Bayesian Cluster Tool

Generated by Doxygen 1.9.1

Fri Jun 16 2023 11:54:55

1 Todo List	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 AuxConfiguration Class Reference	7
4.1.1 Detailed Description	8
4.1.2 Constructor & Destructor Documentation	8
4.1.2.1 AuxConfiguration() [1/2]	8
4.1.2.2 AuxConfiguration() [2/2]	9
4.1.3 Member Function Documentation	9
4.1.3.1 ClusterR()	9
4.1.3.2 ClusterT()	9
4.1.3.3 configFile()	10
4.1.3.4 FromVector()	10
4.1.3.5 inputFile()	10
4.1.3.6 outputFile()	11
4.1.3.7 SetConfigFile()	11
4.1.3.8 SetInputFile()	11
4.1.3.9 SetOutputFile()	12
4.1.3.10 SetValidate()	
4.1.3.11 validate()	12
4.2 Cluster Class Reference	13
4.2.1 Detailed Description	14
4.2.2 Constructor & Destructor Documentation	14
4.2.2.1 Cluster() [1/4]	14
4.2.2.2 Cluster() [2/4]	
4.2.2.3 Cluster() [3/4]	15
4.2.2.4 Cluster() [4/4]	
4.2.3 Member Function Documentation	
4.2.3.1 GetParent()	15
4.2.3.2 operator+=()	16
4.2.3.3 operator=() [1/2]	16
4.2.3.4 operator=() [2/2]	17
4.2.3.5 UpdateLogScore()	
4.3 ClusterWrapper Struct Reference	17
4.3.1 Detailed Description	18
4.3.2 Member Function Documentation	18
4.3.2.1 operator==()	

4.4 Data Class Reference	19
4.4.1 Detailed Description	20
4.4.2 Constructor & Destructor Documentation	20
4.4.2.1 Data() [1/3]	20
4.4.2.2 Data() [2/3]	21
4.4.2.3 Data() [3/3]	21
4.4.3 Member Function Documentation	21
4.4.3.1 CalculateLocalizationScore()	21
4.4.3.2 dPhi()	22
4.4.3.3 dR()	22
4.4.3.4 dR2()	23
4.4.3.5 operator<()	23
4.4.3.6 operator=() [1/2]	23
4.4.3.7 operator=() [2/2]	24
4.4.3.8 Preprocess()	24
4.4.3.9 PreprocessLocalizationScores()	25
4.5 DataProxy Class Reference	25
4.5.1 Detailed Description	26
4.5.2 Constructor & Destructor Documentation	26
4.5.2.1 DataProxy() [1/3]	26
4.5.2.2 DataProxy() [2/3]	27
4.5.2.3 DataProxy() [3/3]	27
4.5.3 Member Function Documentation	27
4.5.3.1 Clusterize() [1/2]	27
4.5.3.2 Clusterize() [2/2]	28
4.5.3.3 GetCluster()	28
4.5.3.4 operator=() [1/2]	28
4.5.3.5 operator=() [2/2]	29
4.6 GSLInterpolator Class Reference	29
4.6.1 Detailed Description	30
4.6.2 Constructor & Destructor Documentation	30
4.6.2.1 GSLInterpolator() [1/4]	30
4.6.2.2 GSLInterpolator() [2/4]	31
4.6.2.3 GSLInterpolator() [3/4]	31
4.6.2.4 GSLInterpolator() [4/4]	31
4.6.3 Member Function Documentation	32
4.6.3.1 Deriv()	32
4.6.3.2 Deriv2()	32
4.6.3.3 Eval()	33
4.6.3.4 Evaluate()	33
4.6.3.5 Integ()	34
4.6.3.6 operator=() [1/2]	34

4.6.3.7 operator=() [2/2]	34
4.6.3.8 SetData() [1/2]	35
4.6.3.9 SetData() [2/2]	35
4.7 LocalizationFile Class Reference	36
4.7.1 Detailed Description	36
4.7.2 Constructor & Destructor Documentation	36
4.7.2.1 LocalizationFile() [1/3]	36
4.7.2.2 LocalizationFile() [2/3]	37
4.7.2.3 LocalizationFile() [3/3]	37
4.7.3 Member Function Documentation	37
4.7.3.1 ExtractRols() [1/2]	37
4.7.3.2 ExtractRols() [2/2]	38
4.7.3.3 operator=() [1/2]	38
4.7.3.4 operator=() [2/2]	39
4.8 ManualRoI Struct Reference	39
4.8.1 Detailed Description	39
4.9 Cluster::Parameter Struct Reference	40
4.9.1 Detailed Description	40
4.9.2 Member Function Documentation	40
4.9.2.1 alt_log_score()	41
4.9.2.2 log_score()	41
4.9.2.3 operator+=()	41
4.10 ProgressBar Class Reference	42
4.10.1 Detailed Description	43
4.10.2 Constructor & Destructor Documentation	43
4.10.2.1 ProgressBar()	43
4.10.3 Member Function Documentation	43
4.10.3.1 operator++()	43
4.11 ProgressTimer Struct Reference	44
4.11.1 Detailed Description	44
4.11.2 Constructor & Destructor Documentation	44
4.11.2.1 ProgressTimer()	44
4.12 Rol Class Reference	45
4.12.1 Detailed Description	46
4.12.2 Constructor & Destructor Documentation	46
4.12.2.1 Rol() [1/3]	46
4.12.2.2 Rol() [2/3]	47
4.12.2.3 Rol() [3/3]	47
4.12.3 Member Function Documentation	47
4.12.3.1 Clusterize()	47
4.12.3.2 getArea()	48
4.12.3.3 getCentreX()	48

4.12.3.4 getCentreY()	. 48
4.12.3.5 getWidthX()	. 49
4.12.3.6 getWidthY()	. 49
4.12.3.7 operator=() [1/2]	. 49
4.12.3.8 operator=() [2/2]	. 50
4.12.3.9 Preprocess()	. 50
4.12.3.10 ScanRT()	. 50
4.12.3.11 SetCentre()	. 51
4.12.3.12 SetWidth()	. 51
4.13 Rolproxy Class Reference	. 52
4.13.1 Detailed Description	. 53
4.13.2 Constructor & Destructor Documentation	. 53
4.13.2.1 Rolproxy() [1/3]	. 53
4.13.2.2 Rolproxy() [2/3]	. 53
4.13.2.3 Rolproxy() [3/3]	. 54
4.13.3 Member Function Documentation	. 54
4.13.3.1 CheckClusterization()	. 54
4.13.3.2 Clusterize()	. 54
4.13.3.3 GetData()	. 55
4.13.3.4 operator=() [1/2]	. 55
4.13.3.5 operator=() [2/2]	. 56
4.13.3.6 ScanRT()	. 56
4.13.3.7 UpdateLogScore()	. 57
4.13.3.8 ValidateLogScore()	. 57
4.14 ScanConfiguration Class Reference	. 57
4.14.1 Detailed Description	. 60
4.14.2 Constructor & Destructor Documentation	. 60
4.14.2.1 ScanConfiguration() [1/4]	. 60
4.14.2.2 ScanConfiguration() [2/4]	. 61
4.14.2.3 ScanConfiguration() [3/4]	. 62
4.14.2.4 ScanConfiguration() [4/4]	. 62
4.14.3 Member Function Documentation	. 62
4.14.3.1 alpha()	. 62
4.14.3.2 FromVector()	. 62
4.14.3.3 log_probability_sigma() [1/2]	. 63
4.14.3.4 log_probability_sigma() [2/2]	. 63
4.14.3.5 logAlpha()	. 64
4.14.3.6 logGammaAlpha()	. 64
4.14.3.7 logPb()	. 64
4.14.3.8 logPbDagger()	. 65
4.14.3.9 operator=() [1/2]	. 65
4.14.3.10 operator=() [2/2]	. 65

4.14.3.11 probability_sigma() [1/2]	66
4.14.3.12 probability_sigma() [2/2]	66
4.14.3.13 Rbounds()	66
4.14.3.14 SetAlpha()	67
4.14.3.15 SetPb()	67
4.14.3.16 SetRBins()	67
4.14.3.17 SetSigmaParameters()	68
4.14.3.18 SetTBins()	68
4.14.3.19 sigmabins() [1/2]	69
4.14.3.20 sigmabins() [2/2]	69
4.14.3.21 sigmabins2() [1/2]	69
4.14.3.22 sigmabins2() [2/2]	70
4.14.3.23 sigmacount()	70
4.14.3.24 sigmaspacing()	71
4.14.3.25 Tbounds()	71
4.15 ScanEntry Struct Reference	71
4.15.1 Detailed Description	72
4.15.2 Member Function Documentation	72
4.15.2.1 operator<()	72
4.15.2.2 operator==()	72
4.16 ScanConfiguration::tBounds Struct Reference	73
	, ,
4.16.1 Detailed Description	
	73
5 File Documentation	73
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference	73 75
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference	73 75 75
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference	73 75 76 77
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback()	73 75 76 77 78
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback()	73 75 76 76 78
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback()	73 75 75 76 77 78 78
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson()	73 75 75 76 77 78 78 79
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback()	73 75 75 76 77 78 78 79 79
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback() 5.1.1.7 ManualRoi_Cluster_SimpleCallback()	73 75 75 76 77 78 78 79 79 80
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback() 5.1.1.7 ManualRoi_Cluster_FullCallback() 5.1.1.8 ManualRoi_Scan_FullCallback()	73 75 75 76 77 78 78 79 79 80 80 81
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback() 5.1.1.7 ManualRoi_Cluster_FullCallback() 5.1.1.8 ManualRoi_Scan_FullCallback() 5.1.1.9 ManualRoi_Scan_SimpleCallback()	73 75 75 76 77 78 78 79 79 80 80 81
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback() 5.1.1.7 ManualRoi_Cluster_FullCallback() 5.1.1.8 ManualRoi_Scan_FullCallback() 5.1.1.9 ManualRoi_Scan_FullCallback() 5.1.1.9 ManualRoi_Scan_SimpleCallback() 5.1.1.10 ManualRoi_Scan_ToJson()	73 75 76 77 78 78 79 79 80 80 81 81
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback() 5.1.1.7 ManualRoi_Cluster_FullCallback() 5.1.1.8 ManualRoi_Scan_FullCallback() 5.1.1.9 ManualRoi_Scan_FullCallback() 5.1.1.10 ManualRoi_Scan_SimpleCallback() 5.1.1.10 ManualRoi_Scan_ToJson()	73 75 75 76 77 78 78 79 79 80 80 81 81 81
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback() 5.1.1.7 ManualRoi_Cluster_FullCallback() 5.1.1.8 ManualRoi_Scan_FullCallback() 5.1.1.9 ManualRoi_Scan_FullCallback() 5.1.1.10 ManualRoi_Scan_SimpleCallback() 5.1.1.10 ManualRoi_Scan_ToJson() 5.2 include/BayesianClustering/Cluster.hpp File Reference 5.3 include/BayesianClustering/Configuration.hpp File Reference	73 75 75 76 77 78 78 79 80 80 81 81 81 82 82
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback() 5.1.1.7 ManualRoi_Cluster_FullCallback() 5.1.1.8 ManualRoi_Scan_FullCallback() 5.1.1.9 ManualRoi_Scan_FullCallback() 5.1.1.9 ManualRoi_Scan_ToJson() 5.2 include/BayesianClustering/Cluster.hpp File Reference 5.3 include/BayesianClustering/Configuration.hpp File Reference	73 75 75 76 77 78 78 79 79 80 80 81 81 82 82 82 83
5 File Documentation 5.1 include/BayesianClustering/API.hpp File Reference 5.1.1 Function Documentation 5.1.1.1 AutoRoi_Cluster_FullCallback() 5.1.1.2 AutoRoi_Cluster_SimpleCallback() 5.1.1.3 AutoRoi_Scan_FullCallback() 5.1.1.4 AutoRoi_Scan_SimpleCallback() 5.1.1.5 AutoRoi_Scan_ToJson() 5.1.1.6 ManualRoi_Cluster_FullCallback() 5.1.1.7 ManualRoi_Cluster_FullCallback() 5.1.1.8 ManualRoi_Scan_FullCallback() 5.1.1.9 ManualRoi_Scan_FullCallback() 5.1.1.10 ManualRoi_Scan_SimpleCallback() 5.1.1.10 ManualRoi_Scan_ToJson() 5.2 include/BayesianClustering/Cluster.hpp File Reference 5.3 include/BayesianClustering/Configuration.hpp File Reference	73 75 75 76 77 78 78 79 79 80 80 81 81 81 82 82 83 84

5.8 include/BayesianClustering/Rol.hpp File Reference
5.9 include/BayesianClustering/Rolproxy.hpp File Reference
5.10 include/Utilities/GSLInterpolator.hpp File Reference
5.11 include/Utilities/ListComprehension.hpp File Reference
5.11.1 Function Documentation
5.11.1.1 operator" () [1/3] 9
5.11.1.2 operator" () [2/3] 9
5.11.1.3 operator" () [3/3]
5.11.1.4 range()
5.12 include/Utilities/MemoryMonitoring.hpp File Reference
5.12.1 Function Documentation
5.12.1.1 mem_usage()
5.13 include/Utilities/ProgressBar.hpp File Reference
5.14 include/Utilities/Units.hpp File Reference
5.14.1 Function Documentation
5.14.1.1 operator""""_micrometer() [1/2]
5.14.1.2 operator""""_micrometer() [2/2]
5.14.1.3 operator""""_nanometer() [1/2]
5.14.1.4 operator""""_nanometer() [2/2]
5.14.1.5 StrToDist()
5.15 include/Utilities/Vectorize.hpp File Reference
5.15.1 Function Documentation
5.15.1.1 operator&&()
5.15.1.2 operator" " ()
5.15.2 Variable Documentation
5.15.2.1 Nthreads
5.16 src/BayesianClustering/API.cpp File Reference
5.16.1 Function Documentation
5.16.1.1 _FullClusterToSimpleCluster_()
5.16.1.2 _FullScanToSimpleScan_()
5.16.1.3 _ScanCallback_Json_()
5.16.1.4 AutoRoi_Cluster_FullCallback()
5.16.1.5 AutoRoi_Cluster_SimpleCallback()
5.16.1.6 AutoRoi_Scan_FullCallback()
5.16.1.7 AutoRoi_Scan_SimpleCallback()
5.16.1.8 AutoRoi_Scan_ToJson()
5.16.1.9 ManualRoi_Cluster_FullCallback()
5.16.1.10 ManualRoi_Cluster_SimpleCallback()
5.16.1.11 ManualRoi_Scan_FullCallback()
5.16.1.12 ManualRoi_Scan_SimpleCallback()
5.16.1.13 ManualRoi_Scan_ToJson()
5.17 src/BayesianClustering/Cluster.cpp File Reference 10

123

5.17.1 Function Documentation)8
5.17.1.1 normal_cdf())8
5.18 src/BayesianClustering/Configuration.cpp File Reference)9
5.19 src/BayesianClustering/Data.cpp File Reference)9
5.20 src/BayesianClustering/DataProxy.cpp File Reference	0
5.21 src/BayesianClustering/LocalizationFile.cpp File Reference	11
5.21.1 Function Documentation	11
5.21.1.1LoadCSV()	11
5.21.1.2RecursiveSearch()	2
5.22 src/BayesianClustering/Rol.cpp File Reference	2
5.23 src/BayesianClustering/Rolproxy.cpp File Reference	13
5.24 src/Cluster.cxx File Reference	3
5.24.1 Function Documentation	4
5.24.1.1 main()	4
5.24.1.2 ReportClusters()	5
5.25 src/PythonBindings/PythonBindings.cpp File Reference	15
5.25.1 Detailed Description	6
5.25.2 Macro Definition Documentation	17
5.25.2.1 ADAPT_CALLBACK_CONSTRUCTOR	7
5.25.2.2 ADAPTED_FN	17
5.25.2.3 EXPOSE_STRUCT	7
5.25.2.4 EXPOSE_VECTOR	8
5.25.2.5 FN	8
5.25.2.6 STRUCT_ARG	8
5.26 src/Scan.cxx File Reference	9
5.26.1 Function Documentation	9
5.26.1.1 main()	9
5.27 src/Utilities/GSLInterpolator.cpp File Reference	20
5.28 src/Utilities/ProgressBar.cpp File Reference	20
5.29 src/Utilities/Units.cpp File Reference	21
5.29.1 Function Documentation	21
5.29.1.1 StrToDist()	21
5.30 src/Utilities/Vectorize.cpp File Reference	22
5.30.1 Variable Documentation	22
5.30.1.1 Nthreads	22

Index

Chapter 1

Todo List

Member Data::CalculateLocalizationScore (const std::vector< Data > &aData, const double &R, const double &aArea) const

Remind myself how this works and what the difference is with above

Member Data::PreprocessLocalizationScores (std::vector< Data > &aData, const ScanConfiguration &a ← ScanConfig, const double &aArea)

Remind myself how this works and what the difference is with below

2 **Todo List**

Chapter 2

Class Index

2.1 Class List

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

AuxConfiguration	
Class for storing the auxilliary configuration parameters	7
Cluster	
A class representing a cluster	13
ClusterWrapper	
A struct for storing a result of an individual scan configuration	17
Data	
A class to store the raw data-points	19
DataProxy	
A light-weight proxy for the raw data-points	25
GSLInterpolator	
A utility wrapper around the GSL interpolator to give it a clean C++ interface	29
LocalizationFile	
A class to store the raw data-points	36
ManualRol	
A struct for storing the manual Rols	39
Cluster::Parameter	
A struct representing the cluster parameters	40
ProgressBar	
A utility progress-bar	42
ProgressTimer	
A utility code timer	44
Rol	
A class which holds the raw Rol data and global parameters	45
Rolproxy	
A lightweight wrapper for the Rol to store clusters for a given scan	52
ScanConfiguration	
Class for storing the scan configuration parameters	57
ScanEntry	
A struct for storing a result of an individual scan configuration	71
ScanConfiguration::tBounds	
A struct to store the bounds of a scan in either R or T	73

Class Index

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

include/BayesianClustering/API.hpp
include/BayesianClustering/Cluster.hpp
include/BayesianClustering/Configuration.hpp
include/BayesianClustering/Data.hpp
include/BayesianClustering/DataProxy.hpp
include/BayesianClustering/LocalizationFile.hpp
include/BayesianClustering/Precision.hpp
include/BayesianClustering/Rol.hpp
include/BayesianClustering/Rolproxy.hpp
include/Utilities/GSLInterpolator.hpp
include/Utilities/ListComprehension.hpp
include/Utilities/MemoryMonitoring.hpp
include/Utilities/ProgressBar.hpp
include/Utilities/Units.hpp
include/Utilities/Vectorize.hpp
src/Cluster.cxx
src/Scan.cxx
src/BayesianClustering/API.cpp
src/BayesianClustering/Cluster.cpp
src/BayesianClustering/Configuration.cpp
src/BayesianClustering/Data.cpp
src/BayesianClustering/DataProxy.cpp
src/BayesianClustering/LocalizationFile.cpp
src/BayesianClustering/Rol.cpp
src/BayesianClustering/Rolproxy.cpp
src/PythonBindings/PythonBindings.cpp
Self-contained sourcefile for producing python-bindings
src/Utilities/GSLInterpolator.cpp
src/Utilities/ProgressBar.cpp
src/Utilities/Units.cpp
ere/Utilities/Vectorize con

6 File Index

Chapter 4

Class Documentation

4.1 AuxConfiguration Class Reference

Class for storing the auxilliary configuration parameters.

```
#include <Configuration.hpp>
```

Public Member Functions

AuxConfiguration (int argc, char **argv)

Default constructor.

AuxConfiguration (const std::vector< std::string > &aArgs)

Constructor which parses the parameters when passed in as commandline arguments.

void SetValidate (const bool &aValidate)

Set whether to validate clusterization.

void SetInputFile (const std::string &aFileName)

Setter for the input file.

void SetOutputFile (const std::string &aFileName)

Setter for the output file.

void SetConfigFile (const std::string &aFileName)

Setter for the config file.

• const bool & validate () const

Getter for whether or not to run the validation on the clustering.

• const std::string & inputFile () const

Getter for the input file.

• const std::string & outputFile () const

Getter for the output file.

• const std::string & configFile () const

Getter for the config file.

• const double & ClusterR () const

Getter for the R value for a clusterization pass.

const double & ClusterT () const

Getter for the T value for a clusterization pass.

Public Attributes

bool mValidate

Whether or not to run the validation on the clustering.

• std::string mInputFile

The input Rol file.

• std::string mOutputFile

The output file.

• std::string mConfigFile

The config file.

· double mClusterR

The value of R for clustering.

· double mClusterT

The value of T for clustering.

Private Member Functions

void FromVector (const std::vector < std::string > &aArgs)
 Parse the parameters when passed in as commandline arguments.

4.1.1 Detailed Description

Class for storing the auxilliary configuration parameters.

Definition at line 263 of file Configuration.hpp.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 AuxConfiguration() [1/2]

```
AuxConfiguration::AuxConfiguration (
    int argc,
    char ** argv )
```

Default constructor.

Constructor which parses the parameters when passed in as commandline arguments

Parameters

argc	The number of commandline arguments
argv	The commandline arguments

Definition at line 214 of file Configuration.cpp.

References FromVector().

4.1.2.2 AuxConfiguration() [2/2]

```
AuxConfiguration::AuxConfiguration ( const std::vector< std::string > & aArgs)
```

Constructor which parses the parameters when passed in as commandline arguments.

Parameters

aArgs	The commandline arguments
	, ,

Definition at line 223 of file Configuration.cpp.

References FromVector().

4.1.3 Member Function Documentation

4.1.3.1 ClusterR()

```
const double& AuxConfiguration::ClusterR ( ) const [inline]
```

Getter for the R value for a clusterization pass.

Returns

The R value for a clusterization pass

Definition at line 323 of file Configuration.hpp.

References mClusterR.

Referenced by main().

4.1.3.2 ClusterT()

```
const double& AuxConfiguration::ClusterT ( ) const [inline]
```

Getter for the T value for a clusterization pass.

Returns

The T value for a clusterization pass

Definition at line 330 of file Configuration.hpp.

References mClusterT.

Referenced by main().

4.1.3.3 configFile()

```
const std::string& AuxConfiguration::configFile ( ) const [inline]
```

Getter for the config file.

Returns

The name of the config file

Definition at line 316 of file Configuration.hpp.

References mConfigFile.

Referenced by main().

4.1.3.4 FromVector()

```
void AuxConfiguration::FromVector ( const \ std::vector < std::string > \& \ aArgs \ ) \quad [private]
```

Parse the parameters when passed in as commandline arguments.

Parameters

aArgs The commandline arguments

Definition at line 262 of file Configuration.cpp.

References mClusterR, mClusterT, Nthreads, SetConfigFile(), SetInputFile(), SetOutputFile(), SetValidate(), and StrToDist().

Referenced by AuxConfiguration().

4.1.3.5 inputFile()

```
const std::string& AuxConfiguration::inputFile ( ) const [inline]
```

Getter for the input file.

Returns

The name of the input Rol file

Definition at line 302 of file Configuration.hpp.

References mInputFile.

Referenced by main().

4.1.3.6 outputFile()

```
const std::string& AuxConfiguration::outputFile ( ) const [inline]
```

Getter for the output file.

Returns

The name of the output file

Definition at line 309 of file Configuration.hpp.

References mOutputFile.

Referenced by main().

4.1.3.7 SetConfigFile()

Setter for the config file.

Parameters

aFileName	The name of the file
-----------	----------------------

Definition at line 250 of file Configuration.cpp.

References mConfigFile.

Referenced by FromVector().

4.1.3.8 SetInputFile()

Setter for the input file.

Parameters

aFileName T	he name of the file
-------------	---------------------

Definition at line 237 of file Configuration.cpp.

References mInputFile.

Referenced by FromVector().

4.1.3.9 SetOutputFile()

Setter for the output file.

Parameters

ahileName The name of the file	aFileName	The name of the file
----------------------------------	-----------	----------------------

Definition at line 243 of file Configuration.cpp.

References mOutputFile.

Referenced by FromVector().

4.1.3.10 SetValidate()

Set whether to validate clusterization.

Parameters

aValidate Whether to validate clusterization
--

Definition at line 231 of file Configuration.cpp.

References mValidate.

Referenced by FromVector().

4.1.3.11 validate()

```
const bool& AuxConfiguration::validate ( ) const [inline]
```

Getter for whether or not to run the validation on the clustering.

Returns

Whether or not to run the validation on the clustering

Definition at line 295 of file Configuration.hpp.

References mValidate.

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

- include/BayesianClustering/Configuration.hpp
- src/BayesianClustering/Configuration.cpp

4.2 Cluster Class Reference

A class representing a cluster.

#include <Cluster.hpp>

Collaboration diagram for Cluster:



Classes

struct Parameter

A struct representing the cluster parameters.

Public Member Functions

• Cluster (const std::size_t &aParamSize)

Default constructor.

Cluster (const Data &aData, const std::vector< double > &aSigmabins2)

Construct a cluster from a single data-point.

Cluster (const Cluster &aOther)=delete

Deleted copy constructor.

• Cluster & operator= (const Cluster &aOther)=delete

Deleted assignment operator.

• Cluster (Cluster &&aOther)=default

Default move constructor.

• Cluster & operator= (Cluster &&aOther)=default

Default move-assignment constructor.

∼Cluster ()

Default destructor.

• Cluster & operator+= (const Cluster &aOther)

Add another cluster to this one.

Cluster * GetParent ()

Get a pointer to this cluster's ultimate parent.

void UpdateLogScore (const ScanConfiguration &aScanConfig)

Update log-probability after a scan.

Public Attributes

• std::vector< Parameter > mParams

The collection of parameters, each corresponding to a different sigma hypothesis.

• std::size_t mClusterSize

The number of points in the current cluster.

• std::size_t mLastClusterSize

The number of points in the cluster on the previous scan iteration.

• PRECISION mClusterScore

The log-probability of the current cluster.

· Cluster * mParent

A pointer to the immediate parent of the current cluster.

std::vector < Data * > mData

List of points in the cluster after clustering.

4.2.1 Detailed Description

A class representing a cluster.

Definition at line 16 of file Cluster.hpp.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Cluster() [1/4]

Default constructor.

Parameters

```
aParamSize The number of sigma-bins
```

Definition at line 93 of file Cluster.cpp.

4.2.2.2 Cluster() [2/4]

Construct a cluster from a single data-point.

Parameters

aData	A data-point with which to initialize the cluster
aSigmabins2	The sigma-bins for initializing clusters

Definition at line 99 of file Cluster.cpp.

References mParams, Data::r2, Data::s, Data::x, and Data::y.

4.2.2.3 Cluster() [3/4]

Deleted copy constructor.

Parameters

aOther	Anonymous argument

4.2.2.4 Cluster() [4/4]

Default move constructor.

Parameters

aOther
aOther

4.2.3 Member Function Documentation

4.2.3.1 GetParent()

```
Cluster * Cluster::GetParent ( )
```

Get a pointer to this cluster's ultimate parent.

Returns

A pointer to this cluster's ultimate parent

Definition at line 165 of file Cluster.cpp.

References GetParent(), and mParent.

Referenced by DataProxy::GetCluster(), and GetParent().

4.2.3.2 operator+=()

Add another cluster to this one.

Parameters

Returns

Reference to this, for chaining calls

Definition at line 155 of file Cluster.cpp.

References mClusterSize, and mParams.

4.2.3.3 operator=() [1/2]

Default move-assignment constructor.

Returns

Reference to this, for chaining calls

Parameters

aOther	Anonymous argument

4.2.3.4 operator=() [2/2]

Deleted assignment operator.

Returns

Reference to this, for chaining calls

Parameters

4.2.3.5 UpdateLogScore()

Update log-probability after a scan.

Parameters

aScanConfig	The configuration parameters for the scan	
-------------	---	--

Definition at line 124 of file Cluster.cpp.

References ScanConfiguration::log_probability_sigma(), mClusterScore, mClusterSize, mLastClusterSize, m \leftarrow Params, and ScanConfiguration::sigmabins().

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

- include/BayesianClustering/Cluster.hpp
- src/BayesianClustering/Cluster.cpp

4.3 ClusterWrapper Struct Reference

A struct for storing a result of an individual scan configuration.

```
#include <API.hpp>
```

Public Member Functions

• bool operator== (const ClusterWrapper &aOther)

Equality operator required by boost python.

Public Attributes

std::size_t localizations

The number of localizations in the cluster.

· long double area

The area of the spanning convex hull.

• long double perimeter

The perimeter of the spanning convex hull.

· double centroid_x

The x-position of the centroid.

· double centroid_y

The y-position of the centroid.

4.3.1 Detailed Description

A struct for storing a result of an individual scan configuration.

Definition at line 44 of file API.hpp.

4.3.2 Member Function Documentation

4.3.2.1 operator==()

Equality operator required by boost python.

Returns

Whether we are equal to the other

Parameters

aOther Another ClusterWrapper to compare against

Definition at line 54 of file API.hpp.

References centroid_x, and centroid_y.

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

• include/BayesianClustering/API.hpp

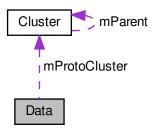
4.4 Data Class Reference 19

4.4 Data Class Reference

A class to store the raw data-points.

#include <Data.hpp>

Collaboration diagram for Data:



Public Member Functions

• Data (const PRECISION &aX, const PRECISION &aY, const PRECISION &aS)

Constructor.

Data (const Data &aOther)=delete

Deleted copy constructor.

• Data & operator= (const Data &aOther)=delete

Deleted assignment operator.

Data (Data &&aOther)=default

Default move constructor.

• Data & operator= (Data &&aOther)=default

Default move-assignment constructor.

virtual ~Data ()

Destructor.

• bool operator< (const Data &aOther) const

Comparison operator for sorting data-points by distance from the origin.

• PRECISION dR2 (const Data &aOther) const

Return the squared-distance of this data-points from another.

PRECISION dR (const Data &aOther) const

Return the distance of this data-points from another.

PRECISION dPhi (const Data &aOther) const

Return the angle between this data-points and another.

All the necessary pre-processing to get this data-point ready for an RT-scan.

void PreprocessLocalizationScores (std::vector < Data > &aData, const ScanConfiguration &aScanConfig, const double &aArea)

Calculate the localization score from the local neighbourhood.

PRECISION CalculateLocalizationScore (const std::vector < Data > &aData, const double &R, const double &AArea) const

Calculate the localization score from the local neighbourhood.

Public Attributes

• PRECISION x

The x-position of the data-point.

· PRECISION y

The y-position of the data-point.

• PRECISION S

The sigma of the data-point.

PRECISION r2

The squared radial distance of the data-point.

• PRECISION r

The radial distance of the data-point.

PRECISION phi

The phi-position of the data-point.

• std::vector< PRECISION > mLocalizationScores

The locaalization scores, one per R-bin.

• std::vector< std::pair< PRECISION, std::size_t >> mNeighbours

The list of neighbours as a pair of squared-distance and index into the list of points.

• Cluster * mProtoCluster

A cluster containing only this data-point.

4.4.1 Detailed Description

A class to store the raw data-points.

Definition at line 17 of file Data.hpp.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 Data() [1/3]

Constructor.

Parameters

аX	The x-position of the data-point in algorithm units
aY	The y-position of the data-point in algorithm units
aS	The sigma of the data-point in algorithm units

Definition at line 14 of file Data.cpp.

4.4 Data Class Reference 21

4.4.2.2 Data() [2/3]

Deleted copy constructor.

Parameters

aOther Anonymous argument	i
---------------------------	---

4.4.2.3 Data() [3/3]

Default move constructor.

Parameters

aOther	Anonymous argument
--------	--------------------

4.4.3 Member Function Documentation

4.4.3.1 CalculateLocalizationScore()

Calculate the localization score from the local neighbourhood.

Todo Remind myself how this works and what the difference is with above

Parameters

aData	?
R	?
aArea	The area of the window for normalizing the log score

Returns

The localization score

Definition at line 106 of file Data.cpp.

References mNeighbours.

4.4.3.2 dPhi()

Return the angle between this data-points and another.

Returns

The angle between this data-points and another

Parameters

ſ	aOther	A data-point to compare against	1
---	--------	---------------------------------	---

Definition at line 71 of file Data.hpp.

References phi.

4.4.3.3 dR()

Return the distance of this data-points from another.

Returns

The distance of this data-points from another

Parameters

aOther	A data-point to compare against
--------	---------------------------------

Definition at line 63 of file Data.hpp.

References dR2().

4.4 Data Class Reference 23

4.4.3.4 dR2()

Return the squared-distance of this data-points from another.

Returns

The squared-distance of this data-points from another

Parameters

Definition at line 54 of file Data.hpp.

References x, and y.

Referenced by dR().

4.4.3.5 operator<()

Comparison operator for sorting data-points by distance from the origin.

Returns

Whether this data-point is closer to the origin than another

Parameters

aOther	A data-point to compare against

Definition at line 46 of file Data.hpp.

References r.

4.4.3.6 operator=() [1/2]

Deleted assignment operator.

Returns

Reference to this, for chaining calls

Parameters

```
aOther Anonymous argument
```

4.4.3.7 operator=() [2/2]

Default move-assignment constructor.

Returns

Reference to this, for chaining calls

Parameters

aOther Anonymo	us argument
----------------	-------------

4.4.3.8 Preprocess()

```
void Data::Preprocess (
    std::vector< Data > & aData,
    const std::size_t & aIndex,
    const double & aMax2R,
    const double & aMax2R2,
    const std::vector< double > & aSigmabins2,
    ProgressBar & aProgressBar )
```

All the necessary pre-processing to get this data-point ready for an RT-scan.

Parameters

aData	The collection of data-points
alndex	The index of the current data-point
aMax2R	Twice the maximum radius out to which we will cluster
aMax2R2	Square of twice the maximum radius out to which we will cluster
aSigmabins2	The sigma-bins for initializing clusters
aProgressBar	The progress bar to update

Definition at line 28 of file Data.cpp.

4.4.3.9 PreprocessLocalizationScores()

```
void Data::PreprocessLocalizationScores (
    std::vector< Data > & aData,
    const ScanConfiguration & aScanConfig,
    const double & aArea )
```

Calculate the localization score from the local neighbourhood.

Todo Remind myself how this works and what the difference is with below

Parameters

aData	?
aScanConfig	The configuration parameters for the scan
aArea	The area of the window for normalizing the log score

Definition at line 77 of file Data.cpp.

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

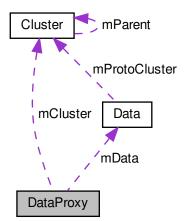
- include/BayesianClustering/Data.hpp
- src/BayesianClustering/Data.cpp

4.5 DataProxy Class Reference

A light-weight proxy for the raw data-points.

```
#include <DataProxy.hpp>
```

Collaboration diagram for DataProxy:



Public Member Functions

DataProxy (Data &aData)

Default constructor.

• DataProxy (const DataProxy &aOther)=delete

Deleted copy constructor.

• DataProxy & operator= (const DataProxy &aOther)=delete

Deleted assignment operator.

• DataProxy (DataProxy &&aOther)=default

Default move constructor.

• DataProxy & operator= (DataProxy &&aOther)=default

Default move-assignment constructor.

void Clusterize (const PRECISION &a2R2, Rolproxy &aRol)

Entry point clusterization function - a new cluster will be created.

• void Clusterize (const PRECISION &a2R2, Rolproxy &aRol, Cluster *aCluster, const std::size_t &d=0)

Recursive clusterization function.

Cluster * GetCluster ()

Get a pointer to this data-proxy's ultimate parent cluster (or null if unclustered.

Public Attributes

• Data * mData

The data-point for which this is the proxy.

· Cluster * mCluster

This data-proxy's immediate parent cluster.

bool mExclude

Whether this data-point is to be included in the clusterization.

4.5.1 Detailed Description

A light-weight proxy for the raw data-points.

Definition at line 18 of file DataProxy.hpp.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 DataProxy() [1/3]

Default constructor.

Parameters

aData The data-point for which this is the proxy

Definition at line 17 of file DataProxy.cpp.

4.5.2.2 DataProxy() [2/3]

Deleted copy constructor.

Parameters

aOther	Anonymous argument
--------	--------------------

4.5.2.3 DataProxy() [3/3]

Default move constructor.

Parameters

aOther Ai	nonymous argument
-----------	-------------------

4.5.3 Member Function Documentation

4.5.3.1 Clusterize() [1/2]

Entry point clusterization function - a new cluster will be created.

Parameters

a2R2	The clusterization radius
aRol	The Rol-proxy in which we are running

Definition at line 23 of file DataProxy.cpp.

References mCluster, Rolproxy::mClusters, mData, mExclude, Cluster::mParams, and Data::mProtoCluster.

Referenced by Clusterize().

4.5.3.2 Clusterize() [2/2]

Recursive clusterization function.

Parameters

a2R2	The clusterization radius
aRol	The Rol-proxy in which we are running
aCluster	The cluster we are building
d	The recursion depth

Definition at line 34 of file DataProxy.cpp.

References Clusterize(), GetCluster(), Rolproxy::GetData(), mCluster, Cluster::mClusterSize, mData, mExclude, Data::mNeighbours, Cluster::mParent, Data::mProtoCluster, and RECURSION_LIMIT.

4.5.3.3 GetCluster()

```
Cluster* DataProxy::GetCluster ( ) [inline]
```

Get a pointer to this data-proxy's ultimate parent cluster (or null if unclustered.

Returns

A pointer to this data-proxy's ultimate parent cluster

Definition at line 53 of file DataProxy.hpp.

References Cluster::GetParent(), and mCluster.

Referenced by Clusterize().

4.5.3.4 operator=() [1/2]

Deleted assignment operator.

Returns

Reference to this, for chaining calls

Parameters

aOther Anonymous argu	ıment
-----------------------	-------

4.5.3.5 operator=() [2/2]

Default move-assignment constructor.

Returns

Reference to this, for chaining calls

Parameters

aOther	Anonymous argument
--------	--------------------

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

- include/BayesianClustering/DataProxy.hpp
- src/BayesianClustering/DataProxy.cpp

4.6 GSLInterpolator Class Reference

A utility wrapper around the GSL interpolator to give it a clean C++ interface.

```
#include <GSLInterpolator.hpp>
```

Public Member Functions

- $\bullet \ \ \textbf{GSLInterpolator} \ (\textbf{const} \ \textbf{gsl_interp_type} \ *\textbf{type}, \ \textbf{const} \ \textbf{unsigned} \ \textbf{int} \ \textbf{\&ndata}) \\$
 - Empty splice constructor.
- GSLInterpolator (const gsl_interp_type *type, const std::vector< double > &x, const std::vector< double > &y)

Initialised splice constructor.

virtual ∼GSLInterpolator ()

Destructor.

GSLInterpolator (const GSLInterpolator &aOther)=delete

Deleted copy constructor.

• GSLInterpolator & operator= (const GSLInterpolator &aOther)=delete

Deleted assignment operator.

GSLInterpolator (GSLInterpolator &&aOther)=default

Default move constructor.

• GSLInterpolator & operator= (GSLInterpolator &&aOther)=default

Default move-assignment constructor.

bool SetData (const std::vector< double > &x, const std::vector< double > &y)

Set the spline data points.

bool SetData (const unsigned int &ndata, const double *x, const double *y)

Set the spline data points.

double Evaluate (const std::function < int(double &) > &aFunction, const std::string &aName)

Utility function that runs the GSL function that has been wrapped in a lambda below.

• double Eval (const double &x)

Evaluate the spline at the given x.

double Deriv (const double &x)

The first derivative of the spline at the given x.

double Deriv2 (const double &x)

The second derivative of the spline at the given x.

double Integ (const double &a, const double &b)

The integral over the spline between two bounds.

Private Attributes

· unsigned int nErrors

An error counter to suppress excess messages.

gsl_interp_accel * fAccel

Underlying GSL machinery.

• gsl spline * fSpline

Underlying GSL machinery for the spline itself.

const gsl_interp_type * fInterpType

Underlying GSL machinery for the interpolation type.

4.6.1 Detailed Description

A utility wrapper around the GSL interpolator to give it a clean C++ interface.

Definition at line 19 of file GSLInterpolator.hpp.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 GSLInterpolator() [1/4]

Empty splice constructor.

Parameters

type	The spline type
ndata	The number of points that will be added to the spline

Definition at line 9 of file GSLInterpolator.cpp.

References fInterpType, and fSpline.

4.6.2.2 GSLInterpolator() [2/4]

Initialised splice constructor.

Parameters

type	The spline type
Х	The points on the x-axis
У	The points on the y-axis

Definition at line 19 of file GSLInterpolator.cpp.

References fInterpType, fSpline, and SetData().

4.6.2.3 GSLInterpolator() [3/4]

Deleted copy constructor.

Parameters

aOther Anonymous argument

4.6.2.4 GSLInterpolator() [4/4]

```
GSLInterpolator::GSLInterpolator (
GSLInterpolator && aOther ) [default]
```

Default move constructor.

Parameters

aOther	Anonymous argument
--------	--------------------

4.6.3 Member Function Documentation

4.6.3.1 Deriv()

The first derivative of the spline at the given x.

Parameters

x The x-coordinate at which to evaluate the derivative

Returns

The first derivative of the spline at the given x-coordinate

Definition at line 104 of file GSLInterpolator.hpp.

References Evaluate(), fAccel, and fSpline.

4.6.3.2 Deriv2()

The second derivative of the spline at the given x.

Parameters

x The x-coordinate at which to evaluate the derivative

Returns

The second derivative of the spline at the given x-coordinate

Definition at line 114 of file GSLInterpolator.hpp.

References Evaluate(), fAccel, and fSpline.

4.6.3.3 Eval()

Evaluate the spline at the given x.

Parameters

X	The x-coordinate at which to evaluate the spline
---	--

Returns

The value of the spline at the given x-coordinate

Definition at line 94 of file GSLInterpolator.hpp.

References Evaluate(), fAccel, and fSpline.

Referenced by ScanConfiguration::FromVector().

4.6.3.4 Evaluate()

```
double GSLInterpolator::Evaluate (  {\tt const \ std::function<\ int(double\ \&)\ >\ \&\ aFunction,}   {\tt const \ std::string\ \&\ aName\ )\ [inline]}
```

Utility function that runs the GSL function that has been wrapped in a lambda below.

Parameters

aFunction	A lambda that will be evaluated
aName	The operation name for the debugging messages

Returns

The interpolated value

Definition at line 75 of file GSLInterpolator.hpp.

References fAccel, and nErrors.

Referenced by Deriv(), Deriv2(), Eval(), and Integ().

4.6.3.5 Integ()

The integral over the spline between two bounds.

Parameters

а	The lower bound of the integral
b	The upper bound of the integral

Returns

The integral over the spline between a and b

Definition at line 125 of file GSLInterpolator.hpp.

References Evaluate(), fAccel, and fSpline.

4.6.3.6 operator=() [1/2]

Deleted assignment operator.

Returns

Reference to this, for chaining calls

Parameters

aOther	Anonymous argument

4.6.3.7 operator=() [2/2]

Default move-assignment constructor.

Returns

Reference to this, for chaining calls

Parameters

aOther Anonymous argumer	it
--------------------------	----

4.6.3.8 SetData() [1/2]

Set the spline data points.

Parameters

	The x-coordinates of the datapoints
У	The y-coordinates of the datapoints

Returns

success or fail

Definition at line 58 of file GSLInterpolator.hpp.

Referenced by GSLInterpolator().

4.6.3.9 SetData() [2/2]

Set the spline data points.

Parameters

ndata	The number of data points
Х	Pointer to the first element of an array of x-coordinates
У	Pointer to the first element of an array of y-coordinates

Returns

success or fail

Definition at line 38 of file GSLInterpolator.cpp.

References fAccel, fInterpType, fSpline, and nErrors.

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

- include/Utilities/GSLInterpolator.hpp
- src/Utilities/GSLInterpolator.cpp

4.7 LocalizationFile Class Reference

A class to store the raw data-points.

```
#include <LocalizationFile.hpp>
```

Public Member Functions

• LocalizationFile (const std::string &aFilename)

Constructor.

LocalizationFile (const LocalizationFile &aOther)=delete

Deleted copy constructor.

• LocalizationFile & operator= (const LocalizationFile &aOther)=delete

Deleted assignment operator.

LocalizationFile (LocalizationFile &&aOther)=default

Default move constructor.

• LocalizationFile & operator= (LocalizationFile &&aOther)=default

Default move-assignment constructor.

∼LocalizationFile ()=default

Default destructor.

void ExtractRols (const std::function < void(Rol &) > &aCallback) const

Automatically extract the Rols.

void ExtractRols (const ManualRol &aRol, const std::function < void(Rol &) > &aCallback) const
 Manually extract an Rol.

Private Attributes

std::vector < Data > mData
 The localizations in the file.

4.7.1 Detailed Description

A class to store the raw data-points.

Definition at line 25 of file LocalizationFile.hpp.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 LocalizationFile() [1/3]

```
\label{localizationFile:LocalizationFile (const std::string \& aFilename )} aFilename \ )
```

Constructor.

Parameters

aFilename The name of the localizations file
--

Definition at line 68 of file LocalizationFile.cpp.

References __LoadCSV__(), mData, Nthreads, and range().

4.7.2.2 LocalizationFile() [2/3]

Deleted copy constructor.

Parameters

aOther Anonymous argument

4.7.2.3 LocalizationFile() [3/3]

```
\label{localizationFile:LocalizationFile (LocalizationFile && aOther ) [default]} \end{substitute}
```

Default move constructor.

Parameters

```
aOther Anonymous argument
```

4.7.3 Member Function Documentation

4.7.3.1 ExtractRols() [1/2]

Manually extract an Rol.

Parameters

aRol	The manual Rol window
aCallback	A handler for each Rol found

Definition at line 268 of file LocalizationFile.cpp.

References ManualRol::height, mData, Rol::SetCentre(), Rol::SetWidth(), ManualRol::width, ManualRol::x, and ManualRol::y.

4.7.3.2 ExtractRols() [2/2]

```
void LocalizationFile::ExtractRoIs ( const std::function< void(RoI &) > & aCallback ) const
```

Automatically extract the Rols.

Parameters

aCallback A handler for each Rol fo	ound	
-------------------------------------	------	--

Definition at line 119 of file LocalizationFile.cpp.

References __RecursiveSearch__(), mData, Rol::SetCentre(), and Rol::SetWidth().

 $Referenced\ by\ AutoRoi_Cluster_FullCallback(),\ AutoRoi_Scan_FullCallback(),\ AutoRoi_Scan_SimpleCallback(),\ ManualRoi_Cluster_FullCallback(),\ ManualRoi_Scan_FullCallback(),\ and\ ManualRoi_Scan_SimpleCallback().$

4.7.3.3 operator=() [1/2]

Deleted assignment operator.

Returns

Reference to this, for chaining calls

Parameters

aOther	Anonymous argument
--------	--------------------

4.7.3.4 operator=() [2/2]

```
\label{localizationFile} LocalizationFile::operator= \mbox{ (} \\ LocalizationFile \&\& aOther \mbox{ ) [} \mbox{ [default]}
```

Default move-assignment constructor.

Returns

Reference to this, for chaining calls

Parameters

aOther Anonymous argun	nent
------------------------	------

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

- include/BayesianClustering/LocalizationFile.hpp
- src/BayesianClustering/LocalizationFile.cpp

4.8 ManualRol Struct Reference

A struct for storing the manual Rols.

```
#include <LocalizationFile.hpp>
```

Public Attributes

double x

The x-centre of the Rol.

· double y

The y-centre of the Rol.

· double width

The width of the Rol.

· double height

The height of the Rol.

4.8.1 Detailed Description

A struct for storing the manual Rols.

Definition at line 15 of file LocalizationFile.hpp.

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

• include/BayesianClustering/LocalizationFile.hpp

4.9 Cluster::Parameter Struct Reference

A struct representing the cluster parameters.

```
#include <Cluster.hpp>
```

Public Member Functions

• Parameter ()

Default constructor.

• Parameter & operator+= (const Parameter &aOther)

Add another set of parameters to this set.

• double log_score () const

Convert the parameters to a log-probability.

• double alt_log_score () const

Sean's alternative function to calculate the log-score using only the A's and B's as per the original paper for debugging.

Public Attributes

PRECISION A

Parameter A defined in the math.

PRECISION Bx

Parameter Bx defined in the math.

PRECISION By

Parameter By defined in the math.

PRECISION C

Parameter C defined in the math.

PRECISION logF

Parameter logF defined in the math.

PRECISION weightedCentreX

Parameters added by Sean for validation.

PRECISION weightedCentreY

Parameters added by Sean for validation.

PRECISION S2

Parameters added by Sean for validation.

4.9.1 Detailed Description

A struct representing the cluster parameters.

Definition at line 21 of file Cluster.hpp.

4.9.2 Member Function Documentation

4.9.2.1 alt_log_score()

```
double Cluster::Parameter::alt_log_score ( ) const
```

Sean's alternative function to calculate the log-score using only the A's and B's as per the original paper for debugging.

Returns

the log-probability of this set of cluster parameters

Definition at line 47 of file Cluster.cpp.

References normal_cdf().

4.9.2.2 log_score()

```
double Cluster::Parameter::log_score ( ) const
```

Convert the parameters to a log-probability.

Returns

the log-probability of this set of cluster parameters

Definition at line 72 of file Cluster.cpp.

References normal cdf().

4.9.2.3 operator+=()

Add another set of parameters to this set.

Parameters

aOther	Another set of parameters to add to this set
actinoi	, mother out of parameters to add to the out

Returns

Reference to this, for chaining calls

Definition at line 37 of file Cluster.cpp.

References A, Bx, By, C, and logF.

The documentation for this struct was generated from the following files:

- include/BayesianClustering/Cluster.hpp
- src/BayesianClustering/Cluster.cpp

4.10 ProgressBar Class Reference

A utility progress-bar.

```
#include <ProgressBar.hpp>
```

Public Member Functions

ProgressBar (const std::string &aLabel, const uint32_t &aMax)

Constructor.

virtual ∼ProgressBar ()

Destructor.

void operator++ ()

Postfix increment.

void operator++ (int aDummy)

Prefix increment.

Private Member Functions

• void print ()

Update the screen.

Private Attributes

float mBlockSize

The size of each increment.

· float mNextThreshold

The next threshold at which we will write a block to stdout.

• std::size t mCount

The number of times we have incremented.

• std::chrono::high_resolution_clock::time_point mStart

A timer for end-of-task stats.

• std::mutex mMutex

A mutex for multi-threaded updates.

std::string mLabel

The label for the start of the line.

• std::size_t mPercent

The current progress.

4.10.1 Detailed Description

A utility progress-bar.

Definition at line 8 of file ProgressBar.hpp.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 ProgressBar()

Constructor.

Parameters

aLabel	A description of the task being timed
aMax	The number of calls equalling 100%

Definition at line 8 of file ProgressBar.cpp.

References mLabel, and print().

4.10.3 Member Function Documentation

4.10.3.1 operator++()

Prefix increment.

Parameters

aDummy	Anonymous argument
abuilling	7 monymous argument

Definition at line 39 of file ProgressBar.cpp.

References operator++().

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

- include/Utilities/ProgressBar.hpp
- src/Utilities/ProgressBar.cpp

4.11 ProgressTimer Struct Reference

```
A utility code timer.
```

```
#include <ProgressBar.hpp>
```

Public Member Functions

• ProgressTimer (const std::string &aLabel)

Constructor.

virtual ∼ProgressTimer ()

Destructor.

Public Attributes

std::chrono::high_resolution_clock::time_point mStart
 A timer for end-of-task stats.

4.11.1 Detailed Description

A utility code timer.

Definition at line 49 of file ProgressBar.hpp.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 ProgressTimer()

Constructor.

Parameters

aLabel	A description of the task being timed

Definition at line 55 of file ProgressBar.cpp.

The documentation for this struct was generated from the following files:

4.12 Rol Class Reference 45

- include/Utilities/ProgressBar.hpp
- src/Utilities/ProgressBar.cpp

4.12 Rol Class Reference

A class which holds the raw Rol data and global parameters.

```
#include <RoI.hpp>
```

Public Member Functions

Rol (std::vector < Data > &&aData, const std::size_t &ald=0)

Default Constructor.

Rol (const Rol &aOther)=delete

Deleted copy constructor.

• Rol & operator= (const Rol &aOther)=delete

Deleted assignment operator.

• Rol (Rol &&aOther)=default

Default move constructor.

• ∼RoI ()

Default destructor.

• Rol & operator= (Rol &&aOther)=default

Default move-assignment constructor.

void Preprocess (const double &aMaxR, const std::vector< double > &aSigmabins2)

All the necessary pre-processing to get the Rol ready for an RT-scan.

void ScanRT (const ScanConfiguration &aScanConfig, const std::function < void(Rolproxy &, const double &, const double &) > &aCallback)

Run the scan.

void Clusterize (const double &R, const double &T, const std::function < void(Rolproxy &) > &aCallback)

Run clusterization for a specific choice of R and T.

void SetCentre (const double &aPhysicalCentreX, const double &aPhysicalCentreY)

Setter for the centre of the scan window.

void SetWidth (const double &aWidthX, const double &aWidthY)

Setter for the size of the Rol window.

• double getCentreX () const

Getter for the x-coordinate of the physical centre.

double getCentreY () const

Getter for the y-coordinate of the physical centre.

• double getWidthX () const

Getter for the width of the ROI window.

• double getWidthY () const

Getter for the height of the ROI window.

• double getArea () const

Getter for the height of the ROI window.

Public Attributes

std::size_t mld

A unique identifier.

std::vector < Data > mData

The collection of raw data points.

Private Attributes

· double mPhysicalCentreX

The x-coordinate of the centre of the window in physical units.

· double mPhysicalCentreY

The y-coordinate of the centre of the window in physical units.

• double mWidthX

The width of the window in the x-direction in physical units.

double mWidthY

The width of the window in the y-direction in physical units.

· double mArea

The area of the window in physical units.

4.12.1 Detailed Description

A class which holds the raw Rol data and global parameters.

Definition at line 17 of file Rol.hpp.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 Rol() [1/3]

Default Constructor.

Parameters

aData	The set of data-points in the Rol
ald	A unique identifier

Definition at line 17 of file Rol.cpp.

References mData.

4.12 Rol Class Reference 47

4.12.2.2 Rol() [2/3]

Deleted copy constructor.

Parameters

4.12.2.3 Rol() [3/3]

Default move constructor.

Parameters

aOther Anonymous argum	ent
------------------------	-----

4.12.3 Member Function Documentation

4.12.3.1 Clusterize()

Run clusterization for a specific choice of R and T.

Parameters

R	The R parameter for clusterization
T	The T parameter for clusterization
aCallback	A callback for the clusterization results

Definition at line 58 of file Rol.cpp.

References Rolproxy::Clusterize(), and Preprocess().

 $Referenced \ by \ AutoRoi_Cluster_FullCallback(), \ and \ ManualRoi_Cluster_FullCallback().$

4.12.3.2 getArea()

```
double RoI::getArea ( ) const [inline]
```

Getter for the height of the ROI window.

Returns

The height of the ROI window

Definition at line 102 of file Rol.hpp.

References mArea.

Referenced by Rolproxy::Clusterize(), and ScanRT().

4.12.3.3 getCentreX()

```
double RoI::getCentreX ( ) const [inline]
```

Getter for the x-coordinate of the physical centre.

Returns

The x-coordinate of the physical centre

Definition at line 75 of file Rol.hpp.

References mPhysicalCentreX.

4.12.3.4 getCentreY()

```
double RoI::getCentreY ( ) const [inline]
```

Getter for the y-coordinate of the physical centre.

Returns

The y-coordinate of the physical centre

Definition at line 81 of file Rol.hpp.

References mPhysicalCentreY.

4.12 Rol Class Reference 49

4.12.3.5 getWidthX()

```
double RoI::getWidthX ( ) const [inline]
```

Getter for the width of the ROI window.

Returns

The width of the ROI window

Definition at line 88 of file Rol.hpp.

References mWidthX.

4.12.3.6 getWidthY()

```
double RoI::getWidthY ( ) const [inline]
```

Getter for the height of the ROI window.

Returns

The height of the ROI window

Definition at line 95 of file Rol.hpp.

References mWidthY.

4.12.3.7 operator=() [1/2]

Deleted assignment operator.

Returns

Reference to this, for chaining calls

Parameters

aOther	Anonymous argument
aUlilei	Anonymous ard

4.12.3.8 operator=() [2/2]

```
RoI& RoI::operator= ( \label{RoI} \mbox{RoI \&\& aOther )} \quad [\mbox{default}]
```

Default move-assignment constructor.

Returns

Reference to this, for chaining calls

Parameters

aOther Anonymous argument	
---------------------------	--

4.12.3.9 Preprocess()

All the necessary pre-processing to get the Rol ready for an RT-scan.

Parameters

aMaxR	The maximum radius out to which we should pre-process
aSigmabins2	The number of sigma bins

Definition at line 35 of file Rol.cpp.

References mData, and range().

Referenced by Clusterize(), and ScanRT().

4.12.3.10 ScanRT()

Run the scan.

Parameters

aScanConfig	The configuration parameters for the scan
aCallback	A callback for each RT-scan result

4.12 Rol Class Reference 51

Definition at line 44 of file Rol.cpp.

References ScanConfiguration::tBounds::bins, getArea(), mData, Nthreads, Preprocess(), range(), Scan Configuration::Rbounds(), ScanConfiguration::sigmabins2(), and ScanConfiguration::Tbounds().

Referenced by _FullScanToSimpleScan_(), AutoRoi_Scan_FullCallback(), and ManualRoi_Scan_FullCallback().

4.12.3.11 SetCentre()

Setter for the centre of the scan window.

Parameters

aPhysicalCentreX	The x-coordinate of the centre of the window in physical units (becomes 0 in algorithm units)
aPhysicalCentreY	The y-coordinate of the centre of the window in physical units (becomes 0 in algorithm units)

Definition at line 70 of file Rol.cpp.

References mPhysicalCentreX, and mPhysicalCentreY.

Referenced by LocalizationFile::ExtractRols().

4.12.3.12 SetWidth()

Setter for the size of the Rol window.

Parameters

aWidthX	The width of the window in physical units
aWidthY	The height of the window in physical units

Definition at line 77 of file Rol.cpp.

References mArea, mWidthX, and mWidthY.

Referenced by LocalizationFile::ExtractRols().

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

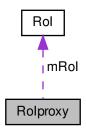
- include/BayesianClustering/Rol.hpp
- src/BayesianClustering/Rol.cpp

4.13 Rolproxy Class Reference

A lightweight wrapper for the Rol to store clusters for a given scan.

#include <RoIproxy.hpp>

Collaboration diagram for Rolproxy:



Public Member Functions

• Rolproxy (Rol &aRol)

Default constructor.

• Rolproxy (const Rolproxy &aOther)=delete

Deleted copy constructor.

Rolproxy & operator= (const Rolproxy &aOther)=delete

Deleted assignment operator.

Rolproxy (Rolproxy &&aOther)=default

Default move constructor.

Rolproxy & operator= (Rolproxy &&aOther)=default

Default move-assignment constructor.

∼Rolproxy ()

Default destructor.

void CheckClusterization (const double &R, const double &T)

Run validation tests on the clusters.

void ScanRT (const ScanConfiguration &aScanConfig, const std::function < void(Rolproxy &, const double &, const double &) > &aCallback, ProgressBar &aProgressBar, const uint8_t &aParallelization=1, const uint8_t &aOffset=0, const bool &aValidate=false)

Run an RT-scan.

void Clusterize (const double &R, const double &T, const std::function < void(Rolproxy &) > &aCallback)

Run clusterization for a specific choice of R and T.

void UpdateLogScore (const ScanConfiguration &aScanConfig)

Update log-probability after a scan.

void ValidateLogScore (const ScanConfiguration &aScanConfig)

Sean's validation code for testing when the running log-score fails.

• DataProxy & GetData (const std::size t &aIndex)

Get the proxy for the Nth neighbour of this data-point.

Public Attributes

• std::vector< DataProxy > mData

The collection of lightweight data-point wrappers used by this Rol wrapper.

• std::vector< Cluster > mClusters

The collection of clusters found by this scan.

• std::size_t mClusteredCount

The number of clustered data-points.

std::size_t mBackgroundCount

The number of background data-points.

• std::size_t mClusterCount

The number of non-Null clusters.

double mLogP

The log-probability density associated with the last scan.

const Rol & mRol

The underlying Rol this is a proxy to.

4.13.1 Detailed Description

A lightweight wrapper for the Rol to store clusters for a given scan.

Definition at line 19 of file Rolproxy.hpp.

4.13.2 Constructor & Destructor Documentation

4.13.2.1 Rolproxy() [1/3]

Default constructor.

Parameters

```
aRol An Rol for which this is a lightweight proxy
```

Definition at line 17 of file Rolproxy.cpp.

References mClusters, RoI::mData, and mData.

4.13.2.2 Rolproxy() [2/3]

Deleted copy constructor.

Parameters

aOther	Anonymous argument
--------	--------------------

4.13.2.3 Rolproxy() [3/3]

Default move constructor.

Parameters

aOther Anonymous argum	ent
------------------------	-----

4.13.3 Member Function Documentation

4.13.3.1 CheckClusterization()

Run validation tests on the clusters.

Parameters

R	The R of the last run scan
T	The T of the last run scan

Definition at line 34 of file Rolproxy.cpp.

References GetData(), mBackgroundCount, mClusterCount, mClusters, and mData.

4.13.3.2 Clusterize()

Run clusterization for a specific choice of R and T.

Parameters

	R	The R parameter for clusterization
	Т	The T parameter for clusterization
Ī	aCallback	A callback for the clusterization results

Definition at line 139 of file Rolproxy.cpp.

References Rol::getArea(), mClusters, Rol::mData, mData, and mRol.

Referenced by RoI::Clusterize().

4.13.3.3 GetData()

Get the proxy for the Nth neighbour of this data-point.

Returns

A reference to the neighbour data-proxy

Parameters

	alndex	The index of the neighbour we are looking for	
--	--------	---	--

Definition at line 74 of file Rolproxy.hpp.

References mData.

Referenced by CheckClusterization(), and DataProxy::Clusterize().

4.13.3.4 operator=() [1/2]

Deleted assignment operator.

Returns

Reference to this, for chaining calls

Parameters

aOther Anonymous argument

4.13.3.5 operator=() [2/2]

Default move-assignment constructor.

Returns

Reference to this, for chaining calls

Parameters

aOther	Anonymous argument
--------	--------------------

4.13.3.6 ScanRT()

Run an RT-scan.

Parameters

aScanConfig	The configuration parameters for the scan
aCallback	A callback for each RT-scan result
aProgressBar	The progress bar to update
aParallelization	The stride with which we will iterate across RT parameters
aOffset	The starting point for the strides as we iterate across RT parameters
aValidate	Run validation of the score calculation

Definition at line 104 of file Rolproxy.cpp.

4.13.3.7 UpdateLogScore()

Update log-probability after a scan.

Parameters

	aScanConfig	The configuration parameters for the scan
--	-------------	---

Definition at line 219 of file Rolproxy.cpp.

References ScanConfiguration::alpha(), ScanConfiguration::logAlpha(), ScanConfiguration::logGammaAlpha(), ScanConfiguration::logPb(), ScanConfiguration::logPbDagger(), mBackgroundCount, mClusterCount, mClusterCount, mClusters, mData, mLogP, and ScanConfiguration::sigmabins().

4.13.3.8 ValidateLogScore()

Sean's validation code for testing when the running log-score fails.

Parameters

aScanConfig	The configuration parameters for the scan

Definition at line 160 of file Rolproxy.cpp.

References mClusters, mData, Cluster::mParams, Data::s, ScanConfiguration::sigmabins2(), ScanConfiguration::sigmacount(), Data::x, and Data::y.

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

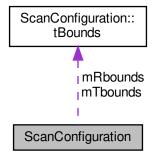
- include/BayesianClustering/Rolproxy.hpp
- src/BayesianClustering/Rolproxy.cpp

4.14 ScanConfiguration Class Reference

Class for storing the scan configuration parameters.

```
#include <Configuration.hpp>
```

Collaboration diagram for ScanConfiguration:



Classes

struct tBounds

A struct to store the bounds of a scan in either R or T.

Public Member Functions

ScanConfiguration (const std::string &aCfgFile)

Constructor which parses the parameters when passed in as commandline arguments.

ScanConfiguration (const std::size_t &aSigmacount, const double &aSigmaMin, const double &aSigmaMax, const std::function< double(const double &) > &aInterpolator, const std::size_t &aRbins, const double &a
 MinScanR, const double &aMaxScanR, const std::size_t &aTbins, const double &aMinScanT, const double &aPB, const double &aAlpha)

Constructor which take the parameters directly.

· ScanConfiguration (const ScanConfiguration &aOther)=delete

Deleted copy constructor.

• ScanConfiguration & operator= (const ScanConfiguration &aOther)=delete

Deleted assignment operator.

• ScanConfiguration (ScanConfiguration &&aOther)=default

Default move constructor.

ScanConfiguration & operator= (ScanConfiguration &&aOther)=default

Default move-assignment constructor.

∼ScanConfiguration ()=default

Default destructor.

void SetSigmaParameters (const std::size_t &aSigmacount, const double &aSigmaMin, const double &a
 SigmaMax, const std::function < double (const double &) > &aInterpolator)

Setter for the sigma-bins to be integrated over.

void SetRBins (const std::size_t &aRbins, const double &aMinScanR, const double &aMaxScanR)

Setter for the R bins for the RT scan.

- void SetTBins (const std::size_t &aTbins, const double &aMinScanT, const double &aMaxScanT)
- void SetPb (const double &aPB)

Setter for the P_b parameter.

void SetAlpha (const double &aAlpha)

Setter for the alpha parameter.

· const std::size_t & sigmacount () const

Getter for the sigma count.

const double & sigmaspacing () const

Getter for the sigma spacing.

• const std::vector< double > & sigmabins () const

Getter for the values of sigma.

const std::vector< double > & sigmabins2 () const

Getter for the values of sigma squared.

const std::vector< double > & probability_sigma () const

Getter for the probabilities of a given sigma.

• const std::vector< double > & log_probability_sigma () const

Getter for the log of the probabilities of a given sigma.

• const double & sigmabins (const std::size_t &i) const

Getter for the i'th value of sigma.

• const double & sigmabins2 (const std::size_t &i) const

Getter for the i'th value of sigma squared.

• const double & probability_sigma (const std::size_t &i) const

Getter for the probability of the i'th value of sigma.

const double & log_probability_sigma (const std::size_t &i) const

Getter for the log-probability of the i'th value of sigma.

· const tBounds & Rbounds () const

Getter for the bounds of R to scan.

· const tBounds & Tbounds () const

Getter for the bounds of T to scan.

const double & logPb () const

Logarithm of the P_b parameter.

• const double & logPbDagger () const

Logarithm of the (1 - P_b) parameter.

· const double & alpha () const

Getter for the alpha parameter.

const double & logAlpha () const

Getter for the logarithm of the alpha parameter.

const double & logGammaAlpha () const

Getter for the logarithm of the gamma function of alpha parameter.

Private Member Functions

void FromVector (const std::vector< std::string > &aArgs)

Parse the parameters when passed in as commandline arguments.

Private Attributes

std::size_t mSigmacount

The number of sigma bins.

· double mSigmaspacing

The spacing of sigma bins.

• std::vector< double > mSigmabins

The values of sigma.

• std::vector< double > mSigmabins2

The values of sigma squared.

• std::vector< double > mProbabilitySigma

The probability of a given sigma.

std::vector< double > mLogProbabilitySigma

The log-probability of a gievn sigma.

· tBounds mRbounds

The bounds of R to scan.

· tBounds mTbounds

The bounds of T to scan.

double mAlpha

The alpha parameter.

• double mLogAlpha

Logarithm of the alpha parameter.

• double mLogGammaAlpha

Logarithm of the gamma function of alpha parameter.

· double mLogPb

Logarithm of the P_b parameter.

· double mLogPbDagger

Logarithm of the(1- P_b) parameter.

4.14.1 Detailed Description

Class for storing the scan configuration parameters.

Definition at line 11 of file Configuration.hpp.

4.14.2 Constructor & Destructor Documentation

4.14.2.1 ScanConfiguration() [1/4]

Constructor which parses the parameters when passed in as commandline arguments.

Parameters

aCfgFile	A Scan-parameter config file name
----------	-----------------------------------

Definition at line 41 of file Configuration.cpp.

References FromVector().

4.14.2.2 ScanConfiguration() [2/4]

Constructor which take the parameters directly.

Parameters

aSigmacount	The number of sigma bins
aSigmaMin	The lowest sigma bin
aSigmaMax	The highest sigma bin
aInterpolator	Function-object to generate the probability of any given sigma
aRbins	The number of R bins to scan over
aMinScanR	The lowest value of R to scan
aMaxScanR	The largest value of R to scan
aTbins	The number of T bins to scan over
aMinScanT	The lowest value of T to scan
aMaxScanT	The largest value of T to scan
aPB	The P_b parameter
aAlpha	The alpha parameter

Definition at line 55 of file Configuration.cpp.

References SetAlpha(), SetPb(), SetRBins(), SetSigmaParameters(), and SetTBins().

4.14.2.3 ScanConfiguration() [3/4]

Deleted copy constructor.

Parameters

```
aOther Anonymous argument
```

4.14.2.4 ScanConfiguration() [4/4]

```
\begin{tabular}{lll} ScanConfiguration::ScanConfiguration ( & aOther ) & [default] \end{tabular}
```

Default move constructor.

Parameters

aOther	Anonymous argument	
--------	--------------------	--

4.14.3 Member Function Documentation

4.14.3.1 alpha()

```
const double& ScanConfiguration::alpha ( ) const [inline]
```

Getter for the alpha parameter.

Returns

The alpha parameter

Definition at line 211 of file Configuration.hpp.

References mAlpha.

Referenced by Rolproxy::UpdateLogScore().

4.14.3.2 FromVector()

```
void ScanConfiguration::FromVector ( {\tt const\ std::vector} < {\tt std::string} > {\tt \&\ aArgs}\ ) \quad [{\tt private}]
```

Parse the parameters when passed in as commandline arguments.

Parameters

aArgs	The commandline arguments
-------	---------------------------

Definition at line 162 of file Configuration.cpp.

 $References\ GSLInterpolator :: Eval(),\ SetAlpha(),\ SetPb(),\ SetRBins(),\ SetSigmaParameters(),\ SetTBins(),\ and\ Str \leftarrow ToDist().$

Referenced by ScanConfiguration().

4.14.3.3 log_probability_sigma() [1/2]

```
const std::vector< double >& ScanConfiguration::log_probability_sigma ( ) const [inline]
```

Getter for the log of the probabilities of a given sigma.

Returns

The log of the probabilities of given sigma

Definition at line 149 of file Configuration.hpp.

References mLogProbabilitySigma.

Referenced by Cluster::UpdateLogScore().

4.14.3.4 log_probability_sigma() [2/2]

```
\begin{tabular}{ll} \begin{tabular}{ll} const double \& ScanConfiguration::log\_probability\_sigma ( \\ const std::size\_t & i ) const [inline] \end{tabular}
```

Getter for the log-probability of the i'th value of sigma.

Parameters

i The index of the value of sigma to get the log-probability for

Returns

The log-probability of sigma_i

Definition at line 178 of file Configuration.hpp.

References mLogProbabilitySigma.

64 Class Documentation

4.14.3.5 logAlpha()

```
const double& ScanConfiguration::logAlpha ( ) const [inline]
```

Getter for the logarithm of the alpha parameter.

Returns

The logarithm of the alpha parameter

Definition at line 217 of file Configuration.hpp.

References mLogAlpha.

Referenced by Rolproxy::UpdateLogScore().

4.14.3.6 logGammaAlpha()

```
const double& ScanConfiguration::logGammaAlpha ( ) const [inline]
```

Getter for the logarithm of the gamma function of alpha parameter.

Returns

The logarithm of the gamma function of alpha parameter

Definition at line 223 of file Configuration.hpp.

References mLogGammaAlpha.

Referenced by Rolproxy::UpdateLogScore().

4.14.3.7 logPb()

```
const double& ScanConfiguration::logPb ( ) const [inline]
```

Logarithm of the P_b parameter.

Returns

Logarithm of the P_b parameter

Definition at line 198 of file Configuration.hpp.

References mLogPb.

Referenced by Rolproxy::UpdateLogScore().

4.14.3.8 logPbDagger()

```
const double& ScanConfiguration::logPbDagger ( ) const [inline]
```

Logarithm of the (1 - P_b) parameter.

Returns

Logarithm of the (1 - P_b) parameter

Definition at line 204 of file Configuration.hpp.

References mLogPbDagger.

Referenced by Rolproxy::UpdateLogScore().

4.14.3.9 operator=() [1/2]

Deleted assignment operator.

Returns

Reference to this, for chaining calls

Parameters

```
aOther Anonymous argument
```

4.14.3.10 operator=() [2/2]

Default move-assignment constructor.

Returns

Reference to this, for chaining calls

Parameters

aOther	Anonymous argument

66 Class Documentation

4.14.3.11 probability_sigma() [1/2]

```
const std::vector< double >& ScanConfiguration::probability_sigma ( ) const [inline]
```

Getter for the probabilities of a given sigma.

Returns

The probabilities of given sigma

Definition at line 143 of file Configuration.hpp.

References mProbabilitySigma.

4.14.3.12 probability_sigma() [2/2]

```
const double \ ScanConfiguration::probability_sigma ( const std::size_t & i ) const [inline]
```

Getter for the probability of the i'th value of sigma.

Parameters

i The index of the value of sigma to get the probability for

Returns

The probability of sigma_i

Definition at line 171 of file Configuration.hpp.

References mProbabilitySigma.

4.14.3.13 Rbounds()

```
const tBounds& ScanConfiguration::Rbounds ( ) const [inline]
```

Getter for the bounds of R to scan.

Returns

The Ibounds of R to scan

Definition at line 185 of file Configuration.hpp.

References mRbounds.

Referenced by Rol::ScanRT().

4.14.3.14 SetAlpha()

Setter for the alpha parameter.

Parameters

aAlpha
aAlpha

Definition at line 124 of file Configuration.cpp.

References mAlpha, mLogAlpha, and mLogGammaAlpha.

Referenced by FromVector(), and ScanConfiguration().

4.14.3.15 SetPb()

Setter for the P_b parameter.

Parameters

aPB	The P_b parameter

Definition at line 117 of file Configuration.cpp.

References mLogPb, and mLogPbDagger.

Referenced by FromVector(), and ScanConfiguration().

4.14.3.16 SetRBins()

Setter for the R bins for the RT scan.

Parameters

aRbins	The number of R bins to scan over
aMinScanR	The lowest value of R to scan
aMaxScanR	The largest value of R to scan

68 Class Documentation

Definition at line 97 of file Configuration.cpp.

References ScanConfiguration::tBounds::bins, ScanConfiguration::tBounds::max, ScanConfiguration::tBounds::min, mRbounds, and ScanConfiguration::tBounds::spacing.

Referenced by FromVector(), and ScanConfiguration().

4.14.3.17 SetSigmaParameters()

Setter for the sigma-bins to be integrated over.

Parameters

aSigmacount	The number of sigma bins
aSigmaMin	The lowest sigma bin
aSigmaMax	The highest sigma bin
aInterpolator	Function-object to generate the probability of any given sigma

Definition at line 69 of file Configuration.cpp.

References mLogProbabilitySigma, mProbabilitySigma, mSigmabins, mSigmabins2, mSigmacount, $m \leftarrow Sigmaspacing$, and range().

Referenced by FromVector(), and ScanConfiguration().

4.14.3.18 SetTBins()

Parameters

aTbii	าร	The number of T bins to scan over
aMin	ScanT	The lowest value of T to scan
aMax	«ScanT	The largest value of T to scan

Definition at line 107 of file Configuration.cpp.

References ScanConfiguration::tBounds::bins, ScanConfiguration::tBounds::max, ScanConfiguration::tBounds::min, mTbounds, and ScanConfiguration::tBounds::spacing.

Referenced by FromVector(), and ScanConfiguration().

4.14.3.19 sigmabins() [1/2]

```
const std::vector< double >& ScanConfiguration::sigmabins ( ) const [inline]
```

Getter for the values of sigma.

Returns

The values of sigma

Definition at line 131 of file Configuration.hpp.

References mSigmabins.

Referenced by Cluster::UpdateLogScore(), and Rolproxy::UpdateLogScore().

4.14.3.20 sigmabins() [2/2]

Getter for the i'th value of sigma.

Parameters

i The index of the value of sigma to get

Returns

The value of sigma_i

Definition at line 157 of file Configuration.hpp.

References mSigmabins.

4.14.3.21 sigmabins2() [1/2]

```
\verb|const| std::vector<| double > & ScanConfiguration::sigmabins2 () const [inline]| \\
```

Getter for the values of sigma squared.

70 Class Documentation

Returns

The values of sigma squared

Definition at line 137 of file Configuration.hpp.

References mSigmabins2.

Referenced by RoI::ScanRT(), and RoIproxy::ValidateLogScore().

4.14.3.22 sigmabins2() [2/2]

Getter for the i'th value of sigma squared.

Parameters

i The index of the value of sigma squared to get

Returns

The value of sigma_i squared

Definition at line 164 of file Configuration.hpp.

References mSigmabins2.

4.14.3.23 sigmacount()

```
const std::size_t& ScanConfiguration::sigmacount ( ) const [inline]
```

Getter for the sigma count.

Returns

The sigma count

Definition at line 117 of file Configuration.hpp.

References mSigmacount.

Referenced by Rolproxy::ValidateLogScore().

4.14.3.24 sigmaspacing()

```
const double& ScanConfiguration::sigmaspacing ( ) const [inline]
```

Getter for the sigma spacing.

Returns

The sigma spacing

Definition at line 124 of file Configuration.hpp.

References mSigmaspacing.

4.14.3.25 Tbounds()

```
const tBounds& ScanConfiguration::Tbounds ( ) const [inline]
```

Getter for the bounds of T to scan.

Returns

The Ibounds of T to scan

Definition at line 191 of file Configuration.hpp.

References mTbounds.

Referenced by RoI::ScanRT().

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

- include/BayesianClustering/Configuration.hpp
- src/BayesianClustering/Configuration.cpp

4.15 ScanEntry Struct Reference

A struct for storing a result of an individual scan configuration.

```
#include <API.hpp>
```

Public Member Functions

• bool operator< (const ScanEntry &aOther)

Comparison operator for sorting.

bool operator== (const ScanEntry &aOther)

Equality operator required by boost python.

72 Class Documentation

Public Attributes

• double r

The R parameter

double t

The T parameter.

• double score

The score.

4.15.1 Detailed Description

A struct for storing a result of an individual scan configuration.

Definition at line 16 of file API.hpp.

4.15.2 Member Function Documentation

4.15.2.1 operator<()

Comparison operator for sorting.

Returns

Whether we are smaller than the other

Parameters

```
aOther Another ScanEntry to compare against
```

Definition at line 24 of file API.hpp.

References r, and t.

4.15.2.2 operator==()

Equality operator required by boost python.

Returns

Whether we are equal to the other

Parameters

aOther	Another ScanEntry to compare against
--------	--------------------------------------

Definition at line 33 of file API.hpp.

References r, score, and t.

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

• include/BayesianClustering/API.hpp

4.16 ScanConfiguration::tBounds Struct Reference

A struct to store the bounds of a scan in either R or T.

#include <Configuration.hpp>

Public Attributes

• double min

The lowest value of R to scan.

· double max

The largest value of R to scan.

double spacing

The spacing of value of R to scan.

std::size_t bins

The number of R values to scan.

4.16.1 Detailed Description

A struct to store the bounds of a scan in either R or T.

Definition at line 16 of file Configuration.hpp.

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

• include/BayesianClustering/Configuration.hpp

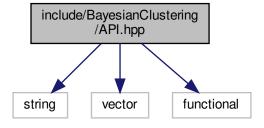
74 **Class Documentation**

Chapter 5

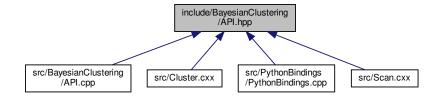
File Documentation

5.1 include/BayesianClustering/API.hpp File Reference

```
#include <string>
#include <vector>
#include <functional>
Include dependency graph for API.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

struct ScanEntry

A struct for storing a result of an individual scan configuration.

struct ClusterWrapper

A struct for storing a result of an individual scan configuration.

Functions

 void AutoRoi_Scan_FullCallback (const std::string &aInFile, const ScanConfiguration &aScanConfig, const std::function< void(Rolproxy &, const double &, const double &) > &aCallback)

Automatically extract Rol, run scan and apply a full call-back.

void AutoRoi_Scan_SimpleCallback (const std::string &aInFile, const ScanConfiguration &aScanConfig, const std::function< void(const std::vector< ScanEntry > &) > &aCallback)

Automatically extract Rol, run scan and apply a simple call-back.

Automatically extract Rol, run scan and dump to JSON file.

 void AutoRoi_Cluster_FullCallback (const std::string &aInFile, const double &aR, const double &aT, const std::function< void(Rolproxy &) > &aCallback)

Automatically extract Rol, clusterize and apply a full call-back.

void AutoRoi_Cluster_SimpleCallback (const std::string &aInFile, const double &aR, const double &aT, const std::function< void(const std::vector< ClusterWrapper > &) > &aCallback)

Automatically extract Rol, clusterize and apply a full call-back.

void ManualRoi_Scan_FullCallback (const std::string &aInFile, const ManualRol &aManualRol, const ScanConfiguration &aScanConfig, const std::function< void(Rolproxy &, const double &, const double &) > &aCallback)

Manually extract Rol, run scan and apply a full call-back.

 void ManualRoi_Scan_SimpleCallback (const std::string &aInFile, const ManualRol &aManualRol, const ScanConfiguration &aScanConfig, const std::function< void(const std::vector< ScanEntry > &) > &a← Callback)

Manually extract Rol, run scan and apply a simple call-back.

• void ManualRoi_Scan_ToJson (const std::string &aInFile, const ManualRol &aManualRol, const ScanConfiguration &aScanConfig, const std::string &aOutFile)

Manually extract Rol, run scan and dump to JSON file.

void ManualRoi_Cluster_FullCallback (const std::string &aInFile, const ManualRol &aManualRol, const double &aR, const double &aT, const std::function
 void(Rolproxy &) > &aCallback)

Manually extract Rol, clusterize and apply a full call-back.

void ManualRoi_Cluster_SimpleCallback (const std::string &aInFile, const ManualRoI &aManualRoI, const double &aR, const double &aT, const std::function< void(const std::vector< ClusterWrapper > &) > &a← Callback)

Manually extract Rol, clusterize and apply a full call-back.

5.1.1 Function Documentation

5.1.1.1 AutoRoi_Cluster_FullCallback()

Automatically extract Rol, clusterize and apply a full call-back.

Parameters

alnFile	The name of the localization file
aR	The R value of the clusterizer
аТ	The T value of the clusterizer
aCallback	The callback to be applied

Definition at line 104 of file API.cpp.

References Rol::Clusterize(), and LocalizationFile::ExtractRols().

Referenced by AutoRoi_Cluster_SimpleCallback().

5.1.1.2 AutoRoi_Cluster_SimpleCallback()

Automatically extract Rol, clusterize and apply a full call-back.

Parameters

alnFile	The name of the localization file
aR	The R value of the clusterizer
аТ	The T value of the clusterizer
aCallback	The callback to be applied

Definition at line 110 of file API.cpp.

References _FullClusterToSimpleCluster_(), and AutoRoi_Cluster_FullCallback().

5.1.1.3 AutoRoi_Scan_FullCallback()

Automatically extract Rol, run scan and apply a full call-back.

Parameters

alnFile	The name of the localization file
aScanConfig	The configuration for the scan
aCallback	The full callback to be applied

Definition at line 84 of file API.cpp.

References LocalizationFile::ExtractRols(), and Rol::ScanRT().

5.1.1.4 AutoRoi_Scan_SimpleCallback()

Automatically extract Rol, run scan and apply a simple call-back.

Parameters

alnFile	The name of the localization file
aScanConfig	The configuration for the scan
aCallback	The simple callback to be applied

Definition at line 90 of file API.cpp.

References _FullScanToSimpleScan_(), and LocalizationFile::ExtractRols().

Referenced by AutoRoi_Scan_ToJson().

5.1.1.5 AutoRoi_Scan_ToJson()

Automatically extract Rol, run scan and dump to JSON file.

Parameters

alnFile	The name of the localization file
aScanConfig	The configuration for the scan
aOutFile	The name of the output JSON file

Definition at line 96 of file API.cpp.

 $References _ScanCallback_Json_(), \ and \ AutoRoi_Scan_SimpleCallback().$

Referenced by BOOST_PYTHON_MODULE(), and main().

5.1.1.6 ManualRoi_Cluster_FullCallback()

Manually extract Rol, clusterize and apply a full call-back.

Parameters

alnFile	The name of the localization file
aManualRol	The manually-specified Rol window
aR	The R value of the clusterizer
аТ	The T value of the clusterizer
aCallback	The callback to be applied

Definition at line 138 of file API.cpp.

References Rol::Clusterize(), and LocalizationFile::ExtractRols().

Referenced by ManualRoi_Cluster_SimpleCallback().

5.1.1.7 ManualRoi_Cluster_SimpleCallback()

Manually extract Rol, clusterize and apply a full call-back.

Parameters

alnFile	The name of the localization file	
aManualRol	The manually-specified Rol window	
aR	The R value of the clusterizer	
аТ	The T value of the clusterizer	
aCallback	The callback to be applied	

Definition at line 144 of file API.cpp.

 $References _Full Cluster To Simple Cluster _(), and Manual Roi_Cluster _Full Callback().$

5.1.1.8 ManualRoi_Scan_FullCallback()

Manually extract Rol, run scan and apply a full call-back.

Parameters

alnFile The name of the localization file	
aManualRol	The manually-specified Rol window
aScanConfig	The configuration for the scan
aCallback	The full callback to be applied

Definition at line 118 of file API.cpp.

References LocalizationFile::ExtractRols(), and Rol::ScanRT().

5.1.1.9 ManualRoi_Scan_SimpleCallback()

Manually extract Rol, run scan and apply a simple call-back.

Parameters

alnFile	The name of the localization file
aManualRol	The manually-specified Rol window
aScanConfig	The configuration for the scan
aCallback	The simple callback to be applied

Definition at line 124 of file API.cpp.

References _FullScanToSimpleScan_(), and LocalizationFile::ExtractRols().

Referenced by ManualRoi_Scan_ToJson().

5.1.1.10 ManualRoi_Scan_ToJson()

Manually extract Rol, run scan and dump to JSON file.

Parameters

alnFile The name of the localization file	
aManualRol	The manually-specified Rol window
aScanConfig	The configuration for the scan
aOutFile	The name of the output JSON file

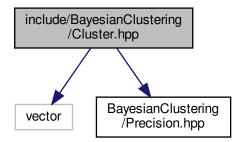
Definition at line 130 of file API.cpp.

References _ScanCallback_Json_(), and ManualRoi_Scan_SimpleCallback().

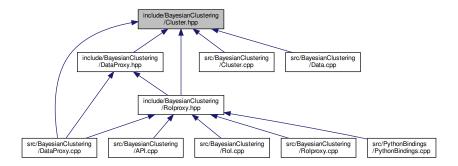
Referenced by BOOST_PYTHON_MODULE().

5.2 include/BayesianClustering/Cluster.hpp File Reference

```
#include <vector>
#include "BayesianClustering/Precision.hpp"
Include dependency graph for Cluster.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

· class Cluster

A class representing a cluster.

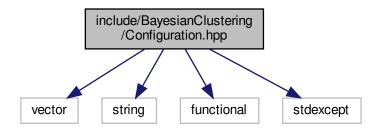
• struct Cluster::Parameter

A struct representing the cluster parameters.

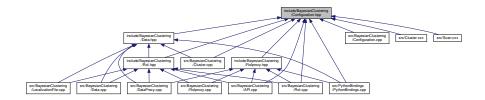
5.3 include/BayesianClustering/Configuration.hpp File Reference

```
#include <vector>
#include <string>
#include <functional>
#include <stdexcept>
```

Include dependency graph for Configuration.hpp:



This graph shows which files directly or indirectly include this file:



Classes

· class ScanConfiguration

Class for storing the scan configuration parameters.

· struct ScanConfiguration::tBounds

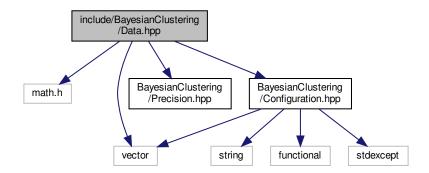
A struct to store the bounds of a scan in either R or T.

class AuxConfiguration

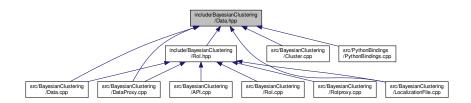
Class for storing the auxilliary configuration parameters.

5.4 include/BayesianClustering/Data.hpp File Reference

```
#include <math.h>
#include <vector>
#include "BayesianClustering/Precision.hpp"
#include "BayesianClustering/Configuration.hpp"
Include dependency graph for Data.hpp:
```



This graph shows which files directly or indirectly include this file:



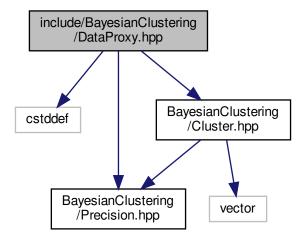
Classes

· class Data

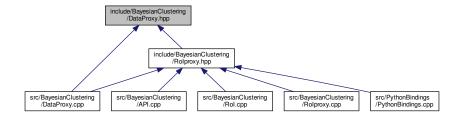
A class to store the raw data-points.

5.5 include/BayesianClustering/DataProxy.hpp File Reference

```
#include <cstddef>
#include "BayesianClustering/Precision.hpp"
#include "BayesianClustering/Cluster.hpp"
Include dependency graph for DataProxy.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

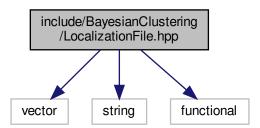
class DataProxy

A light-weight proxy for the raw data-points.

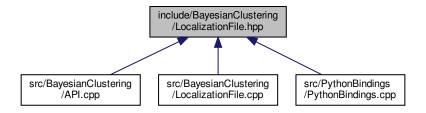
5.6 include/BayesianClustering/LocalizationFile.hpp File Reference

#include <vector>
#include <string>
#include <functional>

Include dependency graph for LocalizationFile.hpp:



This graph shows which files directly or indirectly include this file:



Classes

struct ManualRol

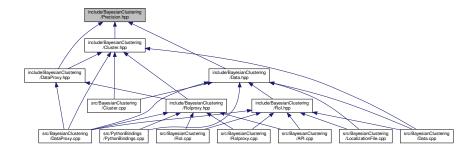
A struct for storing the manual Rols.

· class LocalizationFile

A class to store the raw data-points.

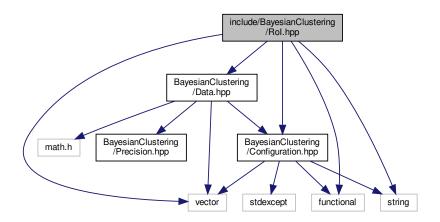
5.7 include/BayesianClustering/Precision.hpp File Reference

This graph shows which files directly or indirectly include this file:

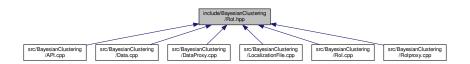


5.8 include/BayesianClustering/Rol.hpp File Reference

```
#include <vector>
#include <functional>
#include <string>
#include "BayesianClustering/Data.hpp"
#include "BayesianClustering/Configuration.hpp"
Include dependency graph for Rol.hpp:
```



This graph shows which files directly or indirectly include this file:



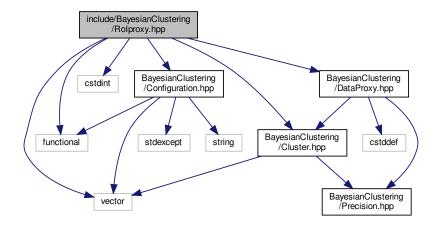
Classes

· class Rol

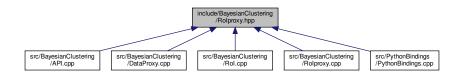
A class which holds the raw Rol data and global parameters.

5.9 include/BayesianClustering/Rolproxy.hpp File Reference

```
#include <vector>
#include <functional>
#include <cstdint>
#include "BayesianClustering/Cluster.hpp"
#include "BayesianClustering/DataProxy.hpp"
#include "BayesianClustering/Configuration.hpp"
Include dependency graph for Rolproxy.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

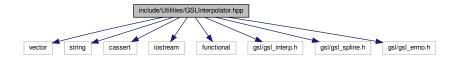
class Rolproxy

A lightweight wrapper for the Rol to store clusters for a given scan.

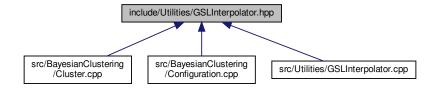
5.10 include/Utilities/GSLInterpolator.hpp File Reference

```
#include <vector>
#include <string>
#include <cassert>
#include <iostream>
#include <functional>
#include "gsl/gsl_interp.h"
#include "gsl/gsl_spline.h"
#include "gsl/gsl_errno.h"
```

Include dependency graph for GSLInterpolator.hpp:



This graph shows which files directly or indirectly include this file:



Classes

· class GSLInterpolator

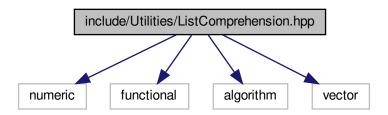
A utility wrapper around the GSL interpolator to give it a clean C++ interface.

5.11 include/Utilities/ListComprehension.hpp File Reference

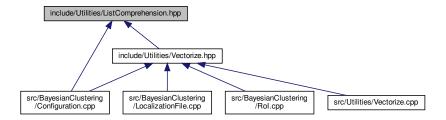
```
#include <numeric>
#include <functional>
#include <algorithm>
```

#include <vector>

Include dependency graph for ListComprehension.hpp:



This graph shows which files directly or indirectly include this file:



Functions

template<typename tContainer , typename tExpr , typename T = typename std::remove_reference<tContainer>::type::value_type, typename U = decltype(std::declval<tExpr>().operator()(std::declval<T>()))>
 std::enable_if< not std::is_same< U, void >::value, std::vector< U > >::type operator| (tExpr &&aExpr, tContainer &&aContainer)

Super nerd template magic emulating list comprehension for function with return type.

template<typename tContainer , typename tExpr , typename T = typename std::remove_reference<tContainer>::type::value_type, typename U = decltype(std::declval<tExpr>().operator()(std::declval<T>()))>
 std::enable_if< std::is_same< U, void >::value, void >::type operator| (tExpr &&aExpr, tContainer &&a Container)

Super nerd template magic emulating list comprehension for function with void return type.

• template<typename tContainer , typename tType , typename tContainerType = typename std::remove_reference<tContainer>::type \leftrightarrow ::value type>

std::vector< tType > operator (tType tContainerType::*aPtr, tContainer &&aContainer)

Return a container holding copies of a member-variable from each object in a container.

std::vector< std::size_t > range (const std::size_t &N)

Emulate the python range function to generate a vector of ints.

5.11.1 Function Documentation

5.11.1.1 operator" | () [1/3]

```
template<typename tContainer , typename tExpr , typename T = typename std::remove_reference<t←
Container>::type::value_type, typename U = decltype( std::declval<tExpr>().operator()( std←
::declval<T>() ) )>
std::enable_if< not std::is_same<U, void>::value, std::vector< U > >::type operator (
tExpr && aExpr,
tContainer && aContainer ) [inline]
```

Super nerd template magic emulating list comprehension for function with return type.

Template Parameters

tContainer	A container type
tExpr	A function-call type
T	Template magic to determine the type of the data in the container
U	Template magic to determine the return-type of the function, given the type of the data in the container

Parameters

aExpr	A function-call to be applied to each element of the container
aContainer	A container holding the arguments to be fed to the expression

Returns

A vector of the results of the vectorized operations

Definition at line 20 of file ListComprehension.hpp.

5.11.1.2 operator" | () [2/3]

Super nerd template magic emulating list comprehension for function with void return type.

Template Parameters

tContainer	A container type
tExpr	A function-call type
T	Template magic to determine the type of the data in the container
U	Template magic to determine the return-type of the function, given the type of the data in the container

Parameters

aExpr	A function-call to be applied to each element of the container
aContainer	A container holding the arguments to be fed to the expression

Returns

Specialization of the vectorization for functions returning void

Definition at line 39 of file ListComprehension.hpp.

5.11.1.3 operator" | () [3/3]

Return a container holding copies of a member-variable from each object in a container.

Template Parameters

tType	A container type
tContainerType	Template magic to determine the type of the data in the container

Parameters

ć	aPtr	A pointer-to-member-variable to be applied to each element of the container
ć	aContainer	A container holding the objects whose member variable is to be extracted

Returns

A vector of the results of the vectorized operations

Definition at line 51 of file ListComprehension.hpp.

5.11.1.4 range()

Emulate the python range function to generate a vector of ints.

Parameters

N The number of elements

Returns

A vector of ints

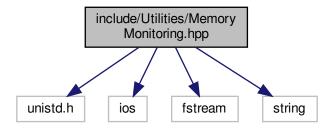
Definition at line 70 of file ListComprehension.hpp.

Referenced by LocalizationFile::LocalizationFile(), Rol::Preprocess(), Rol::ScanRT(), and ScanConfiguration::Set ← SigmaParameters().

5.12 include/Utilities/MemoryMonitoring.hpp File Reference

```
#include <unistd.h>
#include <ios>
#include <fstream>
#include <string>
```

Include dependency graph for MemoryMonitoring.hpp:



Functions

• void mem_usage (double &vm_usage, double &resident_set)

Utility to get Virtual Memory and Resident Set usage.

5.12.1 Function Documentation

5.12.1.1 mem_usage()

Utility to get Virtual Memory and Resident Set usage.

Parameters

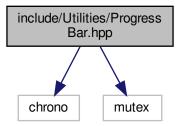
vm_usage	Return the Virtual Memory usage
resident_set	Return the Resident Set usage

Definition at line 12 of file MemoryMonitoring.hpp.

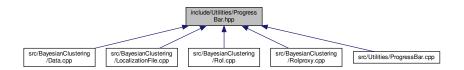
5.13 include/Utilities/ProgressBar.hpp File Reference

#include <chrono>
#include <mutex>

Include dependency graph for ProgressBar.hpp:



This graph shows which files directly or indirectly include this file:



Classes

· class ProgressBar

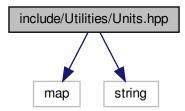
A utility progress-bar.

struct ProgressTimer

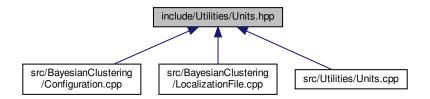
A utility code timer.

5.14 include/Utilities/Units.hpp File Reference

#include <map>
#include <string>
Include dependency graph for Units.hpp:



This graph shows which files directly or indirectly include this file:



Functions

long double operator""_nanometer (long double aVal)

User-defined literals fot nanometer quantities.

• long double operator""_nanometer (unsigned long long aVal)

User-defined literals fot nanometer quantities.

• long double operator""_micrometer (long double aVal)

User-defined literals fot micrometer quantities.

• long double operator""_micrometer (unsigned long long aVal)

User-defined literals fot micrometer quantities.

long double StrToDist (const std::string &aStr)

Convert a string representation to a distance.

Variables

• double nanometer = 1e-9

Define a constant for converting nanometers to meters.

• double micrometer = 1e-6

Define a constant for converting micrometers to meters.

• double millimeter = 1e-3

Define a constant for converting millimeters to meters.

• double meter = 1e-0

Define a constant for converting meters to meters.

const std::map< std::string, double > UnitMap

A map for converting string representations of SI units to scaling factors.

5.14.1 Function Documentation

5.14.1.1 operator"""_micrometer() [1/2]

```
long double operator ""_micrometer ( long \ double \ aVal \ )
```

User-defined literals fot micrometer quantities.

Parameters

```
aVal The specified value
```

Returns

The literal value

Definition at line 39 of file Units.hpp.

References micrometer.

5.14.1.2 operator"""_micrometer() [2/2]

User-defined literals fot micrometer quantities.

Parameters

aVal	The specified value

Returns

The literal value

Definition at line 47 of file Units.hpp.

References micrometer.

5.14.1.3 operator"""_nanometer() [1/2]

```
long double operator""_nanometer ( long double aVal )
```

User-defined literals fot nanometer quantities.

Parameters

```
aVal The specified value
```

Returns

The literal value

Definition at line 23 of file Units.hpp.

References nanometer.

5.14.1.4 operator"""_nanometer() [2/2]

```
long double operator""_nanometer (
          unsigned long long aVal )
```

User-defined literals fot nanometer quantities.

Parameters

```
aVal The specified value
```

Returns

The literal value

Definition at line 31 of file Units.hpp.

References nanometer.

5.14.1.5 StrToDist()

```
long double StrToDist (
            const std::string & aStr)
```

Convert a string representation to a distance.

Parameters

aS	tr	A string representation of a distance
----	----	---------------------------------------

Returns

The literal value

Definition at line 12 of file Units.cpp.

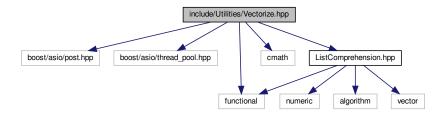
References UnitMap.

Referenced by ScanConfiguration::FromVector(), and AuxConfiguration::FromVector().

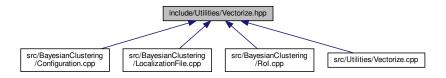
include/Utilities/Vectorize.hpp File Reference

```
#include <boost/asio/post.hpp>
#include <boost/asio/thread_pool.hpp>
#include <functional>
#include <cmath>
#include "ListComprehension.hpp"
```

Include dependency graph for Vectorize.hpp:



This graph shows which files directly or indirectly include this file:



Functions

• template<typename tContainer , typename tExpr , typename tContainerType = typename std::remove_reference<tContainer>::type ← ::value_type>

```
void operator|| (tExpr &&aExpr, tContainer &&aContainer)
```

Syntactic sugar to allow you to interleave parallelize via operator.

• template<typename tContainer , typename tExpr , typename tContainerType = typename std::remove_reference<tContainer>::type ← ::value_type>

```
void operator&& (tExpr &&aExpr, tContainer &&aContainer)
```

Syntactic sugar to allow you to block parallelize via operator.

Variables

std::size_t Nthreads

Utility variable for the concurrency.

5.15.1 Function Documentation

5.15.1.1 operator&&()

Syntactic sugar to allow you to block parallelize via operator.

Template Parameters

tContainer	A container type
tExpr	A function-call type
tContainerType	A SFINAE hack to ensure that the container is a container

Parameters

aExpr	A function-call to be applied to each element of the container
aContainer	A container holding the arguments to be distributed to the parallelized function calls

Definition at line 38 of file Vectorize.hpp.

References Nthreads.

5.15.1.2 operator" | " | ()

Syntactic sugar to allow you to interleave parallelize via operator.

Template Parameters

tContainer	A container type
tExpr	A function-call type
tContainerType	A SFINAE hack to ensure that the container is a container

Parameters

aExpr	A function-call to be applied to each element of the container
aContainer	A container holding the arguments to be distributed to the parallelized function calls

Definition at line 22 of file Vectorize.hpp.

References Nthreads.

5.15.2 Variable Documentation

5.15.2.1 Nthreads

```
std::size_t Nthreads [extern]
```

Utility variable for the concurrency.

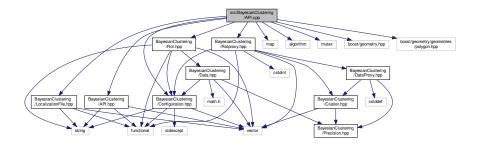
Utility variable for the concurrency.

Definition at line 8 of file Vectorize.cpp.

Referenced by AuxConfiguration::FromVector(), LocalizationFile::LocalizationFile(), operator & (), operator | |()|, and Rol::ScanRT().

5.16 src/BayesianClustering/API.cpp File Reference

```
#include "BayesianClustering/API.hpp"
#include "BayesianClustering/LocalizationFile.hpp"
#include "BayesianClustering/Configuration.hpp"
#include "BayesianClustering/RoI.hpp"
#include "BayesianClustering/RoIproxy.hpp"
#include <map>
#include <algorithm>
#include <mutex>
#include <boost/geometry.hpp>
#include <boost/geometry/geometries/polygon.hpp>
Include dependency graph for API.opp:
```



Functions

- void <u>ScanCallback</u> Json (const std::vector< <u>ScanEntry</u> > &aVector, const std::string &aOutFile)

 A callback to dump a scan to a JSON file.
- void _FullScanToSimpleScan_ (Rol &aRol, const ScanConfiguration &aScanConfig, const std::function
 void(const std::vector< ScanEntry > &) > &aCallback)

A callback to neatly package the scan results for easy consumption.

void _FullClusterToSimpleCluster_ (Rolproxy &aRolproxy, const std::function< void(const std::vector<
 ClusterWrapper > &) > &aCallback)

A callback to neatly package the scan results for easy consumption.

 void AutoRoi_Scan_FullCallback (const std::string &aInFile, const ScanConfiguration &aScanConfig, const std::function< void(Rolproxy &, const double &, const double &) > &aCallback)

Automatically extract Rol, run scan and apply a full call-back.

void AutoRoi_Scan_SimpleCallback (const std::string &aInFile, const ScanConfiguration &aScanConfig, const std::function
 void (const std::vector< ScanEntry > &) > &aCallback)

Automatically extract Rol, run scan and apply a simple call-back.

void AutoRoi_Scan_ToJson (const std::string &aInFile, const ScanConfiguration &aScanConfig, const std
 ::string &aOutFile)

Automatically extract Rol, run scan and dump to JSON file.

 void AutoRoi_Cluster_FullCallback (const std::string &aInFile, const double &aR, const double &aT, const std::function< void(Rolproxy &) > &aCallback)

Automatically extract Rol, clusterize and apply a full call-back.

void AutoRoi_Cluster_SimpleCallback (const std::string &aInFile, const double &aR, const double &aT, const std::function
 void(const std::vector< ClusterWrapper > &) > &aCallback)

Automatically extract Rol, clusterize and apply a full call-back.

void ManualRoi_Scan_FullCallback (const std::string &aInFile, const ManualRol &aManualRol, const ScanConfiguration &aScanConfig, const std::function< void(Rolproxy &, const double &, const double &) > &aCallback)

Manually extract Rol, run scan and apply a full call-back.

 void ManualRoi_Scan_SimpleCallback (const std::string &aInFile, const ManualRol &aManualRol, const ScanConfiguration &aScanConfig, const std::function< void(const std::vector< ScanEntry > &) > &a← Callback)

Manually extract Rol, run scan and apply a simple call-back.

 void ManualRoi_Scan_ToJson (const std::string &aInFile, const ManualRol &aManualRol, const ScanConfiguration &aScanConfig, const std::string &aOutFile)

Manually extract Rol, run scan and dump to JSON file.

void ManualRoi_Cluster_FullCallback (const std::string &aInFile, const ManualRol &aManualRol, const double &aR, const double &aT, const std::function
 void(Rolproxy &) > &aCallback)

Manually extract Rol, clusterize and apply a full call-back.

void ManualRoi_Cluster_SimpleCallback (const std::string &aInFile, const ManualRoI &aManualRoI, const double &aR, const double &aT, const std::function< void(const std::vector< ClusterWrapper > &) > &a← Callback)

Manually extract Rol, clusterize and apply a full call-back.

5.16.1 Function Documentation

5.16.1.1 _FullClusterToSimpleCluster_()

A callback to neatly package the scan results for easy consumption.

Parameters

aRolproxy	The region-proxy containing the clusters
aCallback	The simple callback to be applied

Definition at line 54 of file API.cpp.

References Rolproxy::mData.

Referenced by AutoRoi_Cluster_SimpleCallback(), and ManualRoi_Cluster_SimpleCallback().

5.16.1.2 FullScanToSimpleScan ()

A callback to neatly package the scan results for easy consumption.

Parameters

aRol	The region of interest
aScanConfig	The configuration for the scan
aCallback	The simple callback to be applied

Definition at line 41 of file API.cpp.

References Rolproxy::mLogP, and Rol::ScanRT().

Referenced by AutoRoi_Scan_SimpleCallback(), and ManualRoi_Scan_SimpleCallback().

5.16.1.3 _ScanCallback_Json_()

A callback to dump a scan to a JSON file.

Parameters

aVector	A vector of scan results
aOutFile	The name of the output JSON file

Definition at line 23 of file API.cpp.

 $Referenced\ by\ AutoRoi_Scan_ToJson(),\ and\ ManualRoi_Scan_ToJson().$

5.16.1.4 AutoRoi_Cluster_FullCallback()

Automatically extract Rol, clusterize and apply a full call-back.

Parameters

alnFile	The name of the localization file
aR	The R value of the clusterizer
аТ	The T value of the clusterizer
aCallback	The callback to be applied

Definition at line 104 of file API.cpp.

References Rol::Clusterize(), and LocalizationFile::ExtractRols().

Referenced by AutoRoi_Cluster_SimpleCallback().

5.16.1.5 AutoRoi_Cluster_SimpleCallback()

Automatically extract Rol, clusterize and apply a full call-back.

Parameters

aInFile	The name of the localization file
aR	The R value of the clusterizer
аТ	The T value of the clusterizer
aCallback	The callback to be applied

Definition at line 110 of file API.cpp.

References _FullClusterToSimpleCluster_(), and AutoRoi_Cluster_FullCallback().

5.16.1.6 AutoRoi_Scan_FullCallback()

Automatically extract Rol, run scan and apply a full call-back.

Parameters

alnFile	The name of the localization file
aScanConfig	The configuration for the scan
aCallback	The full callback to be applied

Definition at line 84 of file API.cpp.

References LocalizationFile::ExtractRols(), and Rol::ScanRT().

5.16.1.7 AutoRoi_Scan_SimpleCallback()

Automatically extract Rol, run scan and apply a simple call-back.

Parameters

alnFile	The name of the localization file
aScanConfig	The configuration for the scan
aCallback	The simple callback to be applied

Definition at line 90 of file API.cpp.

References _FullScanToSimpleScan_(), and LocalizationFile::ExtractRols().

Referenced by AutoRoi_Scan_ToJson().

5.16.1.8 AutoRoi_Scan_ToJson()

Automatically extract Rol, run scan and dump to JSON file.

Parameters

alnFile	The name of the localization file
aScanConfig	The configuration for the scan
aOutFile	The name of the output JSON file

Definition at line 96 of file API.cpp.

References _ScanCallback_Json_(), and AutoRoi_Scan_SimpleCallback().

Referenced by BOOST_PYTHON_MODULE(), and main().

5.16.1.9 ManualRoi_Cluster_FullCallback()

```
const ManualRoI & aManualRoI,
const double & aR,
const double & aT,
const std::function< void(RoIproxy &) > & aCallback )
```

Manually extract Rol, clusterize and apply a full call-back.

Parameters

alnFile	The name of the localization file
aManualRol	The manually-specified Rol window
aR	The R value of the clusterizer
аТ	The T value of the clusterizer
aCallback	The callback to be applied

Definition at line 138 of file API.cpp.

References Rol::Clusterize(), and LocalizationFile::ExtractRols().

Referenced by ManualRoi_Cluster_SimpleCallback().

5.16.1.10 ManualRoi_Cluster_SimpleCallback()

Manually extract Rol, clusterize and apply a full call-back.

Parameters

alnFile	The name of the localization file
aManualRol	The manually-specified Rol window
aR	The R value of the clusterizer
аТ	The T value of the clusterizer
aCallback	The callback to be applied

Definition at line 144 of file API.cpp.

 $References _Full Cluster To Simple Cluster _(), and Manual Roi_Cluster _Full Callback().$

5.16.1.11 ManualRoi_Scan_FullCallback()

Manually extract Rol, run scan and apply a full call-back.

Parameters

alnFile	The name of the localization file
aManualRol	The manually-specified Rol window
aScanConfig	The configuration for the scan
aCallback	The full callback to be applied

Definition at line 118 of file API.cpp.

References LocalizationFile::ExtractRols(), and Rol::ScanRT().

5.16.1.12 ManualRoi_Scan_SimpleCallback()

Manually extract Rol, run scan and apply a simple call-back.

Parameters

alnFile	The name of the localization file
aManualRol	The manually-specified Rol window
aScanConfig	The configuration for the scan
aCallback	The simple callback to be applied

Definition at line 124 of file API.cpp.

References _FullScanToSimpleScan_(), and LocalizationFile::ExtractRols().

Referenced by ManualRoi_Scan_ToJson().

5.16.1.13 ManualRoi_Scan_ToJson()

Manually extract Rol, run scan and dump to JSON file.

Parameters

alnFile	The name of the localization file
aManualRol	The manually-specified Rol window
aScanConfig	The configuration for the scan
aOutFile	The name of the output JSON file

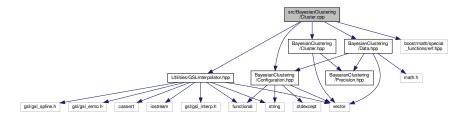
Definition at line 130 of file API.cpp.

References _ScanCallback_Json_(), and ManualRoi_Scan_SimpleCallback().

Referenced by BOOST_PYTHON_MODULE().

5.17 src/BayesianClustering/Cluster.cpp File Reference

```
#include "Utilities/GSLInterpolator.hpp"
#include <boost/math/special_functions/erf.hpp>
#include "BayesianClustering/Cluster.hpp"
#include "BayesianClustering/Data.hpp"
#include "BayesianClustering/Configuration.hpp"
Include dependency graph for Cluster.cpp:
```



Functions

double normal_cdf (const double &x, const double &sigma=1, const double &x0=0)

Evaluate the Gaussian normal_cdf at a given position Copied from the CERN ROOT implementaion, swap ROOT :::Math::erfc and ROOT::Math::erf for the boost::math version.

5.17.1 Function Documentation

5.17.1.1 normal_cdf()

Evaluate the Gaussian normal_cdf at a given position Copied from the CERN ROOT implementation, swap ROOT :: Math::erfc and ROOT::Math::erf for the boost::math version.

Parameters

Х	The position to evaluate the normal_cdf at
sigma	The standard-deviation of the Gaussian
х0	The mean of the Gaussian

Returns

the value of the Gaussian normal cdf at x

Definition at line 22 of file Cluster.cpp.

Referenced by Cluster::Parameter::alt_log_score(), and Cluster::Parameter::log_score().

5.18 src/BayesianClustering/Configuration.cpp File Reference

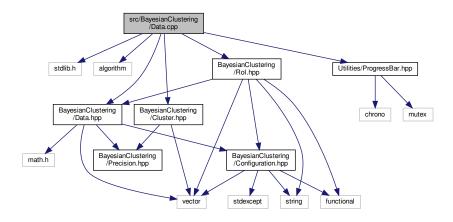
```
#include "Utilities/GSLInterpolator.hpp"
#include "Utilities/ListComprehension.hpp"
#include "Utilities/Vectorize.hpp"
#include "Utilities/Units.hpp"
#include "BayesianClustering/Configuration.hpp"
#include <iostream>
#include <fstream>
#include <streambuf>
#include <boost/math/special_functions/gamma.hpp>
#include "boost/algorithm/string.hpp"
#include "boost/program_options.hpp"
Include dependency graph for Configuration.cpp:
```



5.19 src/BayesianClustering/Data.cpp File Reference

```
#include <stdlib.h>
#include <algorithm>
#include "BayesianClustering/Data.hpp"
#include "BayesianClustering/Cluster.hpp"
#include "BayesianClustering/RoI.hpp"
```

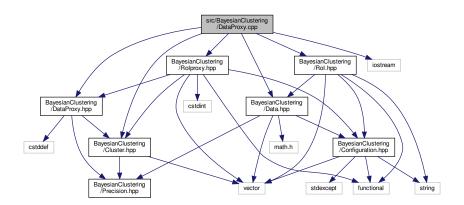
#include "Utilities/ProgressBar.hpp"
Include dependency graph for Data.cpp:



5.20 src/BayesianClustering/DataProxy.cpp File Reference

```
#include "BayesianClustering/DataProxy.hpp"
#include "BayesianClustering/Data.hpp"
#include "BayesianClustering/Cluster.hpp"
#include "BayesianClustering/RoIproxy.hpp"
#include "BayesianClustering/RoI.hpp"
#include <iostream>
```

Include dependency graph for DataProxy.cpp:



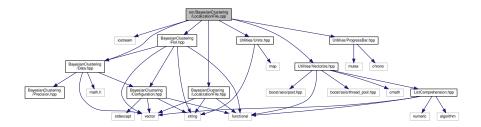
Macros

• #define RECURSION_LIMIT 75000

The maximum depth for recursive clustering.

5.21 src/BayesianClustering/LocalizationFile.cpp File Reference

```
#include <iostream>
#include "BayesianClustering/LocalizationFile.hpp"
#include "BayesianClustering/Data.hpp"
#include "BayesianClustering/RoI.hpp"
#include "Utilities/ProgressBar.hpp"
#include "Utilities/Vectorize.hpp"
#include "Utilities/Units.hpp"
Include dependency graph for LocalizationFile.cpp:
```



Typedefs

typedef std::array< std::array< int, 512 >, 512 > tArray
 Typedef an array for histogramming a Localization File.

Functions

void <u>LoadCSV</u> (const std::string &aFilename, std::vector < <u>Data</u> > &aData, const std::size_t &aOffset, int aCount)

Multithreading handler for loading a chunk of data from CSV file.

• void __RecursiveSearch__ (tArray &aHist, const int &aRolid, const int &i, const int &j)

Recursively search histogram for continuously connected regions over threshold.

5.21.1 Function Documentation

5.21.1.1 __LoadCSV__()

Multithreading handler for loading a chunk of data from CSV file.

Parameters

aFilename	The name of the file to open
aData	A vector into which to fill data
aOffset	The offset into the file
aCount	The (approximate) number of bytes to be handled by this handler

Definition at line 23 of file LocalizationFile.cpp.

References nanometer.

Referenced by LocalizationFile::LocalizationFile().

5.21.1.2 __RecursiveSearch__()

Recursively search histogram for continuously connected regions over threshold.

Parameters

aHist	The histogram being searched
aRolid	The id of the region being allocated
i	The horizontal index of the current cell in the histogram
j	The vertical index of the current cell in the histogram

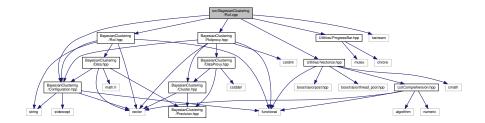
Definition at line 109 of file LocalizationFile.cpp.

Referenced by LocalizationFile::ExtractRols().

5.22 src/BayesianClustering/Rol.cpp File Reference

```
#include "BayesianClustering/RoI.hpp"
#include "BayesianClustering/RoIproxy.hpp"
#include "BayesianClustering/Configuration.hpp"
#include "Utilities/ProgressBar.hpp"
#include "Utilities/Vectorize.hpp"
#include <iostream>
```

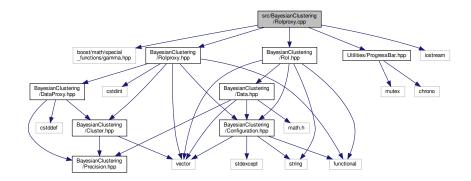
Include dependency graph for Rol.cpp:



5.23 src/BayesianClustering/Rolproxy.cpp File Reference

```
#include <boost/math/special_functions/gamma.hpp>
#include "BayesianClustering/RoIproxy.hpp"
#include "BayesianClustering/RoI.hpp"
#include "Utilities/ProgressBar.hpp"
#include <iostream>
```

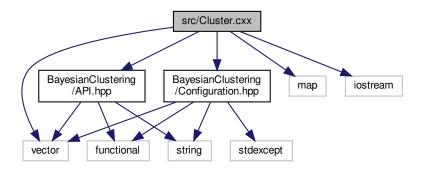
Include dependency graph for Rolproxy.cpp:



5.24 src/Cluster.cxx File Reference

```
#include "BayesianClustering/API.hpp"
#include "BayesianClustering/Configuration.hpp"
#include <map>
#include <vector>
#include <iostream>
```

Include dependency graph for Cluster.cxx:



Functions

- void ReportClusters (const std::vector< ClusterWrapper > &aClusters)
 - Callback to report clusters.
- int main (int argc, char **argv)

The main function.

5.24.1 Function Documentation

5.24.1.1 main()

```
int main (
          int argc,
          char ** argv )
```

The main function.

Parameters

argc	The number of commandline arguments
argv	The commandline arguments

Returns

The exit code

Definition at line 28 of file Cluster.cxx.

References AuxConfiguration::ClusterR(), AuxConfiguration::ClusterT(), AuxConfiguration::inputFile(), and ReportClusters().

5.24.1.2 ReportClusters()

Callback to report clusters.

Parameters

aClusters	A vector of clusters
-----------	----------------------

Definition at line 14 of file Cluster.cxx.

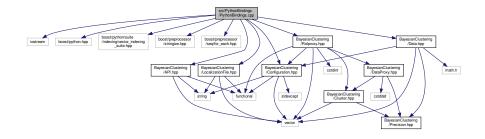
Referenced by main().

5.25 src/PythonBindings/PythonBindings.cpp File Reference

Self-contained sourcefile for producing python-bindings.

```
#include <iostream>
#include <boost/python.hpp>
#include <boost/python/suite/indexing/vector_indexing_suite.hpp>
#include <boost/preprocessor/stringize.hpp>
#include <boost/preprocessor/seq/for_each.hpp>
#include "BayesianClustering/API.hpp"
#include "BayesianClustering/LocalizationFile.hpp"
#include "BayesianClustering/Configuration.hpp"
#include "BayesianClustering/RoIproxy.hpp"
#include "BayesianClustering/Data.hpp"
```

Include dependency graph for PythonBindings.cpp:



Macros

Helper macro to define a constructor for a std::function of a given signature from a boost::python callback object Using explicit specialization of the class to the specified type and explicit specialization of the constructor to take a boost::python::object (hence two template<>>'s at the start), create a generic variadic lambda that captures the boost::python callback object and pass the lambda to a deferred constructor.

#define FN(X, ...) def(#X , &X , args(__VA_ARGS__))

Helper Macro to simplify defining functions.

#define ADAPTED_FN(X, ...) def(#X , +Adapted::X , args(__VA_ARGS__))

Helper Macro to simplify defining functions with python callbacks.

- #define STRUCT_ARG(r, CLASS, ARG) .def_readwrite(BOOST_PP_STRINGIZE(ARG) , &CLASS::ARG) Helper Macro to deal with the boilerplate when dealing with structs.
- #define EXPOSE_STRUCT(CLASS, ARGS) class_< CLASS >(#CLASS) BOOST_PP_SEQ_FOR_EACH(STRUCT_ARG , CLASS , ARGS)

Helper Macro to deal with the boilerplate when dealing with structs.

#define EXPOSE_VECTOR(CLASS) class_< std::vector< CLASS> >("Vector##CLASS").def(vector_← indexing_suite< std::vector< CLASS > >());

Helper Macro to deal with the boilerplate when dealing with vectors of objects.

Functions

- ADAPT_CALLBACK_CONSTRUCTOR (std::function < void(Rolproxy &, const double &, const double &) >)
 Define a std::function constructor for full scan callback.
- ADAPT_CALLBACK_CONSTRUCTOR (std::function < void(const std::vector < ScanEntry > &) >)

Define a std::function constructor for simple scan callback.

ADAPT CALLBACK CONSTRUCTOR (std::function < void(Rolproxy &) >)

Define a std::function constructor for full clusterizer callback.

- ADAPT_CALLBACK_CONSTRUCTOR (std::function < void(const std::vector < ClusterWrapper > &) >)
 Define a std::function constructor for simple clusterizer callback.
- BOOST PYTHON MODULE (BayesianClustering)

Boost Python Wrapper providing bindings for our C++ functions.

Variables

auto Adapted::AutoRoi_Scan_SimpleCallback = [](const std::string& alnFile , const ScanConfiguration& aScanConfig , const boost::python::object& aCallback){ ::AutoRoi_Scan_SimpleCallback(alnFile , aScan← Config , aCallback); }

Lambda to automatically extract Rol, run scan and apply a simple python callback.

• auto Adapted::AutoRoi_Cluster_SimpleCallback = [](const std::string& alnFile , const double& aR , const double& aT , const boost::python::object& aCallback){ ::AutoRoi_Cluster_SimpleCallback(alnFile , aR , aT , aCallback); }

Lambda to automatically extract Rol, clusterize and apply a simple python callback.

auto Adapted::ManualRoi_Scan_SimpleCallback = [](const std::string& alnFile , const ManualRol& a
 ManualRol , const ScanConfiguration& aScanConfig , const boost::python::object& aCallback){ ::Manual
 Roi_Scan_SimpleCallback(alnFile , aManualRol , aScanConfig , aCallback); }

Lambda to manually extract Rol, run scan and apply a simple python callback.

auto Adapted::ManualRoi_Cluster_SimpleCallback = [](const std::string& alnFile , const ManualRol& a
 ManualRol , const double& aR , const double& aT , const boost::python::object& aCallback){ ::ManualRoi
 — Cluster_SimpleCallback(alnFile , aManualRol , aR , aT , aCallback); }

Lambda to manually extract Rol, clusterize and apply a simple python callback.

5.25.1 Detailed Description

Self-contained sourcefile for producing python-bindings.

5.25.2 Macro Definition Documentation

5.25.2.1 ADAPT_CALLBACK_CONSTRUCTOR

```
#define ADAPT_CALLBACK_CONSTRUCTOR( SIGNATURE \ ) \ template<> \ SIGNATURE::function< boost::python::object& >( boost::python::object& aCallback ) : function( [&] ( auto&&... aArgs ) { aCallback ( a \leftarrow Args... ); } ) {}
```

Helper macro to define a constructor for a std::function of a given signature from a boost::python callback object Using explicit specialization of the class to the specified type and explicit specialization of the constructor to take a boost::python::object (hence two template<>'s at the start), create a generic variadic lambda that captures the boost::python callback object and pass the lambda to a deferred constructor.

Gnarly!

Parameters

SIGNATURE	The signature of the std::function we are creating a constructor for
-----------	--

Definition at line 38 of file PythonBindings.cpp.

5.25.2.2 ADAPTED_FN

Helper Macro to simplify defining functions with python callbacks.

Parameters

X	The function being defined
	List of strings giving argument names

Definition at line 73 of file PythonBindings.cpp.

5.25.2.3 EXPOSE_STRUCT

Helper Macro to deal with the boilerplate when dealing with structs.

Parameters

CLASS	The Class name
ARGS	One of the arguments

Definition at line 84 of file PythonBindings.cpp.

5.25.2.4 EXPOSE_VECTOR

Helper Macro to deal with the boilerplate when dealing with vectors of objects.

Parameters

|--|

Definition at line 88 of file PythonBindings.cpp.

5.25.2.5 FN

Helper Macro to simplify defining functions.

Parameters

Χ	The function being defined
	List of strings giving argument names

Definition at line 68 of file PythonBindings.cpp.

5.25.2.6 STRUCT ARG

Helper Macro to deal with the boilerplate when dealing with structs.

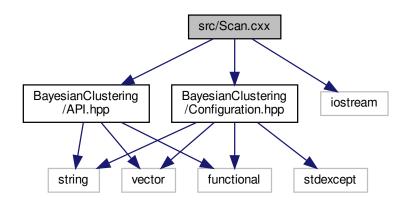
Parameters

r	BOOST PP internal
CLASS	The Class name
ARG	One of the arguments

Definition at line 79 of file PythonBindings.cpp.

5.26 src/Scan.cxx File Reference

```
#include "BayesianClustering/API.hpp"
#include "BayesianClustering/Configuration.hpp"
#include <iostream>
Include dependency graph for Scan.cxx:
```



Functions

• int main (int argc, char **argv)

The main function.

5.26.1 Function Documentation

5.26.1.1 main()

```
int main (  \mbox{int $argc$,} \\ \mbox{char $**$ $argv$ )}
```

The main function.

Parameters

argc	The number of commandline arguments
argv	The commandline arguments

Returns

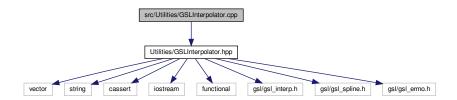
The exit code

Definition at line 15 of file Scan.cxx.

References AutoRoi_Scan_ToJson(), AuxConfiguration::configFile(), AuxConfiguration::inputFile(), and Aux \hookleftarrow Configuration::outputFile().

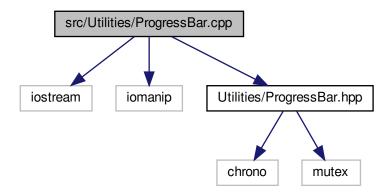
5.27 src/Utilities/GSLInterpolator.cpp File Reference

#include "Utilities/GSLInterpolator.hpp"
Include dependency graph for GSLInterpolator.cpp:



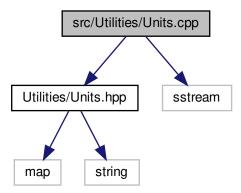
5.28 src/Utilities/ProgressBar.cpp File Reference

#include <iostream>
#include <iomanip>
#include "Utilities/ProgressBar.hpp"
Include dependency graph for ProgressBar.cpp:



5.29 src/Utilities/Units.cpp File Reference

```
#include "Utilities/Units.hpp"
#include <sstream>
Include dependency graph for Units.cpp:
```



Functions

long double StrToDist (const std::string &aStr)
 Convert a string representation to a distance.

Variables

• const std::map< std::string, double > UnitMap { {"nm",nanometer}, {"um",micrometer}, {"mm",millimeter}, {"m",meter} }

A map for converting string representations of SI units to scaling factors.

5.29.1 Function Documentation

5.29.1.1 StrToDist()

Convert a string representation to a distance.

Parameters

aStr A string representation of a distance

Returns

The literal value

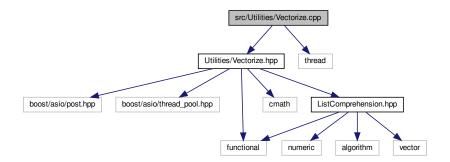
Definition at line 12 of file Units.cpp.

References UnitMap.

Referenced by ScanConfiguration::FromVector(), and AuxConfiguration::FromVector().

5.30 src/Utilities/Vectorize.cpp File Reference

#include "Utilities/Vectorize.hpp"
#include <thread>
Include dependency graph for Vectorize.cpp:



Variables

std::size_t Nthreads = std::thread::hardware_concurrency()

The number of threads used, initialized to the number of hardware threads.

5.30.1 Variable Documentation

5.30.1.1 Nthreads

std::size_t Nthreads = std::thread::hardware_concurrency()

The number of threads used, initialized to the number of hardware threads.

Utility variable for the concurrency.

Definition at line 8 of file Vectorize.cpp.

Referenced by AuxConfiguration::FromVector(), LocalizationFile::LocalizationFile(), operator & (), operator | |(), and Rol::ScanRT().

Index

FullClusterToSimpleCluster	AutoRoi_Scan_FullCallback
API.cpp, 102	API.cpp, 104
FullScanToSimpleScan	API.hpp, 78
API.cpp, 102	AutoRoi_Scan_SimpleCallback
_ScanCallback_Json_	API.cpp, 104
API.cpp, 103	API.hpp, 79
LoadCSV	AutoRoi_Scan_ToJson
LocalizationFile.cpp, 111	API.cpp, 105
RecursiveSearch	API.hpp, 79
LocalizationFile.cpp, 112	AuxConfiguration, 7
•••	AuxConfiguration, 8, 9
ADAPT_CALLBACK_CONSTRUCTOR	ClusterR, 9
PythonBindings.cpp, 117	ClusterT, 9
ADAPTED_FN	configFile, 9
PythonBindings.cpp, 117	FromVector, 10
alpha	inputFile, 10
ScanConfiguration, 62	outputFile, 10
alt_log_score	SetConfigFile, 11
Cluster::Parameter, 40	SetInputFile, 11
API.cpp	SetOutputFile, 12
FullClusterToSimpleCluster , 102	SetValidate, 12
FullScanToSimpleScan, 102	validate, 12
_ScanCallback_Json_, 103	validate, 12
AutoRoi Cluster FullCallback, 103	CalculateLocalizationScore
AutoRoi_Cluster_SimpleCallback, 104	Data, 21
AutoRoi_Scan_FullCallback, 104	CheckClusterization
AutoRoi_Scan_SimpleCallback, 104	Rolproxy, 54
AutoRoi Scan ToJson, 105	Cluster, 13
ManualRoi_Cluster_FullCallback, 105	Cluster, 14, 15
ManualRoi_Cluster_SimpleCallback, 106	GetParent, 15
ManualRoi_Scan_FullCallback, 106	operator+=, 16
ManualRoi_Scan_SimpleCallback, 107	operator=, 16
ManualRoi Scan ToJson, 107	UpdateLogScore, 17
API.hpp	
• •	Cluster.cpp
AutoRoi_Cluster_FullCallback, 76	normal_cdf, 108
AutoRoi_Cluster_SimpleCallback, 78	Cluster.cxx
AutoRoi_Scan_FullCallback, 78	main, 114
AutoRoi_Scan_SimpleCallback, 79	ReportClusters, 114
AutoRoi_Scan_ToJson, 79	Cluster::Parameter, 40
ManualRoi_Cluster_FullCallback, 79	alt_log_score, 40
ManualRoi_Cluster_SimpleCallback, 80	log_score, 41
ManualRoi_Scan_FullCallback, 80	operator+=, 41
ManualRoi_Scan_SimpleCallback, 81	Clusterize
ManualRoi_Scan_ToJson, 81	DataProxy, 27, 28
AutoRoi_Cluster_FullCallback	Rol, 47
API.cpp, 103	Rolproxy, 54
API.hpp, 76	ClusterR
AutoRoi_Cluster_SimpleCallback	AuxConfiguration, 9
API.cpp, 104	ClusterT
API.hpp, 78	AuxConfiguration, 9

124 INDEX

ClusterWrapper, 17	Cluster, 15
operator==, 18	getWidthX
configFile	Rol, 48
AuxConfiguration, 9	getWidthY
	Rol, 49
Data, 19	GSLInterpolator, 29
CalculateLocalizationScore, 21	Deriv, 32
Data, 20, 21	Deriv2, 32
dPhi, 22	Eval, 33
dR, 22	Evaluate, 33
dR2, 22	GSLInterpolator, 30, 31
operator<, 23	Integ, 33
operator=, 23, 24	operator=, 34
Preprocess, 24	SetData, 35
PreprocessLocalizationScores, 25	
DataProxy, 25	include/BayesianClustering/API.hpp, 75
Clusterize, 27, 28	include/BayesianClustering/Cluster.hpp, 82
DataProxy, 26, 27	include/BayesianClustering/Configuration.hpp, 83
GetCluster, 28	include/BayesianClustering/Data.hpp, 84
operator=, 28, 29	include/BayesianClustering/DataProxy.hpp, 85
Deriv	include/BayesianClustering/LocalizationFile.hpp, 86
GSLInterpolator, 32	include/BayesianClustering/Precision.hpp, 87
Deriv2	include/BayesianClustering/Rol.hpp, 87
GSLInterpolator, 32	include/BayesianClustering/Rolproxy.hpp, 88
dPhi	include/Utilities/GSLInterpolator.hpp, 89
Data, 22	include/Utilities/ListComprehension.hpp, 89
dR	include/Utilities/MemoryMonitoring.hpp, 93
Data, 22	include/Utilities/ProgressBar.hpp, 94
dR2	include/Utilities/Units.hpp, 95
Data, 22	include/Utilities/Vectorize.hpp, 98
- 	inputFile
Eval	AuxConfiguration, 10
GSLInterpolator, 33	Integ
Evaluate	GSLInterpolator, 33
GSLInterpolator, 33	actine polator, co
EXPOSE STRUCT	ListComprehension.hpp
PythonBindings.cpp, 117	operator , 90-92
EXPOSE_VECTOR	range, 92
PythonBindings.cpp, 118	LocalizationFile, 36
ExtractRols	ExtractRols, 37, 38
LocalizationFile, 37, 38	LocalizationFile, 36, 37
	operator=, 38
FN	LocalizationFile.cpp
PythonBindings.cpp, 118	LoadCSV, 111
FromVector	RecursiveSearch , 112
AuxConfiguration, 10	log_probability_sigma
ScanConfiguration, 62	ScanConfiguration, 63
	log score
getArea	Cluster::Parameter, 41
Rol, 47	logAlpha
getCentreX	ScanConfiguration, 63
Rol, 48	logGammaAlpha
getCentreY	ScanConfiguration, 64
Rol, 48	logPb
GetCluster	ScanConfiguration, 64
DataProxy, 28	logPbDagger
GetData	ScanConfiguration, 64
Rolproxy, 55	ocanooninguration, 04
GetParent	main

INDEX 125

Cluster.cxx, 114	outputFile
Scan.cxx, 119	AuxConfiguration, 10
ManualRol, 39	Advortingulation, To
ManualRoi_Cluster_FullCallback	Preprocess
API.cpp, 105	Data, 24
API.hpp, 79	Rol, 50
ManualRoi_Cluster_SimpleCallback	PreprocessLocalizationScores
API.cpp, 106	Data, 25
API.hpp, 80	probability_sigma
ManualRoi Scan FullCallback	ScanConfiguration, 66
API.cpp, 106	ProgressBar, 42
API.hpp, 80	operator++, 43
ManualRoi_Scan_SimpleCallback	ProgressBar, 43
API.cpp, 107	ProgressTimer, 44
API.hpp, 81	ProgressTimer, 44
ManualRoi_Scan_ToJson	PythonBindings.cpp
API.cpp, 107	ADAPT_CALLBACK_CONSTRUCTOR, 117
API.hpp, 81	ADAPTED_FN, 117
mem_usage	EXPOSE_STRUCT, 117
MemoryMonitoring.hpp, 93	EXPOSE_VECTOR, 118
MemoryMonitoring.hpp	FN, 118
mem_usage, 93	STRUCT_ARG, 118
normal_cdf	range
Cluster.cpp, 108	ListComprehension.hpp, 92 Rbounds
Nthreads	
Vectorize.cpp, 122	ScanConfiguration, 66
Vectorize.hpp, 100	ReportClusters Cluster.cxx, 114
operator/	Rol, 45
operator<	
Data, 23 ScanEntry, 72	Clusterize, 47 getArea, 47
operator++	getCentreX, 48
ProgressBar, 43	getCentreY, 48
operator+=	getWidthX, 48
Cluster, 16	getWidthY, 49
Cluster::Parameter, 41	operator=, 49
operator=	Preprocess, 50
Cluster, 16	Rol, 46, 47
Data, 23, 24	ScanRT, 50
DataProxy, 28, 29	SetCentre, 51
GSLInterpolator, 34	SetWidth, 51
LocalizationFile, 38	Rolproxy, 52
Rol, 49	CheckClusterization, 54
Rolproxy, 55, 56	Clusterize, 54
ScanConfiguration, 65	GetData, 55
operator==	operator=, 55, 56
ClusterWrapper, 18	Rolproxy, 53, 54
ScanEntry, 72	ScanRT, 56
operator&&	UpdateLogScore, 56
Vectorize.hpp, 99	ValidateLogScore, 57
operator""_micrometer	- aa = - a.g. = - a
Units.hpp, 96	Scan.cxx
operator""_nanometer	main, 119
Units.hpp, 97	ScanConfiguration, 57
operator	alpha, 62
ListComprehension.hpp, 90–92	FromVector, 62
operator	log_probability_sigma, 63
Vectorize.hpp, 99	logAlpha, 63
voctorizonipp, vv	- U 7

126 INDEX

logGammaAlpha, 64	src/BayesianClustering/Configuration.cpp, 109
logPb, 64	src/BayesianClustering/Data.cpp, 109
logPbDagger, 64	src/BayesianClustering/DataProxy.cpp, 110
operator=, 65	src/BayesianClustering/LocalizationFile.cpp, 111
probability_sigma, 66	src/BayesianClustering/Rol.cpp, 112
Rbounds, 66	src/BayesianClustering/Rolproxy.cpp, 113
ScanConfiguration, 60–62	src/Cluster.cxx, 113
SetAlpha, 66	src/PythonBindings/PythonBindings.cpp, 115
SetPb, 67	src/Scan.cxx, 119
SetRBins, 67	src/Utilities/GSLInterpolator.cpp, 120
SetSigmaParameters, 68	src/Utilities/ProgressBar.cpp, 120
SetTBins, 68	src/Utilities/Units.cpp, 121
sigmabins, 69	src/Utilities/Vectorize.cpp, 122
sigmabins2, 69, 70	StrToDist
sigmacount, 70	Units.cpp, 121
sigmaspacing, 70	Units.hpp, 97
Tbounds, 71	STRUCT ARG
ScanConfiguration::tBounds, 73	PythonBindings.cpp, 118
ScanEntry, 71	7 3 1 - 1
operator<, 72	Tbounds
operator==, 72	ScanConfiguration, 71
ScanRT	oodgaration, 77
	Units.cpp
Rol, 50	StrToDist, 121
Rolproxy, 56	Units.hpp
SetAlpha	operator""_micrometer, 96
ScanConfiguration, 66	operator""_nanometer, 97
SetCentre	
Rol, 51	StrToDist, 97
SetConfigFile	UpdateLogScore
AuxConfiguration, 11	Cluster, 17
SetData	Rolproxy, 56
GSLInterpolator, 35	
SetInputFile	validate
AuxConfiguration, 11	AuxConfiguration, 12
SetOutputFile	ValidateLogScore
AuxConfiguration, 12	Rolproxy, 57
SetPb	Vectorize.cpp
	Nthreads, 122
ScanConfiguration, 67	Vectorize.hpp
SetRBins	Nthreads, 100
ScanConfiguration, 67	operator&&, 99
SetSigmaParameters	operator , 99
ScanConfiguration, 68	1
SetTBins	
ScanConfiguration, 68	
SetValidate	
AuxConfiguration, 12	
SetWidth	
Rol, 51	
sigmabins	
ScanConfiguration, 69	
sigmabins2	
ScanConfiguration, 69, 70	
_	
sigmacount ScanConfiguration 70	
ScanConfiguration, 70	
sigmaspacing	
ScanConfiguration, 70	
src/BayesianClustering/API.cpp, 101	
src/BayesianClustering/Cluster.cpp, 108	