [252](#)
 - edu::ou::asgbook::filters::Oriented-EllipseFilter, [273](#)
 - edu::ou::asgbook::filters::Saturate-Filter, [328](#)
 - edu::ou::asgbook::filters::Separable-ConvolutionFilter, [335](#)
 - edu::ou::asgbook::filters::Simple-Threshold, [342](#)
 - edu::ou::asgbook::filters::Sobel-EdgeFilter, [351](#)
 - edu::ou::asgbook::filters::Spatial-Filter, [354](#)
 - edu::ou::asgbook::filters::Speckle-Filter, [356](#)
 - edu::ou::asgbook::transforms::FFTBandpass-Filter, [132](#)
 - edu::ou::asgbook::transforms::FFTConvolution-Filter, [135](#)
- findBin
 - edu::ou::asgbook::imgstat::Graylevel-CooccurrenceMatrix, [161](#)
- findHighestValued
 - edu::ou::asgbook::filters::Max Value-Filter, [243](#)
 - edu::ou::asgbook::filters::NHighest, [257](#)
 - edu::ou::asgbook::filters::NHighest-LevelSetImpl, [260](#)
- findLines
 - edu::ou::asgbook::transforms::Hough-Transform, [178](#)
- findPatterns
 - edu::ou::asgbook::datamining::City-GdiModels, [62](#)
- findPrimaryCities
 - edu::ou::asgbook::datamining::Primary-Cities, [300](#)
- findSkeleton

- edu::ou::asgbook::thinning::Hilditch-Skeletonization, [162](#)
- edu::ou::asgbook::thinning::Medial-AxisSkeletonization, [245](#)
- findSprawl
 - edu::ou::asgbook::usage::Sprawl, [358](#)
- findTrainingPattern
 - edu::ou::asgbook::datamining::GdiPattern, [152](#)
- first
 - edu::ou::asgbook::core::Pair< X, Y >, [280](#)
- fit
 - edu::ou::asgbook::rbf::RadialBasisFunction, [316](#)
- FuzzyCandidateMarket
 - edu::ou::asgbook::datamining::FuzzyCandidateMarket, [137](#)
- FuzzyCandidateMarket.java, [427](#)
- FuzzyLogic.java, [428](#)
- gauss
 - edu::ou::asgbook::filters::ConvolutionFilter, [69](#)
 - edu::ou::asgbook::filters::SeparableConvolutionFilter, [335](#), [336](#)
- GaussianComponent
 - edu::ou::asgbook::gmm::GaussianComponent, [143](#)
- GaussianComponent.java, [429](#)
- GaussianMixtureModel
 - edu::ou::asgbook::gmm::GaussianMixtureModel, [147](#)
- GaussianMixtureModel.java, [430](#)
- GaussWeighting
 - edu::ou::asgbook::oban::GaussWeighting, [151](#)
- GaussWeighting.java, [431](#)
- GdiPattern.java, [432](#)
- get
 - edu::ou::asgbook::segmentation::Snake-ActiveContour::Snake, [348](#)
- getArea
 - edu::ou::asgbook::rasterization::Polygon, [298](#)
- getAssignments
 - edu::ou::asgbook::motion::HungarianAssigner, [182](#)
 - edu::ou::asgbook::motion::ObjectTracker::Assigner, [270](#)
- getBinNumber
 - edu::ou::asgbook::histogram::Histogram, [166](#)
 - edu::ou::asgbook::imgstat::Quantizer, [311](#)
 - edu::ou::asgbook::imgstat::VectorQuantizer, [374](#)
- getBoundingBox
 - edu::ou::asgbook::rasterization::Polygon, [298](#)
- getCenters
 - edu::ou::asgbook::rbf::ProjectionPursuit, [303](#)
- getCenterValue
 - edu::ou::asgbook::histogram::Histogram, [166](#)
 - edu::ou::asgbook::imgstat::Quantizer, [312](#)
 - edu::ou::asgbook::imgstat::VectorQuantizer, [374](#)
- getCentroid
 - edu::ou::asgbook::rasterization::Polygon, [298](#)
- getCol
 - edu::ou::asgbook::core::LatLonGrid, [213](#)
 - edu::ou::asgbook::core::Pixel, [286](#)
- getCval
 - edu::ou::asgbook::segmentation::RegionProperty, [320](#)
- getCx
 - edu::ou::asgbook::gmm::GaussianComponent, [145](#)
 - edu::ou::asgbook::segmentation::RegionProperty, [320](#)
- getCy
 - edu::ou::asgbook::gmm::GaussianComponent, [145](#)
 - edu::ou::asgbook::segmentation::RegionProperty, [320](#)
- getData

- edu::ou::asgbook::core::LatLonGrid, 213
- getDefault
 - edu::ou::asgbook::io::OutputDirectory, 278
- getDistanceSquared
 - edu::ou::asgbook::core::Pixel, 287
- getDistanceTransform
 - edu::ou::asgbook::distance::Euclidean-DT, 117
 - edu::ou::asgbook::distance::Euclidean-DTPropagation, 119
 - edu::ou::asgbook::distance::Euclidean-DTRecursivePropagation, 121
 - edu::ou::asgbook::distance::Euclidean-DTSaito, 123
- getEdges
 - edu::ou::asgbook::rasterization::Polygon, 298
- getEllipseFit
 - edu::ou::asgbook::segmentation::Region-Property, 321
- getEntryClosestTo
 - edu::ou::asgbook::geocode::Usa-Zipcode, 370
- getFilterNumCols
 - edu::ou::asgbook::filters::Convolution-Filter, 70
- getFilterNumRows
 - edu::ou::asgbook::filters::Convolution-Filter, 70
- getHist
 - edu::ou::asgbook::histogram::Histogram, 167
- getIncr
 - edu::ou::asgbook::histogram::CumulativeDistributionFunction, 84
 - edu::ou::asgbook::histogram::Histogram, 167
- getInstance
 - edu::ou::asgbook::geocode::Usa-Zipcode, 371
- getLambert
 - edu::ou::asgbook::projections::Lambert-Conformal2SP, 204
- getLat
 - edu::ou::asgbook::core::LatLon, 207
- getLat0
 - edu::ou::asgbook::rasterization::Line, 221
- getLat1
 - edu::ou::asgbook::rasterization::Line, 221
- getLatLon
 - edu::ou::asgbook::projections::Lambert-Conformal2SP, 204
- getLatRes
 - edu::ou::asgbook::core::LatLonGrid, 213
- getLocation
 - edu::ou::asgbook::core::LatLonGrid, 213, 214
- getLon
 - edu::ou::asgbook::core::LatLon, 207
- getLon0
 - edu::ou::asgbook::rasterization::Line, 221
- getLon1
 - edu::ou::asgbook::rasterization::Line, 221
- getLonRes
 - edu::ou::asgbook::core::LatLonGrid, 214
- getMax
 - edu::ou::asgbook::core::Scalar-Statistic, 329
 - edu::ou::asgbook::imgstat::Local-Measures, 230
- getMaxCost
 - edu::ou::asgbook::motion::Object-Tracker::CostEstimator, 271
- getMaximumProbability
 - edu::ou::asgbook::imgstat::Structural-Measures, 362
- getMaxValue
 - edu::ou::asgbook::core::Point-Observations, 294
- getMean
 - edu::ou::asgbook::core::Scalar-Statistic, 329
 - edu::ou::asgbook::imgstat::Local-Measures, 230

- getMin
 - edu::ou::asgbook::core::Scalar-Statistic, [329](#)
 - edu::ou::asgbook::histogram::Cumulative-DistributionFunction, [84](#)
 - edu::ou::asgbook::histogram::HistogramgetPositionIn [167](#)
 - edu::ou::asgbook::imgstat::Local-Measures, [230](#)
- getMissing
 - edu::ou::asgbook::core::LatLonGrid, [214](#)
 - edu::ou::asgbook::core::Point-Observations, [295](#)
- getMixture
 - edu::ou::asgbook::gmm::Gaussian-MixtureModel, [148](#)
- getNewCenterAndSigmas
 - edu::ou::asgbook::rbf::Projection-Pursuit::NextRBF, [306](#)
- getNodes
 - edu::ou::asgbook::segmentation::Snake-ActiveContour::Snake, [349](#)
- getNumBins
 - edu::ou::asgbook::histogram::Histogram, [167](#)
- getNumLat
 - edu::ou::asgbook::core::LatLonGrid, [214](#)
- getNumLon
 - edu::ou::asgbook::core::LatLonGrid, [214](#)
- getNumSamples
 - edu::ou::asgbook::core::Scalar-Statistic, [330](#)
- getNwCorner
 - edu::ou::asgbook::core::LatLonGrid, [215](#)
- getOptimalThreshold
 - edu::ou::asgbook::histogram::Otsu-ThresholdSelector, [276](#)
- getPixel
 - edu::ou::asgbook::core::LatLonGrid, [215](#)
- getPixelsInRegions
 - edu::ou::asgbook::segmentation::Region-Property, [321](#)
- getPoints
 - edu::ou::asgbook::core::Point-Observations, [295](#)
- getPositionIn
 - edu::ou::asgbook::rasterization::Catmull-Rom, [55](#)
 - edu::ou::asgbook::rasterization::Line, [221](#)
- getProb
 - edu::ou::asgbook::histogram::Cumulative-DistributionFunction, [84](#)
- getRateOfChange
 - edu::ou::asgbook::motion::Kalman-Filter, [194](#)
- getRow
 - edu::ou::asgbook::core::LatLonGrid, [215](#)
 - edu::ou::asgbook::core::Pixel, [287](#)
- getSeCorner
 - edu::ou::asgbook::core::LatLonGrid, [215](#)
- getSigmax
 - edu::ou::asgbook::gmm::Gaussian-Component, [145](#)
 - edu::ou::asgbook::rbf::Projection-Pursuit, [303](#)
- getSigmaxy
 - edu::ou::asgbook::gmm::Gaussian-Component, [145](#)
- getSigmay
 - edu::ou::asgbook::gmm::Gaussian-Component, [145](#)
 - edu::ou::asgbook::rbf::Projection-Pursuit, [303](#)
- getSize
 - edu::ou::asgbook::segmentation::Region-Property, [322](#)
- getStdDeviation
 - edu::ou::asgbook::core::Scalar-Statistic, [330](#)
 - edu::ou::asgbook::imgstat::Local-Measures, [230](#)
- getUniformity

- edu::ou::asgbook::imgstat::Structural-Measures, 362
- getValue
 - edu::ou::asgbook::core::LatLonGrid, 215, 216
 - edu::ou::asgbook::core::Pixel, 287
 - edu::ou::asgbook::motion::Kalman-Filter, 194
- getVariance
 - edu::ou::asgbook::core::Scalar-Statistic, 330
- getWeight
 - edu::ou::asgbook::gmm::Gaussian-Component, 145
- getX
 - edu::ou::asgbook::core::Pixel, 287
- getXIntercept
 - edu::ou::asgbook::rasterization::Line, 222
- getY
 - edu::ou::asgbook::core::Pixel, 287
- getYIntercept
 - edu::ou::asgbook::rasterization::Line, 223
- GlobalPopulation.java, 433
- GraylevelCooccurrenceMatrix
 - edu::ou::asgbook::imgstat::Graylevel-CooccurrenceMatrix, 159
- GraylevelCooccurrenceMatrix.java, 434
- growRegion
 - edu::ou::asgbook::segmentation::Region-Growing, 318
- HighIncomeNonOECD
 - edu::ou::asgbook::dataset::WorldBankGDI, 390
- HighIncomeOECD
 - edu::ou::asgbook::dataset::WorldBankGDI, 390
- HilditchSkeletonization.java, 435
- Histogram
 - edu::ou::asgbook::histogram::Histogram, 165
- Histogram.java, 436
- HistogramBinSelection.java, 437
- HornSchunk.java, 438
- HoughTransform.java, 439
- HungarianAssigner.java, 440
- HybridTracker
 - edu::ou::asgbook::motion::Hybrid-Tracker, 185
- HybridTracker.java, 441
- HysteresisSegmenter
 - edu::ou::asgbook::segmentation::Hysteresis-Segmenter, 189
- HysteresisSegmenter.java, 442
- ifft
 - edu::ou::asgbook::transforms::FFT, 125
 - edu::ou::asgbook::transforms::FFT2D, 128
- init
 - edu::ou::asgbook::motion::Kalman-Filter, 194
- interpolate
 - edu::ou::asgbook::rasterization::Catmull-Rom, 56
- INVALID_WEIGHT
 - edu::ou::asgbook::oban::Weight-Function, 388
- inverse
 - edu::ou::asgbook::linearity::Data-Transform, 90
 - edu::ou::asgbook::linearity::Linear-Scaling, 228
 - edu::ou::asgbook::linearity::Log-Scaling, 236
- invert
 - edu::ou::asgbook::filters::Inverter, 192
- Inverter
 - edu::ou::asgbook::filters::Inverter, 191
- Inverter.java, 443
- isGoodCandidate
 - edu::ou::asgbook::datamining::Fuzzy-CandidateMarket, 138
- isValid
 - edu::ou::asgbook::core::LatLonGrid, 216

- edu::ou::asgbook::gmm::Gaussian-Component, 146
- KalmanFilter
 - edu::ou::asgbook::motion::Kalman-Filter, 193
- KalmanFilter.java, 444
- KmlWriter.java, 445
- kth_element
 - edu::ou::asgbook::filters::Quick-Select, 314, 315
- label
 - edu::ou::asgbook::segmentation::Contiguity-EnhancedKMeansSegmenter, 65
 - edu::ou::asgbook::segmentation::EnhancedWatershedSegmenter, 105
 - edu::ou::asgbook::segmentation::Hysteresis-Segmenter, 189
 - edu::ou::asgbook::segmentation::Label-Result, 202
 - edu::ou::asgbook::segmentation::Multiscale-KMeansSegmenter, 254
 - edu::ou::asgbook::segmentation::Segmenter, 332
 - edu::ou::asgbook::segmentation::Threshold-Segmenter, 368
 - edu::ou::asgbook::segmentation::Watershed-Segmenter, 378
- LabelResult
 - edu::ou::asgbook::segmentation::Label-Result, 201
- LabelResult.java, 446
- LambertConformal2SP
 - edu::ou::asgbook::projections::Lambert-Conformal2SP, 204
- LambertConformal2SP.java, 447
- LatLon
 - edu::ou::asgbook::core::LatLon, 206
- LatLon.java, 448
- LatLonGrid
 - edu::ou::asgbook::core::LatLonGrid, 210, 211
- LatLonGrid.java, 449
- latres
 - edu::ou::asgbook::oban::Weighted-Average, 383
- LevelSet.java, 450
- Line
 - edu::ou::asgbook::rasterization::Line, 220, 221
- Line.java, 451
- LinearityVerifier.java, 452
- LinearScaling
 - edu::ou::asgbook::linearity::Linear-Scaling, 228
- LinearScaling.java, 453
- LocalMeasures
 - edu::ou::asgbook::imgstat::Local-Measures, 230
- LocalMeasures.java, 454
- LoGEdgeFilter
 - edu::ou::asgbook::filters::LoGEdge-Filter, 233
- LoGEdgeFilter.java, 455
- LogScaling
 - edu::ou::asgbook::linearity::Log-Scaling, 235
- LogScaling.java, 456
- longitudewrap
 - edu::ou::asgbook::core::LatLonGrid, 216
- lonres
 - edu::ou::asgbook::oban::Weighted-Average, 383
- LowerMiddleIncome
 - edu::ou::asgbook::dataset::WorldBankGDI, 390
- LowIncome
 - edu::ou::asgbook::dataset::WorldBankGDI, 390
- MadisTemperature.java, 457
- main
 - edu::ou::asgbook::core::LatLon, 207
 - edu::ou::asgbook::core::Pixel, 288
 - edu::ou::asgbook::datamining::City-Categories, 59
 - edu::ou::asgbook::datamining::City-GdiModels, 63

- edu::ou::asgbook::datamining::Fuzzy-CandidateMarket, [138](#)
- edu::ou::asgbook::datamining::Gdi-Pattern, [152](#)
- edu::ou::asgbook::datamining::Primary-Cities, [301](#)
- edu::ou::asgbook::dataset::Country-Polygons, [73](#)
- edu::ou::asgbook::dataset::Daily-Rainfall, [86](#)
- edu::ou::asgbook::dataset::Global-Population, [155](#)
- edu::ou::asgbook::dataset::Madis-Temperature, [237](#)
- edu::ou::asgbook::dataset::Nighttime-Lights, [262](#)
- edu::ou::asgbook::dataset::Seviri-InfraredTemperature, [338](#)
- edu::ou::asgbook::dataset::Surface-Albedo, [363](#)
- edu::ou::asgbook::dataset::World-BankGDI, [390](#)
- edu::ou::asgbook::distance::Euclidean-DTPropagation, [119](#)
- edu::ou::asgbook::distance::Euclidean-DTRecursivePropagation, [121](#)
- edu::ou::asgbook::distance::Euclidean-DTSaito, [123](#)
- edu::ou::asgbook::filters::Convolution-Filter, [70](#)
- edu::ou::asgbook::filters::Dilate-ErodeFilter, [95](#)
- edu::ou::asgbook::filters::Dilation-Filter, [97](#)
- edu::ou::asgbook::filters::Erode-DilateFilter, [110](#)
- edu::ou::asgbook::filters::Erosion-Filter, [112](#)
- edu::ou::asgbook::filters::LoGEdge-Filter, [234](#)
- edu::ou::asgbook::filters::Matched-Filter, [240](#)
- edu::ou::asgbook::filters::MaxValue-Filter, [244](#)
- edu::ou::asgbook::filters::Median-Filter, [248](#)
- edu::ou::asgbook::filters::NHighest, [258](#)
- edu::ou::asgbook::filters::NHighest-LevelSetImpl, [261](#)
- edu::ou::asgbook::filters::Oriented-EllipseFilter, [273](#)
- edu::ou::asgbook::filters::Separable-ConvolutionFilter, [336](#)
- edu::ou::asgbook::filters::Simple-Threshold, [342](#)
- edu::ou::asgbook::filters::Sobel-EdgeFilter, [352](#)
- edu::ou::asgbook::filters::Speckle-Filter, [356](#)
- edu::ou::asgbook::geocode::Usa-Zipcode, [371](#)
- edu::ou::asgbook::gmm::Gaussian-MixtureModel, [148](#)
- edu::ou::asgbook::histogram::Cumulative-DistributionFunction, [84](#)
- edu::ou::asgbook::histogram::Entropy, [107](#)
- edu::ou::asgbook::histogram::Histogram, [167](#)
- edu::ou::asgbook::histogram::Histogram-BinSelection, [172](#)
- edu::ou::asgbook::histogram::Otsu-ThresholdSelector, [276](#)
- edu::ou::asgbook::imgstat::Local-Measures, [230](#)
- edu::ou::asgbook::imgstat::Quantizer, [312](#)
- edu::ou::asgbook::imgstat::Structural-Measures, [362](#)
- edu::ou::asgbook::imgstat::Vector-Quantizer, [374](#)
- edu::ou::asgbook::io::KmlWriter, [197](#)
- edu::ou::asgbook::io::PngWriter, [292](#)
- edu::ou::asgbook::linearity::Linearity-Verifier, [224](#)
- edu::ou::asgbook::motion::Cross-Correlation, [81](#)
- edu::ou::asgbook::motion::Differencer, [92](#)

- edu::ou::asgbook::motion::Edge-Based, [100](#)
- edu::ou::asgbook::motion::Horn-Schunk, [176](#)
- edu::ou::asgbook::motion::Hungarian-Assigner, [183](#)
- edu::ou::asgbook::motion::Hybrid-Tracker, [187](#)
- edu::ou::asgbook::motion::Kalman-Filter, [194](#)
- edu::ou::asgbook::motion::Object-Tracker, [268](#)
- edu::ou::asgbook::motion::Phase-Correlation, [283](#)
- edu::ou::asgbook::motion::Pyramidal-CrossCorrelation, [309](#)
- edu::ou::asgbook::oban::Cressman-Weighting, [77](#)
- edu::ou::asgbook::oban::Gauss-Weighting, [151](#)
- edu::ou::asgbook::oban::Weighted-Average, [382](#)
- edu::ou::asgbook::oban::Weighted-AverageOptimized, [386](#)
- edu::ou::asgbook::projections::Lambert-Conformal2SP, [205](#)
- edu::ou::asgbook::rasterization::Catmull-Rom, [57](#)
- edu::ou::asgbook::rasterization::Line, [223](#)
- edu::ou::asgbook::rasterization::Polygon, [298](#)
- edu::ou::asgbook::rbf::Data-Simulator, [88](#)
- edu::ou::asgbook::rbf::Projection-Pursuit, [303](#)
- edu::ou::asgbook::rbf::RadialBasis-Function, [317](#)
- edu::ou::asgbook::segmentation::Contiguity-EnhancedKMeansSegmenter, [66](#)
- edu::ou::asgbook::segmentation::Enhanced-WatershedSegmenter, [106](#)
- edu::ou::asgbook::segmentation::Hysteresis-Segmenter, [189](#)
- edu::ou::asgbook::segmentation::Multiscale-KMeansSegmenter, [254](#)
- edu::ou::asgbook::segmentation::Region-Property, [322](#)
- edu::ou::asgbook::segmentation::Snake-ActiveContour, [345](#)
- edu::ou::asgbook::segmentation::Threshold-Segmenter, [368](#)
- edu::ou::asgbook::segmentation::Watershed-Segmenter, [378](#)
- edu::ou::asgbook::thinning::Hilditch-Skeletonization, [163](#)
- edu::ou::asgbook::thinning::Medial-AxisSkeletonization, [246](#)
- edu::ou::asgbook::transforms::Alignment-Estimator, [49](#)
- edu::ou::asgbook::transforms::FFT, [125](#)
- edu::ou::asgbook::transforms::FFT2D, [128](#)
- edu::ou::asgbook::transforms::FFTBandpass-Filter, [133](#)
- edu::ou::asgbook::transforms::FFTConvolution-Filter, [135](#)
- edu::ou::asgbook::transforms::Hough-Transform, [179](#)
- edu::ou::asgbook::usage::Assignment4, [51](#)
- edu::ou::asgbook::usage::Sprawl, [360](#)
- edu::ou::asgbook::filters::Matched-Filter, [241](#)
- edu::ou::asgbook::filters::Matched-Filter, [240](#)
- edu::ou::asgbook::filters::MatchedFilter.java, [458](#)
- edu::ou::asgbook::segmentation::Label-Result, [202](#)
- edu::ou::asgbook::motion::Phase-Correlation, [284](#)
- edu::ou::asgbook::transforms::Alignment-Estimator, [50](#)
- edu::ou::asgbook::motion::Phase-

- Correlation, [284](#)
- edu::ou::asgbook::transforms::Alignment-
Estimator, [50](#)
- MaxValueFilter.java, [459](#)
- MedialAxisSkeletonization.java, [460](#)
- MedianFilter
 - edu::ou::asgbook::filters::Median-
Filter, [248](#)
- MedianFilter.java, [461](#)
- minPoints
 - edu::ou::asgbook::oban::Weighted-
Average, [383](#)
- MOT_SCALE
 - edu::ou::asgbook::motion::Horn-
Schunk, [177](#)
- motEW
 - edu::ou::asgbook::transforms::Alignment-
Estimator, [50](#)
- MotionEstimator.java, [462](#)
- motNS
 - edu::ou::asgbook::transforms::Alignment-
Estimator, [50](#)
- moveSnake
 - edu::ou::asgbook::segmentation::Snake-
ActiveContour, [346](#)
- MultiFilter
 - edu::ou::asgbook::filters::Multi-
Filter, [251](#)
- MultiFilter.java, [463](#)
- MultiscaleKMeansSegmenter
 - edu::ou::asgbook::segmentation::Multiscale-
KMeansSegmenter, [254](#)
- MultiscaleKMeansSegmenter.java, [464](#)
- NAD27
 - edu::ou::asgbook::projections::EllipsoidPair, [103](#)
- nearestNeighbor
 - edu::ou::asgbook::projections::Remapper, [326](#)
- NHighest
 - edu::ou::asgbook::filters::NHighest, [257](#)
- NHighest.java, [465](#)
- NHighestLevelSetImpl
 - edu::ou::asgbook::filters::NHighest-
LevelSetImpl, [260](#)
 - NHighestLevelSetImpl.java, [466](#)
 - NighttimeLights.java, [467](#)
 - normalize
 - edu::ou::asgbook::imgstat::Vector-
Quantizer, [375](#), [376](#)
 - NORTHAMERICA
 - edu::ou::asgbook::dataset::Global-
Population, [157](#)
 - NORTHAMERICA1990
 - edu::ou::asgbook::dataset::Global-
Population, [157](#)
 - NORTHEAST
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, [159](#)
 - ObjectiveAnalysisUtils.java, [468](#)
 - ObjectTracker
 - edu::ou::asgbook::motion::Object-
Tracker, [267](#)
 - ObjectTracker.java, [469](#)
 - OrientedEllipseFilter
 - edu::ou::asgbook::filters::Oriented-
EllipseFilter, [273](#)
 - OrientedEllipseFilter.java, [470](#)
 - OtsuThresholdSelector
 - edu::ou::asgbook::histogram::Otsu-
ThresholdSelector, [275](#)
 - OtsuThresholdSelector.java, [471](#)
 - OutputDirectory.java, [472](#)
 - package-info.java, [473–492](#)
 - Pair
 - edu::ou::asgbook::core::Pair< X, Y
>, [280](#)
 - Pair.java, [493](#)
 - PhaseCorrelation
 - edu::ou::asgbook::motion::Phase-
Correlation, [281](#)
 - PhaseCorrelation.java, [494](#)
 - Pixel
 - edu::ou::asgbook::core::Pixel, [286](#)
 - Pixel.java, [495](#)
 - PngWriter.java, [496](#)
 - PointObservations

- edu::ou::asgbook::core::Point-
Observations, [294](#)
- PointObservations.java, [497](#)
- Polygon
 - edu::ou::asgbook::rasterization::Polygon, [297](#)
- Polygon.java, [498](#)
- PrimaryCities.java, [499](#)
- ProjectionPursuit
 - edu::ou::asgbook::rbf::Projection-
Pursuit, [303](#)
- ProjectionPursuit.java, [500](#)
- prune
 - edu::ou::asgbook::segmentation::Region-
Property, [323](#)
- pruneAndResample
 - edu::ou::asgbook::segmentation::Snake-
ActiveContour, [346](#)
- pruneBySize
 - edu::ou::asgbook::segmentation::Region-
Property, [323](#)
- PyramidalCrossCorrelation
 - edu::ou::asgbook::motion::Pyramidal-
CrossCorrelation, [308](#)
- PyramidalCrossCorrelation.java, [501](#)
- quantize
 - edu::ou::asgbook::segmentation::Multiscale-
KMeansSegmenter, [255](#)
- Quantizer
 - edu::ou::asgbook::imgstat::Quantizer, [310](#)
- Quantizer.java, [502](#)
- QuickSelect.java, [503](#)
- RadialBasisFunction.java, [504](#)
- read
 - edu::ou::asgbook::dataset::Daily-
Rainfall, [86](#)
 - edu::ou::asgbook::dataset::Global-
Population, [156](#)
 - edu::ou::asgbook::dataset::Madis-
Temperature, [237](#)
 - edu::ou::asgbook::dataset::Nighttime-
Lights, [262](#)
 - edu::ou::asgbook::dataset::Seviri-
InfraredTemperature, [339](#)
 - edu::ou::asgbook::dataset::Surface-
Albedo, [363](#), [364](#)
 - edu::ou::asgbook::dataset::World-
BankGDI, [391](#)
 - edu::ou::asgbook::io::EsriGrid, [114](#),
[115](#)
 - edu::ou::asgbook::dataset::Seviri-
InfraredTemperature, [339](#)
 - edu::ou::asgbook::dataset::World-
BankGDI, [391](#)
 - edu::ou::asgbook::dataset::Country-
Polygons, [74](#)
 - edu::ou::asgbook::dataset::World-
BankGDI, [392](#)
 - edu::ou::asgbook::dataset::Country-
Polygons, [74](#)
 - RegionGrowing.java, [505](#)
 - RegionProperty.java, [506](#)
 - edu::ou::asgbook::io::Output-
Directory, [278](#)
 - Remapper.java, [507](#)
 - edu::ou::asgbook::core::LatLonGrid,
[217](#)
 - edu::ou::asgbook::core::LatLonGrid,
[217](#)
 - edu::ou::asgbook::segmentation::Snake-
ActiveContour, [347](#)
 - edu::ou::asgbook::segmentation::Snake-
ActiveContour, [347](#)
 - edu::ou::asgbook::rbf::Projection-
Pursuit, [304](#)
 - edu::ou::asgbook::usage::Sprawl,
[360](#)
- readAll
- readAsMap
- readGrid
- readKml
- relative
- replace
- resample
- resampleNodes
- runOnPopDensity
- runOnSimulatedInput

- edu::ou::asgbook::rbf::Projection-Pursuit, 304
- saturate
 - edu::ou::asgbook::filters::Saturate-Filter, 328
 - edu::ou::asgbook::filters::Sobel-EdgeFilter, 352
- SaturateFilter
 - edu::ou::asgbook::filters::Saturate-Filter, 327
- SaturateFilter.java, 508
- ScalarStatistic.java, 509
- second
 - edu::ou::asgbook::core::Pair< X, Y >, 280
- Segmenter.java, 510
- SeparableConvolutionFilter
 - edu::ou::asgbook::filters::Separable-ConvolutionFilter, 335
- SeparableConvolutionFilter.java, 511
- setMissing
 - edu::ou::asgbook::core::LatLonGrid, 217
- setValue
 - edu::ou::asgbook::core::LatLonGrid, 217
- setValueInGrid
 - edu::ou::asgbook::gmm::Gaussian-MixtureModel, 148
- SeviriInfraredTemperature.java, 512
- SimpleThresholder
 - edu::ou::asgbook::filters::Simple-Thresholder, 342
- SimpleThresholder.java, 513
- simulateData
 - edu::ou::asgbook::rbf::Data-Simulator, 88, 89
- smooth
 - edu::ou::asgbook::filters::Convolution-Filter, 71
 - edu::ou::asgbook::filters::Median-Filter, 248
 - edu::ou::asgbook::filters::Oriented-EllipseFilter, 274
- edu::ou::asgbook::filters::Separable-ConvolutionFilter, 336
- Snake
 - edu::ou::asgbook::segmentation::Snake-ActiveContour::Snake, 348
- SnakeActiveContour
 - edu::ou::asgbook::segmentation::Snake-ActiveContour, 345
- SnakeActiveContour.java, 514
- SobelEdgeFilter
 - edu::ou::asgbook::filters::Sobel-EdgeFilter, 351
- SobelEdgeFilter.java, 515
- sort_and_interpolate
 - edu::ou::asgbook::rasterization::Catmull-Rom, 57
- SOUTHEAST
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, 159
- SOUTHWARD
 - edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix, 159
- SpatialFilter.java, 516
- SpeckleFilter
 - edu::ou::asgbook::filters::Speckle-Filter, 356
- speckleFilter
 - edu::ou::asgbook::filters::Speckle-Filter, 356
- SpeckleFilter.java, 517
- Sprawl.java, 518
- STRATEGY
 - edu::ou::asgbook::rbf::Projection-Pursuit, 305
- StructuralMeasures
 - edu::ou::asgbook::imgstat::Structural-Measures, 361
- StructuralMeasures.java, 519
- SuppressWarnings
 - edu::ou::asgbook::dataset::Daily-Rainfall, 87
 - edu::ou::asgbook::dataset::Madis-Temperature, 238
 - edu::ou::asgbook::segmentation::Enhanced-WatershedSegmenter, 106
- SurfaceAlbedo.java, 520

- temporary
 - edu::ou::asgbook::io::Output-Directory, [278](#)
- test
 - edu::ou::asgbook::motion::Cross-Correlation, [82](#)
 - edu::ou::asgbook::motion::Phase-Correlation, [283](#)
- text
 - edu::ou::asgbook::dataset::WorldBankGDI, [390](#)
- threshold
 - edu::ou::asgbook::filters::Simple-Thresholder, [342](#)
- ThresholdSegmenter
 - edu::ou::asgbook::segmentation::Threshold-Segmenter, [368](#)
- ThresholdSegmenter.java, [521](#)
- TN_Oct2010
 - edu::ou::asgbook::dataset::Daily-Rainfall, [87](#)
 - edu::ou::asgbook::dataset::Madis-Temperature, [238](#)
- toString
 - edu::ou::asgbook::core::LatLon, [207](#)
 - edu::ou::asgbook::core::Pixel, [288](#)
 - edu::ou::asgbook::core::Scalar-Statistic, [330](#)
 - edu::ou::asgbook::datamining::Gdi-Pattern, [153](#)
 - edu::ou::asgbook::gmm::Gaussian-Component, [146](#)
 - edu::ou::asgbook::histogram::Cumulative-DistributionFunction, [85](#)
 - edu::ou::asgbook::histogram::Histogram, [168](#)
 - edu::ou::asgbook::imgstat::Quantizer, [312](#)
 - edu::ou::asgbook::imgstat::Vector-Quantizer, [376](#)
 - edu::ou::asgbook::rbf::Projection-Pursuit, [305](#)
- transform
 - edu::ou::asgbook::linearity::Data-Transform, [90](#)
 - edu::ou::asgbook::linearity::Linear-Scaling, [228](#)
 - edu::ou::asgbook::linearity::Log-Scaling, [236](#)
 - transformAndRoundoff
 - edu::ou::asgbook::linearity::Data-Transform, [90](#)
 - update
 - edu::ou::asgbook::core::Scalar-Statistic, [330](#), [331](#)
 - edu::ou::asgbook::histogram::Histogram, [168](#)
 - edu::ou::asgbook::motion::Kalman-Filter, [195](#)
 - edu::ou::asgbook::rasterization::Bounding-Box, [54](#)
 - updated
 - edu::ou::asgbook::motion::Kalman-Filter, [196](#)
 - UpperMiddleIncome
 - edu::ou::asgbook::dataset::WorldBankGDI, [390](#)
 - UsaZipcode.java, [522](#)
 - VectorQuantizer
 - edu::ou::asgbook::imgstat::Vector-Quantizer, [373](#)
 - VectorQuantizer.java, [523](#)
 - verify
 - edu::ou::asgbook::linearity::Linearity-Verifier, [225](#)
 - WatershedSegmenter
 - edu::ou::asgbook::segmentation::Watershed-Segmenter, [378](#)
 - WatershedSegmenter.java, [524](#)
 - WeightedAverage
 - edu::ou::asgbook::oban::Weighted-Average, [381](#)
 - WeightedAverage.java, [525](#)
 - WeightedAverageOptimized
 - edu::ou::asgbook::oban::Weighted-AverageOptimized, [385](#)
 - WeightedAverageOptimized.java, [526](#)
 - WeightFunction.java, [527](#)

WGS84
 edu::ou::asgbook::projections::Ellipsoid,
 103

WORLD
 edu::ou::asgbook::dataset::Global-
 Population, 157
 edu::ou::asgbook::dataset::Nighttime-
 Lights, 263

WORLD_GRID
 edu::ou::asgbook::dataset::Country-
 Polygons, 75
 edu::ou::asgbook::dataset::World-
 BankGDI, 392

WORLD_KML
 edu::ou::asgbook::dataset::Country-
 Polygons, 75

WORLD_TABULAR
 edu::ou::asgbook::dataset::World-
 BankGDI, 392

WorldBankGDI.java, 528

write
 edu::ou::asgbook::datamining::Gdi-
 Pattern, 154
 edu::ou::asgbook::io::EsriGrid, 116
 edu::ou::asgbook::io::KmlWriter,
 198, 199

writeAutoScaled
 edu::ou::asgbook::io::PngWriter,
 292

writeScaled
 edu::ou::asgbook::io::PngWriter,
 293

wtFunc
 edu::ou::asgbook::oban::Weighted-
 Average, 383

xadd
 edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix,
 159

yadd
 edu::ou::asgbook::imgstat::GraylevelCooccurrenceMatrix,
 159

zeropad
 edu::ou::asgbook::transforms::FFT2D,
 129, 130