



CrossGrid Developer Manual Guide

SLEUTH-G Application

WP 1.3.2

Document Filename:	CG-DeveloperManual
Workpackage:	WP 1.3.2
Partner(s):	INS
Lead Partner:	INS
Config ID:	cg-developermanual-v0.1
Document classification:	PUBLIC

Abstract: This is a developer manual for the parallel version of the SLEUTH algorithm developed within the CrossGrid project.



Delivery Slip

	Name	Partner	Date	Signature
From	Jan Iwaszkiewicz	ICM	Jan 2005	
Verified By				
Approved By				

Document Log

Version	Date	Summary of changes	Author
0.1	Jan 28th, 2005	First draft version	Jan Iwaszkiewicz

Contents

Copyright Notice	6
1 Introduction	7
1.1 Abbreviations and Acronyms	7
1.2 References and Source Code	7
2 Implementation Structure	8
2.1 Product Use Cases	8
2.2 Product Component Model	8
2.3 Architecture - parallelization	8
3 CG_WP_1_3_2-SLEUTH Namespace Index	11
3.1 CG_WP_1_3_2-SLEUTH Namespace List	11
4 CG_WP_1_3_2-SLEUTH Hierarchical Index	12
4.1 CG_WP_1_3_2-SLEUTH Class Hierarchy	12
5 CG_WP_1_3_2-SLEUTH Class Index	13
5.1 CG_WP_1_3_2-SLEUTH Class List	13
6 CG_WP_1_3_2-SLEUTH File Index	14
6.1 CG_WP_1_3_2-SLEUTH File List	14
7 CG_WP_1_3_2-SLEUTH Namespace Documentation	16
7.1 GlobalVariables Namespace Reference	16
7.2 IOUtilities Namespace Reference	17
7.3 Math Namespace Reference	18
7.4 RCPPParameters Namespace Reference	28
7.5 std Namespace Reference	30
8 Class Documentation	31
8.1 Anomalousness Class Reference	31
8.2 BkgEvent Class Reference	33
8.3 BkgSet Class Reference	35
8.4 BkgSubSet Class Reference	39
8.5 DataEvent Class Reference	41
8.6 DataSet Class Reference	43
8.7 EmptyEventSet Class Reference	46

8.8	Event Class Reference	47
8.9	EventSet< EventType > Class Template Reference	49
8.10	Math::FunctionObject Class Reference	51
8.11	generateRequest Struct Reference	52
8.12	Hse Class Reference	53
8.13	HseSet Class Reference	54
8.14	IDNumber Class Reference	56
8.15	Math::KeepNSmallest Class Reference	57
8.16	Math::KeepWSmallest Class Reference	58
8.17	matrix Class Reference	59
8.18	MultiChannelResult Class Reference	62
8.19	Problem Class Reference	63
8.20	Solution Class Reference	64
9	CG_WP_1_3_2-SLEUTH File Documentation	66
9.1	addTwiddle.cpp File Reference	66
9.2	Anomalousness.cpp File Reference	67
9.3	Anomalousness.hpp File Reference	68
9.4	BkgEvent.cpp File Reference	69
9.5	BkgEvent.hpp File Reference	70
9.6	BkgSet.cpp File Reference	71
9.7	BkgSet.hpp File Reference	72
9.8	BkgSubSet.cpp File Reference	73
9.9	BkgSubSet.hpp File Reference	74
9.10	calc.cpp File Reference	75
9.11	combineScriptPs.cpp File Reference	76
9.12	DataEvent.cpp File Reference	78
9.13	DataEvent.hpp File Reference	79
9.14	DataSet.cpp File Reference	80
9.15	DataSet.hpp File Reference	81
9.16	EmptyEventSet.hpp File Reference	82
9.17	Event.cpp File Reference	83
9.18	Event.hpp File Reference	84
9.19	EventSet.hpp File Reference	85
9.20	GlobalVariables.cpp File Reference	86
9.21	GlobalVariables.hpp File Reference	87
9.22	grid.cpp File Reference	88
9.23	grid.hpp File Reference	90
9.24	gSleuth-ocmg.cpp File Reference	92
9.25	gSleuth.cpp File Reference	93

9.26 Hse.cpp File Reference	94
9.27 Hse.hpp File Reference	95
9.28 HseSet.cpp File Reference	96
9.29 HseSet.hpp File Reference	97
9.30 IDNumber.cpp File Reference	98
9.31 IDNumber.hpp File Reference	99
9.32 IOUtilities.cpp File Reference	100
9.33 IOUtilities.hpp File Reference	101
9.34 jacobi.cpp File Reference	102
9.35 lubksb.cpp File Reference	103
9.36 ludcmp.cpp File Reference	104
9.37 Math.cpp File Reference	105
9.38 Math.hpp File Reference	106
9.39 matrix.cpp File Reference	107
9.40 matrix.hpp File Reference	108
9.41 MultiChannelResult.cpp File Reference	109
9.42 MultiChannelResult.hpp File Reference	110
9.43 nrtutil.cpp File Reference	111
9.44 nrtutil.hpp File Reference	115
9.45 Problem.hpp File Reference	119
9.46 RCPPParameters.cpp File Reference	120
9.47 RCPPParameters.hpp File Reference	121
9.48 simp1.cpp File Reference	122
9.49 simp2.cpp File Reference	123
9.50 simp3.cpp File Reference	124
9.51 simplx.cpp File Reference	125
9.52 sleuth.cpp File Reference	126
9.53 sleuth.hpp File Reference	127
9.54 Solution.cpp File Reference	128
9.55 Solution.hpp File Reference	129
9.56 zbrent.cpp File Reference	130
10 Product Testing	131
11 EDG License Agreement	133

Copyright Notice

Copyright (c) 2005 by **A.Soltan Institute for Nuclear Studies (INS)**. All rights reserved.

Use of this product is subject to the terms and licenses stated in the EDG license agreement

SLEUTH is a registered trademark of **Bruce Knuteson**. All rights reserved.

This research is partly funded by the European Commission IST-2001-32243 Project "CrossGrid".

1 Introduction

The SLEUTH-G is a grid version of sequential SLEUTH algorithm, developed by B. Knuteson from Fermilab. It can be used for evidence of physics beyond the Standard Model in a quasi-model-independent way. It was used on data collected by Tevatron. Detailed description of the method and results of search at the D0 experiment are described in ref. [1].

SLEUTH is an algorithm of finding regions of excess in HEP experiment's data. The excess is measured in comparison to expected background. The background is simulated experiment data which is based on Standard Model. SLEUTH also quantifies the significance of detected excess. Such fiducial regions are volumes in the space defined by physical variables such as transverse momenta of reconstructed objects. These regions contain some data points and expected numbers of background events. For each region of excess SLEUTH computes the probability, that the expected background fluctuates up to or beyond the level of real data. If this probability is small, then the region is potentially interesting. The probability space is potentially a complicated multidimensional distribution of physics variables and only numerical approximation of the probability is possible. SLEUTH determines it by generating hypothetical similar experiments (HSEs) i.e. by generating the random sets of events according to the background distribution and checking how often they are above expectations in the considered, potentially interesting region.

1.1 Abbreviations and Acronyms

CrossGrid The EU CrossGrid Project IST-2001-32243

MPI Message Passing Interface [2]

MPICH Freely available implementation of MPI [3]

MPICH-G2 Grid-enabled implementation of MPICH [5]

1.2 References and Source Code

Sleuth-G source code is available on GridPortal at:

http://savannah.fzk.de/cgi-bin/viewcvs.cgi/crossgrid/crossgrid/wp1/wp1_3-hep/wp1_3_2-sleuth/

2 Implementation Structure

2.1 Product Use Cases

SLEUTH-G can be executed in two ways: on local cluster and in grid environment. In both cases it requires MPI library. Sleuth-G is designed to work with GRID GUI. For local running the shell interface is provided.

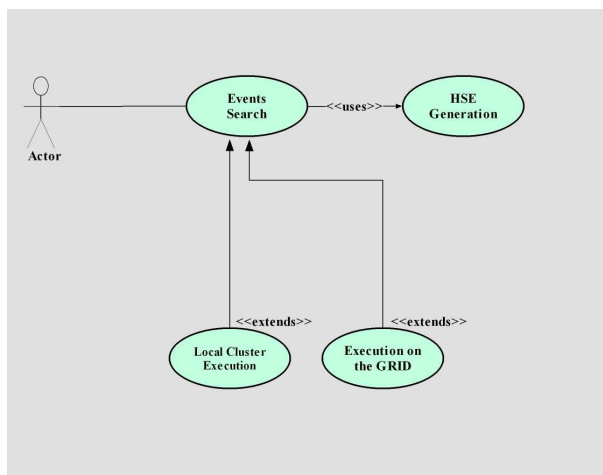


Figure 2.1: Use Cases for Sleuth-G

2.2 Product Component Model

The project consist of two main components: sequential algorithm (Sleuth) and the parallelization part which enables it to be run in GRID environment and provides interoperation with the GRID GUI. As the GRID GUI we use Migrating Desktop [4].

2.3 Architecture - parallelization

Computations of SLEUTH algorithm are performed in Master - Worker architecture. The master node defines fiducial regions, asks for generation of HSEs, collects them and computes probabilities for regions. Worker nodes generate HSEs. Generation of each HSE is independent of other HSEs so many worker nodes can work in parallel.

The application is parallelized in such a way that it can adjust to the architecture of available resources. Each of the worker nodes gets an order form the master node, generates the desired number of HSEs and repeats these actions until it receives an empty order. Thanks to this kind of load balancing there is no bottleneck effect. The resources are used efficiently even if there are differences between nodes.

The algorithm was parallelized with *Message Passing Interface* (MPI) [2]. MPI standard is the mainstream solution for parallel applications in science and has many implementations. We used MPICH [3] which is a freely available

implementation. We tested SLEUTH with two kinds of MPICH, which are available on the CrossGrid testbed: MPICH-P4 (using `ch_p4` device), working on only one cluster and grid enabled MPICH-G2 [5]. The later one uses Globus Toolkit 2 [7] to access grid resources and to perform secure communication between distributed nodes.

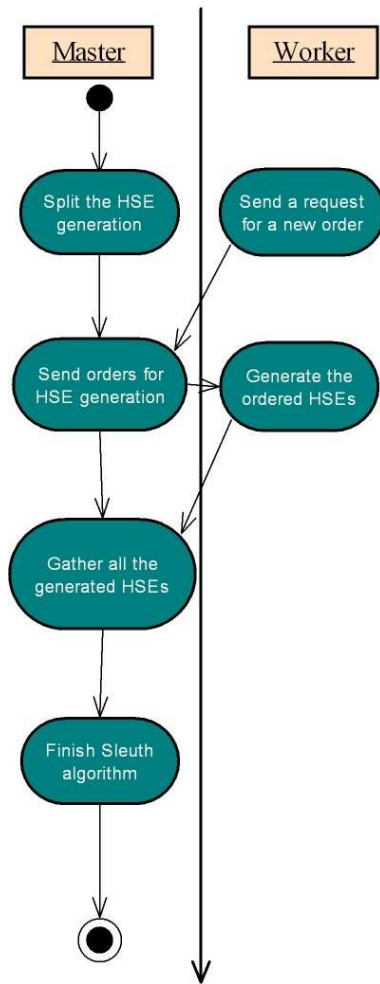


Figure 2.2: Master - Worker model in Sleuth-G application

3 CG_WP_1_3_2-SLEUTH Namespace Index

3.1 CG_WP_1_3_2-SLEUTH Namespace List

Here is a list of all namespaces with brief descriptions:

GlobalVariables	16
IOUtilities	17
Math	18
RCPPParameters	28
std	30

4 CG_WP_1_3_2-SLEUTH Hierarchical Index

4.1 CG_WP_1_3_2-SLEUTH Class Hierarchy

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

Anomalousness	31
BkgSet	35
EmptyEventSet	46
Event	47
BkgEvent	33
DataEvent	41
EventSet< EventType >	49
EventSet< BkgEvent >	49
BkgSubSet	39
EventSet< DataEvent >	49
DataSet	43
Math::FunctionObject	51
generateRequest	52
Hse	53
HseSet	54
IDNumber	56
Math::KeepNSmallest	57
Math::KeepWSmallest	58
matrix	59
MultiChannelResult	62
Problem	63
Solution	64

5 CG_WP_1_3_2-SLEUTH Class Index

5.1 CG_WP_1_3_2-SLEUTH Class List

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

Anomalousness	31
BkgEvent	33
BkgSet	35
BkgSubSet	39
DataEvent	41
DataSet	43
EmptyEventSet	46
Event	47
EventSet< EventType >	49
Math::FunctionObject	51
generateRequest	52
Hse	53
HseSet	54
IDNumber	56
Math::KeepNSmallest	57
Math::KeepWSmallest	58
matrix	59
MultiChannelResult	62
Problem	63
Solution	64

6 CG_WP_1_3_2-SLEUTH File Index

6.1 CG_WP_1_3_2-SLEUTH File List

Here is a list of all files with brief descriptions:

addTwiddle.cpp	66
Anomalousness.cpp	67
Anomalousness.hpp	68
BkgEvent.cpp	69
BkgEvent.hpp	70
BkgSet.cpp	71
BkgSet.hpp	72
BkgSubSet.cpp	73
BkgSubSet.hpp	74
calc.cpp	75
combineScriptPs.cpp	76
DataEvent.cpp	78
DataEvent.hpp	79
DataSet.cpp	80
DataSet.hpp	81
EmptyEventSet.hpp	82
Event.cpp	83
Event.hpp	84
EventSet.hpp	85
GlobalVariables.cpp	86
GlobalVariables.hpp	87
grid.cpp	88
grid.hpp	90
gSleuth-ocmg.cpp	92
gSleuth.cpp	93
Hse.cpp	94
Hse.hpp	95
HseSet.cpp	96
HseSet.hpp	97
IDNumber.cpp	98
IDNumber.hpp	99
IOUtilities.cpp	100
IOUtilities.hpp	101
jacobi.cpp	102
lubksb.cpp	103
ludcmp.cpp	104
Math.cpp	105
Math.hpp	106
matrix.cpp	107
matrix.hpp	108
MultiChannelResult.cpp	109
MultiChannelResult.hpp	110
nrutil.cpp	111

nrutil.hpp	115
Problem.hpp	119
RCPPParameters.cpp	120
RCPPParameters.hpp	121
simp1.cpp	122
simp2.cpp	123
simp3.cpp	124
simplx.cpp	125
sleuth.cpp	126
sleuth.hpp	127
Solution.cpp	128
Solution.hpp	129
zbrent.cpp	130

7 CG_WP_1_3_2-SLEUTH Namespace Documentation

7.1 GlobalVariables Namespace Reference

Variables

- int `startTime` = 0
- int `maximumRegionSize` = 50
- `std::string idString1` = ""

7.1.1 Variable Documentation

7.1.1.1 `std::string GlobalVariables::idString1` = ""

7.1.1.2 `int GlobalVariables::maximumRegionSize` = 50

7.1.1.3 `int GlobalVariables::startTime` = 0

7.2 IOUtilities Namespace Reference

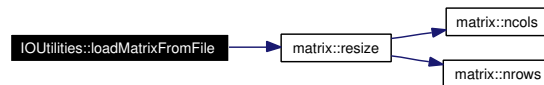
Functions

- `template<class T> void print (std::vector< T > vec, std::ostream &fout=std::cout)`
- `void loadMatrixFromFile (std::string filename, matrix &mat)`

7.2.1 Function Documentation

7.2.1.1 `void IOUtilities::loadMatrixFromFile (std::string filename, matrix &mat)`

Here is the call graph for this function:



7.2.1.2 `template<class T> void print (std::vector< T > vec, std::ostream &fout = std::cout)`

7.3 Math Namespace Reference

Classes

- class [KeepNSmallest](#)
- class [KeepWSmallest](#)
- class [FunctionObject](#)

Functions

- double [poissonConvolutedExceptionWithPoissonIntegrated](#) (double b, double wtMC, int N)
- double [poissonConvolutedExceptionWithGaussianIntegrated](#) (double b, double deltab, int N)
- double [probOfThisEffect](#) (double b, int N, double deltab=0, std::string opt=">=")
- double [roughMagnitudeOfDiscrepancy](#) (int N, double b, double deltab)
- double [bkgFromEffect](#) (double effect, int N)
- double [sigma2prob](#) (double s)
- double [prob2sigma](#) (double p)
- bool [MPEquality](#) (double a, double b, double tol=1.e-6)
- bool [MPEquality](#) (const std::vector< double > &a, const std::vector< double > &b, double tol=1.e-6)
- double [gasdev](#) (double mu, double sigma)
- std::vector< double > [randMultiGauss](#) (std::vector< double > mu, [matrix](#) &sigma)
- std::vector< double > [randMultiGauss](#) (std::vector< double > mu, std::vector< std::vector< double > > sigma)
- double [expdev](#) (double lambda)
- int [poisson](#) (double mu)
- int [fluctuate](#) (double mean, double systematicError)
- double [computeSum](#) (const std::vector< double > &x)
- double [computeAverage](#) (const std::vector< double > &x)
- long double [computeAverage](#) (std::vector< long double > x)
- double [computeAverage](#) (std::vector< int > x)
- double [computeRMS](#) (std::vector< double > x)
- long double [computeRMS](#) (std::vector< long double > x)
- double [effectiveNumberOfEvents](#) (const std::vector< double > &wt)
- double [computeMCerror](#) (const std::vector< double > &wt, double epsilonWt=0)
- double [gamma](#) (double x)
- double [volumeOfUnitSphere](#) (int d)
- double [distanceBetweenPoints](#) (const std::vector< double > &a, const std::vector< double > &b)
- double [distanceBetweenPoints](#) (const std::vector< double > &a, const std::vector< double > &b, [matrix](#) &covarianceMatrixInv)
- double [distanceSqdBetweenPoints](#) (const double *a, const double *b, int n)
- double [min](#) (double a, double b)
- double [max](#) (double a, double b)
- std::string [ftoa](#) (double x)
- std::vector< int > [getDigits](#) (int n, int base, int size=0)
- std::vector< int > [getDigits](#) (int n, std::vector< int > base, int size=0)
- std::vector< int > [integerNthRoot](#) (int a, int n)
- std::vector< std::vector< int > > [permutationSet](#) (std::vector< int > q)

- `std::vector< std::vector< int > >` [permutationSet](#) (int n)
- double [toleranceRound](#) (double x, double tol=1)
- double [sigFigRound](#) (double x, int nSigFigs=2)
- double [nice](#) (double x, int addedprecision=0)
- `std::vector< double >` [linearEquationSolve](#) ([matrix](#) A, `std::vector< double >` B)
- double [calculateSmoothingParameter](#) (double fractionalWeight, int nvars)
- double [eta2theta](#) (double eta)
- double [theta2eta](#) (double theta)
- double [sgnpow](#) (double x, double n)
- void [tossAwayTail](#) (`std::vector< std::vector< double > >` &events, double alpha=0)
- `std::vector< double >` [computeMedian](#) (`std::vector< std::vector< double > >` events)
- `std::vector< std::vector< double > >` [computeCorrelationMatrix](#) (`std::vector< std::vector< double > >` events, double alpha, int lNorm=2)
- `std::vector< std::string >` [vectorizeString](#) (std::string s, std::string separator)
- void [loadMatrixFromFile](#) (std::string filename, `std::vector< std::vector< double > >` &events)
- void [loadMatrixFromFile](#) (std::string filename, `std::vector< std::vector< double > >` &events, `std::vector< double >` &weights)
- double [minimize](#) (`std::vector< double >` &x, [FunctionObject](#) *funk, `std::vector< double >` dx=`std::vector< double >`(0), double tol=1e-2)
- double [amotry](#) ([matrix](#) &p, `std::vector< double >` &y, `std::vector< double >` &psum, int ndim, [FunctionObject](#) *funk, int ihi, double fac)
- void [amoeba](#) ([matrix](#) &p, `std::vector< double >` &y, int ndim, [FunctionObject](#) *funk, double ftol=1e-6, int NMAX=10000)
- double [binomialError](#) (double p, int N)
- double [deltaR](#) (double phi1, double eta1, double phi2, double eta2)
- void [makeNiceHistogramRange](#) (`std::vector< double >` a, int &nbins, double &lo, double &hi)
- void [makeNiceHistogramRange](#) (const `std::vector< double >` &bkgWeights, const `std::vector< double >` &sigWeights, const `std::vector< std::vector< double > >` &bkgEvents, const `std::vector< std::vector< double > >` &sigEvents, const `std::vector< std::vector< double > >` &dataEvents, `std::vector< std::vector< double > >` &range, `std::vector< int >` &nbins)
- `std::vector< double >` [putIntoBins](#) (`std::vector< double >` binEdges, `std::vector< double >` points, `std::vector< double >` wt)
- `std::vector< int >` [putIntoBins](#) (`std::vector< double >` binEdges, `std::vector< double >` points)
- `template<class T1, class T2>` void [parallelBubbleSort](#) (`std::vector< T1 >` &x, `std::vector< T2 >` &y)
- `template<class T1, class T2>` void [parallelQuickSort](#) (`std::vector< T1 >` &x, `std::vector< T2 >` &y)
- `std::vector< std::string >` [getFilesInDirectory](#) (std::string dir, std::string pattern="*")
- int [intpow](#) (int i, int j)

7.3.1 Function Documentation

7.3.1.1 void Math::amoeba ([matrix](#) & p, `std::vector< double >` & y, int ndim, [FunctionObject](#) * funk, double ftol = 1e-6, int NMAX = 10000)

Here is the call graph for this function:

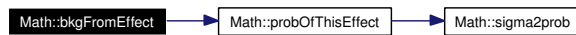


7.3.1.2 `double Math::amotry (matrix & p, std::vector< double > & y, std::vector< double > & psum, int ndim, FunctionObject * funk, int ihi, double fac)`

7.3.1.3 `double Math::binomialError (double p, int N)`

7.3.1.4 `double Math::bkgFromEffect (double effect, int N)`

Here is the call graph for this function:



7.3.1.5 `double Math::calculateSmoothingParameter (double fractionalWeight, int nvars)`

7.3.1.6 `double Math::computeAverage (std::vector< int > x)`

7.3.1.7 `long double Math::computeAverage (std::vector< long double > x)`

7.3.1.8 `double Math::computeAverage (const std::vector< double > & x)`

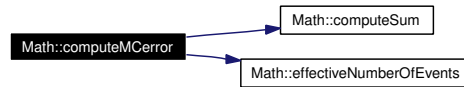
Here is the call graph for this function:



7.3.1.9 `std::vector<std::vector<double> > computeCorrelationMatrix (std::vector< std::vector< double > > events, double alpha, int lNorm = 2)`

7.3.1.10 `double Math::computeMCErrror (const std::vector< double > & wt, double epsilon Wt = 0)`

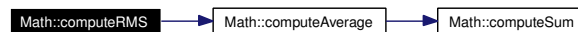
Here is the call graph for this function:



7.3.1.11 `std::vector<double> computeMedian (std::vector< std::vector< double > > events)`

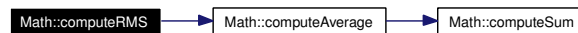
7.3.1.12 `long double Math::computeRMS (std::vector< long double > x)`

Here is the call graph for this function:



7.3.1.13 `double Math::computeRMS (std::vector< double > x)`

Here is the call graph for this function:



7.3.1.14 `double Math::computeSum (const std::vector< double > & x)`

7.3.1.15 `double Math::deltaR (double phi1, double eta1, double phi2, double eta2)`

7.3.1.16 `double distanceBetweenPoints (const std::vector< double > & a, const std::vector< double > & b, matrix & covarianceMatrixInv) [inline]`

7.3.1.17 `double distanceBetweenPoints (const std::vector< double > & a, const std::vector< double > & b) [inline]`

7.3.1.18 `double distanceSqdBetweenPoints (const double * a, const double * b, int n)` [inline]

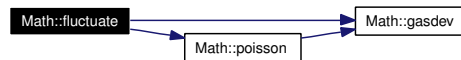
7.3.1.19 `double Math::effectiveNumberOfEvents (const std::vector< double > & wt)`

7.3.1.20 `double Math::eta2theta (double eta)`

7.3.1.21 `double Math::expdev (double lambda)`

7.3.1.22 `int Math::fluctuate (double mean, double systematicError)`

Here is the call graph for this function:



7.3.1.23 `string Math::ftoa (double x)`

7.3.1.24 `double Math::gamma (double x)`

7.3.1.25 `double Math::gasdev (double mu, double sigma)`

7.3.1.26 `vector< int > Math::getDigits (int n, std::vector< int > base, int size = 0)`

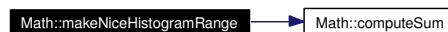
7.3.1.27 `vector< int > Math::getDigits (int n, int base, int size = 0)`

7.3.1.28 `std::vector< std::string > Math::getFilesInDirectory (std::string dir, std::string pattern = "*")`

Here is the call graph for this function:

**7.3.1.29** `vector< int > Math::integerNthRoot (int a, int n)`**7.3.1.30** `int Math::intpow (int i, int j)`**7.3.1.31** `std::vector<double> linearEquationSolve (matrix A, std::vector< double > B)`**7.3.1.32** `void Math::loadMatrixFromFile (std::string filename, std::vector< std::vector< double > > & events, std::vector< double > & weights)`**7.3.1.33** `void Math::loadMatrixFromFile (std::string filename, std::vector< std::vector< double > > & events)`**7.3.1.34** `void Math::makeNiceHistogramRange (const std::vector< double > & bkgWeights, const std::vector< double > & sigWeights, const std::vector< std::vector< double > > & bkgEvents, const std::vector< std::vector< double > > & sigEvents, const std::vector< std::vector< double > > & dataEvents, std::vector< std::vector< double > > & range, std::vector< int > & nbins)`

Here is the call graph for this function:



7.3.1.35 `void Math::makeNiceHistogramRange (std::vector< double > a, int & nbins, double & lo, double & hi)`

Here is the call graph for this function:

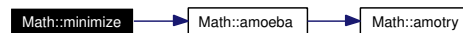


7.3.1.36 `double max (double a, double b) [inline]`

7.3.1.37 `double min (double a, double b) [inline]`

7.3.1.38 `double Math::minimize (std::vector< double > & x, FunctionObject * funk, std::vector< double > dx = std::vector< double >(0), double tol = 1e-2)`

Here is the call graph for this function:



7.3.1.39 `bool Math::MPEquality (const std::vector< double > & a, const std::vector< double > & b, double tol = 1.e-6)`

Here is the call graph for this function:



7.3.1.40 `bool Math::MPEquality (double a, double b, double tol = 1.e-6)`

7.3.1.41 `double Math::nice (double x, int addedprecision = 0)`

7.3.1.42 `template<class T1, class T2> void parallelBubbleSort (std::vector< T1 > & x, std::vector< T2 > & y)`

7.3.1.43 `template<class T1, class T2> void parallelQuickSort (std::vector< T1 > & x, std::vector< T2 > & y)`

7.3.1.44 `vector< vector< int > > Math::permutationSet (int n)`

Here is the call graph for this function:



7.3.1.45 `vector< vector< int > > Math::permutationSet (std::vector< int > q)`

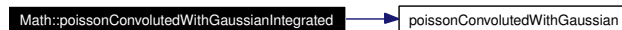
7.3.1.46 `int Math::poisson (double mu)`

Here is the call graph for this function:



7.3.1.47 `double Math::poissonConvolvedWithGaussianIntegrated (double b, double deltab, int N)`

Here is the call graph for this function:



7.3.1.48 `double Math::poissonConvolvedWithPoissonIntegrated (double b, double wtMC, int N)`

Here is the call graph for this function:



7.3.1.49 `double Math::prob2sigma (double p)`

Here is the call graph for this function:



7.3.1.50 `double Math::probOfThisEffect (double b , int N , double $deltab = 0$, std::string $opt = ">="$)`

Here is the call graph for this function:



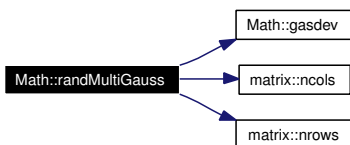
7.3.1.51 `vector< int > Math::putIntoBins (std::vector< double > $binEdges$, std::vector< double > $points$)`

7.3.1.52 `std::vector< double > Math::putIntoBins (std::vector< double > $binEdges$, std::vector< double > $points$, std::vector< double > wt)`

7.3.1.53 `std::vector<double> randMultiGauss (std::vector< double > mu , std::vector< std::vector< double > > $sigma$)`

7.3.1.54 `std::vector< double > Math::randMultiGauss (std::vector< double > mu , $matrix$ & $sigma$)`

Here is the call graph for this function:



7.3.1.55 `double Math::roughMagnitudeOfDiscrepancy (int N , double b , double $deltab$)`

7.3.1.56 `double Math::sgnpow (double x, double n)`

7.3.1.57 `double Math::sigFigRound (double x, int nSigFigs = 2)`

7.3.1.58 `double Math::sigma2prob (double s)`

7.3.1.59 `double Math::theta2eta (double theta)`

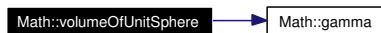
7.3.1.60 `double Math::toleranceRound (double x, double tol = 1)`

7.3.1.61 `void Math::tossAwayTail (std::vector< std::vector< double > > & events, double alpha = 0)`

7.3.1.62 `vector< string > Math::vectorizeString (std::string s, std::string separator)`

7.3.1.63 `double Math::volumeOfUnitSphere (int d)`

Here is the call graph for this function:



7.4 RCPPParameters Namespace Reference

Variables

- int `debugLevel` = 1
- `std::string` `hseLibraryDirectory` = `"/"`
- `std::string` `saveDirectory` = `"/"`
- bool `useBkgWeights` = true
- bool `useIDs` = true
- bool `useSystematicErrors` = true
- bool `useStatisticalErrors` = true
- `StoppingCriteriaType` `stoppingCriteriaType` = `scriptPabs`
- int `allottedTime` = `60*60`
- double `scriptPabsError` = `.10`
- double `scriptPfractionalError` = `.10`
- int `maxNumberHse` = 100

7.4.1 Variable Documentation

7.4.1.1 int `RCPPParameters::allottedTime` = `60*60`

7.4.1.2 int `RCPPParameters::debugLevel` = 1

7.4.1.3 `std::string` `RCPPParameters::hseLibraryDirectory` = `"/"`

7.4.1.4 int `RCPPParameters::maxNumberHse` = 100

7.4.1.5 `std::string` `RCPPParameters::saveDirectory` = `"/"`

7.4.1.6 double `RCPPParameters::scriptPabsError` = `.10`

7.4.1.7 **double RCPPParameters::scriptPfractionalError = .10**

7.4.1.8 **StoppingCriteriaType RCPPParameters::stoppingCriteriaType = scriptPabs**

7.4.1.9 **bool RCPPParameters::useBkgWeights = true**

7.4.1.10 **bool RCPPParameters::useIDs = true**

7.4.1.11 **bool RCPPParameters::useStatisticalErrors = true**

7.4.1.12 **bool RCPPParameters::useSystematicErrors = true**

7.5 std Namespace Reference

8 Class Documentation

8.1 Anomalousness Class Reference

```
#include <Anomalousness.hpp>
```

Public Member Functions

- [Anomalousness](#) ()
- [Anomalousness](#) (std::vector< double > _p_N, std::vector< double > _P_N, double _bigP, int _Nmin, double _scriptP, double _scriptPLo, double _scriptPHi)
- int [getNmin](#) ()
- double [getScriptP](#) ()
- double [getp_Nmin](#) ()
- void [print](#) (std::ostream &fout=std::cout)

8.1.1 Constructor & Destructor Documentation

8.1.1.1 [Anomalousness::Anomalousness](#) ()

8.1.1.2 [Anomalousness::Anomalousness](#) (std::vector< double > *_p_N*, std::vector< double > *_P_N*, double *_bigP*, int *_Nmin*, double *_scriptP*, double *_scriptPLo*, double *_scriptPHi*)

8.1.2 Member Function Documentation

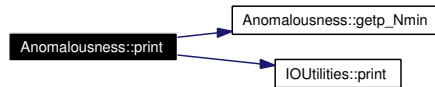
8.1.2.1 int [Anomalousness::getNmin](#) ()

8.1.2.2 double [Anomalousness::getp_Nmin](#) ()

8.1.2.3 double [Anomalousness::getScriptP](#) ()

8.1.2.4 void Anomalousness::print (std::ostream & *fout* = std::cout)

Here is the call graph for this function:



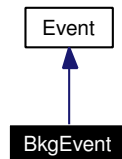
The documentation for this class was generated from the following files:

- [Anomalousness.hpp](#)
- [Anomalousness.cpp](#)

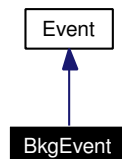
8.2 BkgEvent Class Reference

```
#include <BkgEvent.hpp>
```

Inheritance diagram for BkgEvent:



Collaboration diagram for BkgEvent:



Public Member Functions

- [BkgEvent](#) (std::vector< double > v)
- double [getWeight](#) ()
- void [normalizeWeightTo](#) (double bkgSubSetWeightSum)

8.2.1 Constructor & Destructor Documentation

8.2.1.1 BkgEvent::BkgEvent (std::vector< double > v)

Here is the call graph for this function:



8.2.2 Member Function Documentation

8.2.2.1 double BkgEvent::getWeight ()

8.2.2.2 void BkgEvent::normalizeWeightTo (double *bkgSubSetWeightSum*)

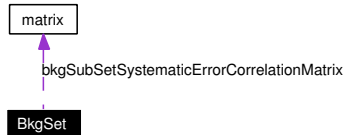
The documentation for this class was generated from the following files:

- [BkgEvent.hpp](#)
- [BkgEvent.cpp](#)

8.3 BkgSet Class Reference

```
#include <BkgSet.hpp>
```

Collaboration diagram for BkgSet:



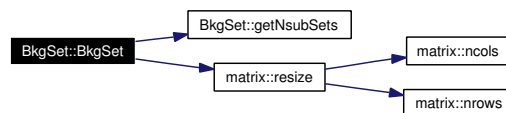
Public Member Functions

- [BkgSet](#) (std::string _bkgSetFileName, bool useMultipleBkgSubSets=false)
- [BkgSet](#) (const [BkgSet](#) &rhs)
- double [getTotalBkgExpected](#) () const
- double [getErrorOnTotalBkgExpected](#) ()
- std::vector< [BkgSubSet](#) * > [getBkgSubSets](#) ()
- [BkgEvent](#) * [getThisEvent](#) (int i)
- int [getNvars](#) () const
- int [getNevents](#) () const
- int [getNsubSets](#) ()
- void [print](#) (std::ostream &fout=std::cout)
- [BkgSubSet](#) * [getThisSubSet](#) (int i)
- std::string [getBkgSetFileName](#) ()
- [matrix](#) [generatePseudoEvents](#) (std::vector< int > nPseudoEvents=std::vector< int >(0))
- bool [containsContentlessSubSetsQ](#) ()
- double [getBkgUpAndToTheRightOf](#) (std::vector< double > v)
- std::string [getBkgCode](#) ()
- [matrix](#) * [getBkgSubSetSystematicErrorCorrelationMatrix](#) ()

8.3.1 Constructor & Destructor Documentation

8.3.1.1 [BkgSet::BkgSet](#) (std::string _bkgSetFileName, bool useMultipleBkgSubSets = false)

Here is the call graph for this function:



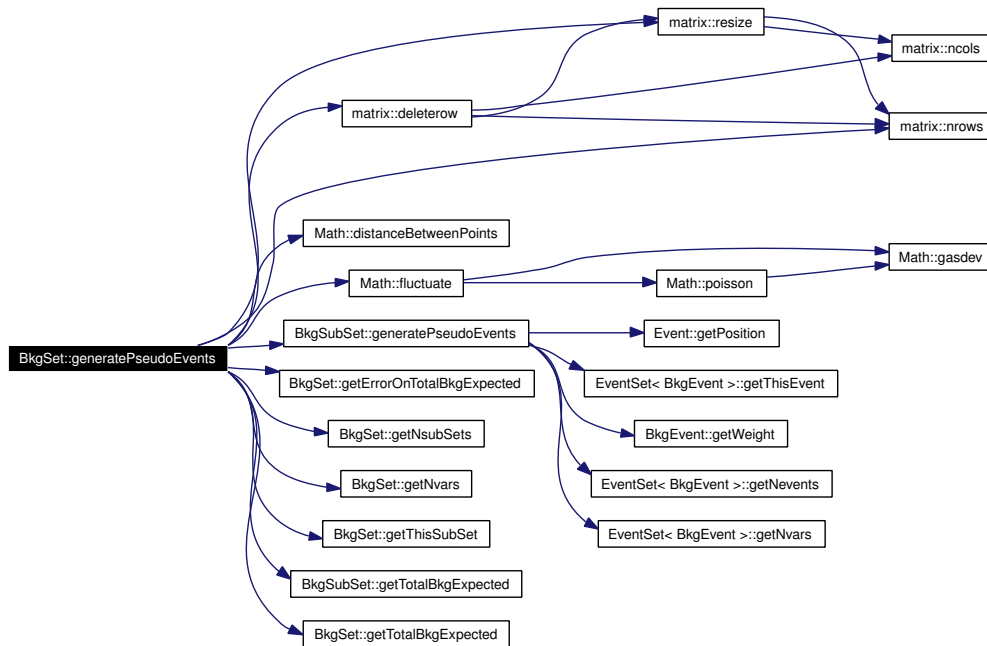
8.3.1.2 BkgSet::BkgSet (const BkgSet & rhs)

8.3.2 Member Function Documentation

8.3.2.1 bool BkgSet::containsContentlessSubSetsQ ()

8.3.2.2 matrix BkgSet::generatePseudoEvents (std::vector< int > nPseudoEvents = std::vector< int >(0))

Here is the call graph for this function:



8.3.2.3 string BkgSet::getBkgCode ()

Here is the call graph for this function:



8.3.2.4 string BkgSet::getBkgSetFileName ()

8.3.2.5 `vector< BkgSubSet * > BkgSet::getBkgSubSets ()`

8.3.2.6 `matrix * BkgSet::getBkgSubSetSystematicErrorCorrelationMatrix ()`

8.3.2.7 `double BkgSet::getBkgUpAndToTheRightOf (std::vector< double > v)`

Here is the call graph for this function:



8.3.2.8 `double BkgSet::getErrorOnTotalBkgExpected ()`

8.3.2.9 `int BkgSet::getNevents () const`

8.3.2.10 `int BkgSet::getNsubSets ()`

8.3.2.11 `int BkgSet::getNvars () const`

8.3.2.12 `BkgEvent * BkgSet::getThisEvent (int i)`

8.3.2.13 `BkgSubSet * BkgSet::getThisSubSet (int i)`

8.3.2.14 `double BkgSet::getTotalBkgExpected () const`

8.3.2.15 void BkgSet::print (std::ostream & *fout* = std::cout)

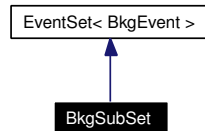
The documentation for this class was generated from the following files:

- [BkgSet.hpp](#)
- [BkgSet.cpp](#)

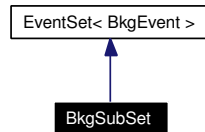
8.4 BkgSubSet Class Reference

```
#include <BkgSubSet.hpp>
```

Inheritance diagram for BkgSubSet:



Collaboration diagram for BkgSubSet:



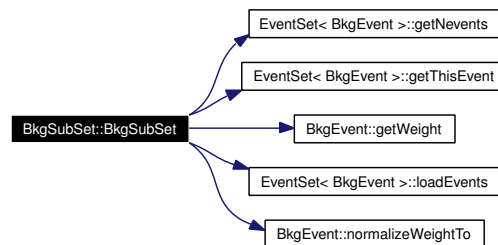
Public Member Functions

- [BkgSubSet](#) (double *_totalBkgExpected*, double *_errorOnTotalBkgExpected*, std::string *filename*)
- [BkgSubSet](#) (const [BkgSubSet](#) &rhs)
- double [getTotalBkgExpected](#) () const
- double [getErrorOnTotalBkgExpected](#) () const
- double [getBkgUpAndToTheRightOf](#) (std::vector< double > v)
- [matrix generatePseudoEvents](#) (int nPseudoEvents)

8.4.1 Constructor & Destructor Documentation

8.4.1.1 [BkgSubSet::BkgSubSet](#) (double *_totalBkgExpected*, double *_errorOnTotalBkgExpected*, std::string *filename*)

Here is the call graph for this function:

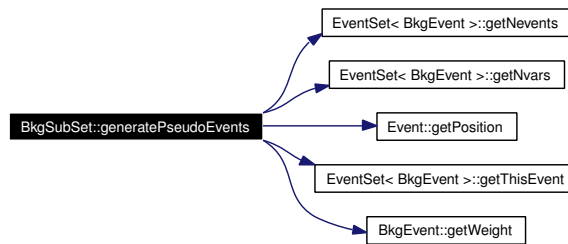


8.4.1.2 BkgSubSet::BkgSubSet (const BkgSubSet & rhs)

8.4.2 Member Function Documentation

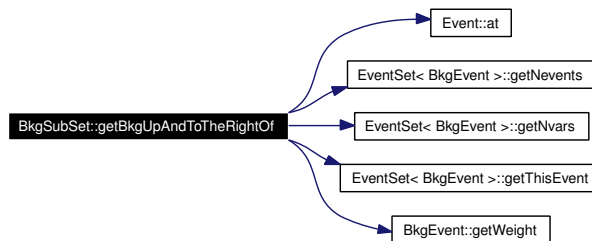
8.4.2.1 matrix BkgSubSet::generatePseudoEvents (int nPseudoEvents)

Here is the call graph for this function:



8.4.2.2 double BkgSubSet::getBkgUpAndToTheRightOf (std::vector< double > v)

Here is the call graph for this function:



8.4.2.3 double BkgSubSet::getErrorOnTotalBkgExpected () const

8.4.2.4 double BkgSubSet::getTotalBkgExpected () const

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

- [BkgSubSet.hpp](#)
- [BkgSubSet.cpp](#)

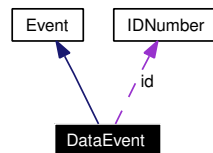
8.5 DataEvent Class Reference

```
#include <DataEvent.hpp>
```

Inheritance diagram for DataEvent:



Collaboration diagram for DataEvent:



Public Member Functions

- [DataEvent](#) (std::vector< double > v)
- [IDNumber](#) getId ()
- void [print](#) (std::ostream &fout=std::cout)

8.5.1 Constructor & Destructor Documentation

8.5.1.1 DataEvent::DataEvent (std::vector< double > v)

Here is the call graph for this function:



8.5.2 Member Function Documentation

8.5.2.1 IDNumber DataEvent::getId ()

8.5.2.2 void DataEvent::print (std::ostream & *fout* = std::cout)

Here is the call graph for this function:



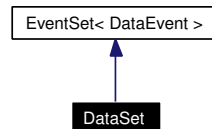
The documentation for this class was generated from the following files:

- [DataEvent.hpp](#)
- [DataEvent.cpp](#)

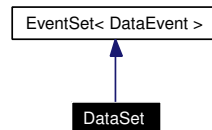
8.6 DataSet Class Reference

```
#include <DataSet.hpp>
```

Inheritance diagram for DataSet:



Collaboration diagram for DataSet:



Public Member Functions

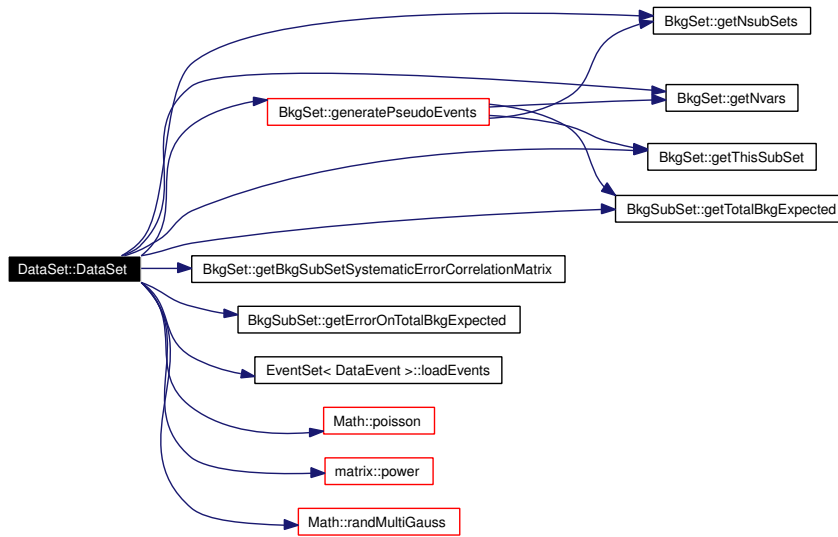
- [DataSet \(\)](#)
- [DataSet \(BkgSet *bkgSet\)](#)
- [DataSet \(matrix events\)](#)
- [DataSet \(std::string filename\)](#)
- [DataEvent * getThisEventById \(int id\)](#)

8.6.1 Constructor & Destructor Documentation

8.6.1.1 DataSet::DataSet ()

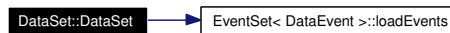
8.6.1.2 DataSet::DataSet (BkgSet * bkgSet)

Here is the call graph for this function:



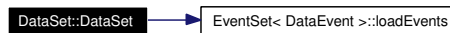
8.6.1.3 DataSet::DataSet (matrix events)

Here is the call graph for this function:



8.6.1.4 DataSet::DataSet (std::string filename)

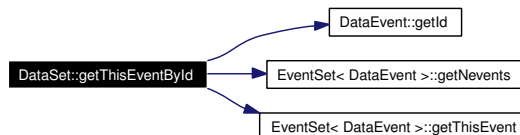
Here is the call graph for this function:



8.6.2 Member Function Documentation

8.6.2.1 DataEvent * DataSet::getThisEventById (int id)

Here is the call graph for this function:



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

- [DataSet.hpp](#)
- [DataSet.cpp](#)

8.7 EmptyEventSet Class Reference

```
#include <EmptyEventSet.hpp>
```

Public Member Functions

- [EmptyEventSet \(\)](#)

8.7.1 Constructor & Destructor Documentation

8.7.1.1 EmptyEventSet::EmptyEventSet () [inline]

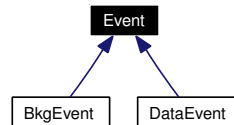
The documentation for this class was generated from the following file:

- [EmptyEventSet.hpp](#)

8.8 Event Class Reference

```
#include <Event.hpp>
```

Inheritance diagram for Event:



Public Member Functions

- `Event` (`std::vector< double > _position`)
- `Event & operator= (Event rhs)`
- `double at (int i) const`
- `bool operator== (Event &rhs)`
- `const std::vector< double > & getPosition () const`
- `int getNvars () const`
- `void print (std::ostream &fout=std::cout) const`

Protected Attributes

- `std::vector< double > position`

8.8.1 Constructor & Destructor Documentation

8.8.1.1 `Event::Event (std::vector< double > _position)`

8.8.2 Member Function Documentation

8.8.2.1 `double Event::at (int i) const`

8.8.2.2 `int Event::getNvars () const`

8.8.2.3 `const vector< double > & Event::getPosition () const`

8.8.2.4 `Event` & `Event::operator=` (`Event rhs`)

8.8.2.5 `bool Event::operator==` (`Event & rhs`)

8.8.2.6 `void Event::print` (`std::ostream & fout = std::cout`) `const`

Here is the call graph for this function:



8.8.3 Member Data Documentation

8.8.3.1 `std::vector<double> Event::position` [protected]

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

- [Event.hpp](#)
- [Event.cpp](#)

8.9 EventSet< EventType > Class Template Reference

```
#include <EventSet.hpp>
```

Public Member Functions

- [EventSet](#) ()
- void [loadEvents](#) (std::string filename)
- void [loadEvents](#) (matrix _events)
- EventType * [getThisEvent](#) (int i)
- bool [operator==](#) (EventSet< EventType > &rhs)
- int [getNvars](#) () const
- int [getNevents](#) () const
- void [print](#) (std::ostream &fout=std::cout)

```
template<class EventType> class EventSet< EventType >
```

8.9.1 Constructor & Destructor Documentation

8.9.1.1 `template<class EventType> EventSet< EventType >::EventSet ()`

8.9.2 Member Function Documentation

8.9.2.1 `template<class EventType> int EventSet< EventType >::getNevents () const`

8.9.2.2 `template<class EventType> int EventSet< EventType >::getNvars () const`

8.9.2.3 `template<class EventType> EventType* EventSet< EventType >::getThisEvent (int i)`

8.9.2.4 `template<class EventType> void EventSet< EventType >::loadEvents (matrix _events)`

8.9.2.5 `template<class EventType> void EventSet< EventType >::loadEvents (std::string filename)`

8.9.2.6 `template<class EventType> bool EventSet< EventType >::operator==(EventSet< EventType > & rhs)`

8.9.2.7 `template<class EventType> void EventSet< EventType >::print (std::ostream & fout = std::cout)`

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

- [EventSet.hpp](#)

8.10 Math::FunctionObject Class Reference

```
#include <Math.hpp>
```

Public Member Functions

- [FunctionObject](#) ()
- virtual double [operator](#)() (const std::vector< double > &x)

8.10.1 Constructor & Destructor Documentation

8.10.1.1 `Math::FunctionObject::FunctionObject ()` [inline]

8.10.2 Member Function Documentation

8.10.2.1 `virtual double Math::FunctionObject::operator() (const std::vector< double > &x)`
[inline, virtual]

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

- [Math.hpp](#)

8.11 generateRequest Struct Reference

```
#include <grid.hpp>
```

Public Attributes

- int [n](#)
- long [tag](#)

8.11.1 Member Data Documentation

8.11.1.1 int [generateRequest::n](#)

8.11.1.2 long [generateRequest::tag](#)

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

- [grid.hpp](#)

8.12 Hse Class Reference

```
#include <Hse.hpp>
```

Public Member Functions

- [Hse](#) (std::vector< double > _effects=std::vector< double >(0))
- std::vector< double > * [getEffects](#) ()
- double [isDataMoreAnomalous](#) (double effect, int i)

8.12.1 Constructor & Destructor Documentation

8.12.1.1 [Hse::Hse](#) (std::vector< double > _effects = std::vector< double >(0))

8.12.2 Member Function Documentation

8.12.2.1 [vector< double > * Hse::getEffects](#) ()

8.12.2.2 [double Hse::isDataMoreAnomalous](#) (double *effect*, int *i*) [inline]

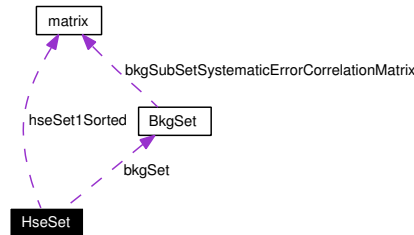
The documentation for this class was generated from the following files:

- [Hse.hpp](#)
- [Hse.cpp](#)

8.13 HseSet Class Reference

```
#include <HseSet.hpp>
```

Collaboration diagram for HseSet:



Public Member Functions

- [HseSet](#) ([BkgSet](#) *bkgSet)
- [Anomalousness determineAnomalousness](#) (std::vector< double > p_N)
- void [generateMore](#) (int n=100)

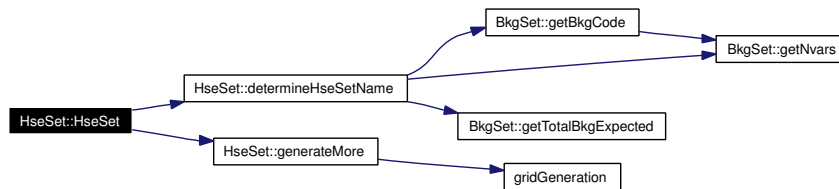
Static Public Member Functions

- std::string [determineHseSetName](#) ([BkgSet](#) *bkgSet)

8.13.1 Constructor & Destructor Documentation

8.13.1.1 HseSet::HseSet ([BkgSet](#) * bkgSet)

Here is the call graph for this function:



8.13.2 Member Function Documentation

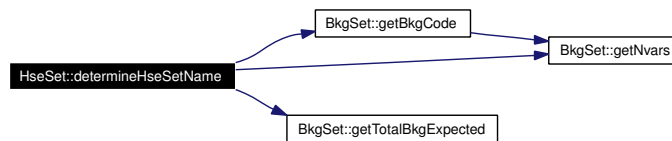
8.13.2.1 Anomalousness `HseSet::determineAnomalousness (std::vector< double > p_N)`

Here is the call graph for this function:



8.13.2.2 `string HseSet::determineHseSetName (BkgSet * bkgSet) [static]`

Here is the call graph for this function:



8.13.2.3 `void HseSet::generateMore (int n = 100)`

Here is the call graph for this function:



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

- [HseSet.hpp](#)
- [HseSet.cpp](#)

8.14 IDNumber Class Reference

```
#include <IDNumber.hpp>
```

Public Member Functions

- [IDNumber](#) (int *_id*)
- bool [operator==](#) (const [IDNumber](#) &*rhs*)

Friends

- std::ostream & [operator<<](#) (std::ostream &*os*, const [IDNumber](#) &*rhs*)

8.14.1 Constructor & Destructor Documentation

8.14.1.1 [IDNumber::IDNumber](#) (int *_id*)

8.14.2 Member Function Documentation

8.14.2.1 bool [IDNumber::operator==](#) (const [IDNumber](#) & *rhs*)

8.14.3 Friends And Related Function Documentation

8.14.3.1 std::ostream& [operator<<](#) (std::ostream & *os*, const [IDNumber](#) & *rhs*) [friend]

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

- [IDNumber.hpp](#)
- [IDNumber.cpp](#)

8.15 Math::KeepNSmallest Class Reference

```
#include <Math.hpp>
```

Public Member Functions

- [KeepNSmallest](#) (int N)
- void [update](#) (double value, int number)
- std::vector< int > [getNumbers](#) ()
- std::vector< double > [getValues](#) ()

8.15.1 Constructor & Destructor Documentation

8.15.1.1 [Math::KeepNSmallest::KeepNSmallest](#) (int N)

8.15.2 Member Function Documentation

8.15.2.1 [std::vector<int> Math::KeepNSmallest::getNumbers](#) ()

8.15.2.2 [std::vector<double> Math::KeepNSmallest::getValues](#) ()

8.15.2.3 [void Math::KeepNSmallest::update](#) (double *value*, int *number*)

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

- [Math.hpp](#)

8.16 Math::KeepWSmallest Class Reference

```
#include <Math.hpp>
```

Public Member Functions

- [KeepWSmallest](#) (double W, int N=0)
- void [update](#) (double value, int number, double weight=1)
- std::vector< int > [getNumbers](#) ()
- std::vector< double > [getValues](#) ()

8.16.1 Constructor & Destructor Documentation

8.16.1.1 `Math::KeepWSmallest::KeepWSmallest (double W, int N = 0)`

8.16.2 Member Function Documentation

8.16.2.1 `std::vector<int> Math::KeepWSmallest::getNumbers ()`

8.16.2.2 `std::vector<double> Math::KeepWSmallest::getValues ()`

8.16.2.3 `void Math::KeepWSmallest::update (double value, int number, double weight = 1)`

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

- [Math.hpp](#)

8.17 matrix Class Reference

```
#include <matrix.hpp>
```

Public Member Functions

- [matrix](#) (int n=1, int m=1)
- void [resize](#) (int n, int m)
- int [nrows](#) () const
- int [ncols](#) () const
- std::vector< double > & [operator\[\]](#) (int row)
- [matrix](#) & [operator=](#) ([matrix](#) rhs)
- void [deletecolumn](#) (int n)
- void [deleterow](#) (int n)
- void [print](#) ()
- [matrix power](#) (double n)
- [matrix safeInverse](#) (double &determinant)
- double [det](#) ()

8.17.1 Constructor & Destructor Documentation

8.17.1.1 [matrix::matrix](#) (int $n = 1$, int $m = 1$)

8.17.2 Member Function Documentation

8.17.2.1 void [matrix::deletecolumn](#) (int n)

Here is the call graph for this function:



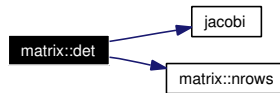
8.17.2.2 void matrix::deleterow (int *n*)

Here is the call graph for this function:



8.17.2.3 double matrix::det ()

Here is the call graph for this function:



8.17.2.4 int matrix::ncols () const

8.17.2.5 int matrix::nrows () const

8.17.2.6 matrix & matrix::operator= (matrix *rhs*)

Here is the call graph for this function:

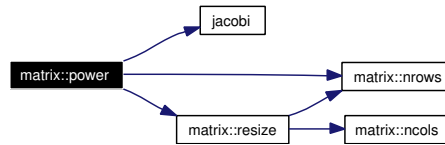


8.17.2.7]

vector< double > & matrix::operator[] (int *row*)

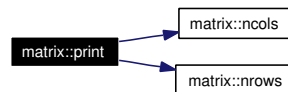
8.17.2.8 `matrix` `matrix::power (double n)`

Here is the call graph for this function:



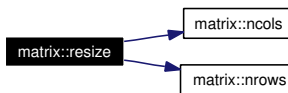
8.17.2.9 `void matrix::print ()`

Here is the call graph for this function:



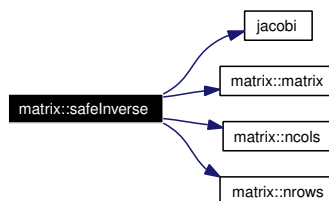
8.17.2.10 `void matrix::resize (int n, int m)`

Here is the call graph for this function:



8.17.2.11 `matrix` `matrix::safeInverse (double & determinant)`

Here is the call graph for this function:



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

- [matrix.hpp](#)
- [matrix.cpp](#)

8.18 MultiChannelResult Class Reference

```
#include <MultiChannelResult.hpp>
```

Public Member Functions

- [MultiChannelResult](#) (std::string fileName)
- double [getGothicP](#) ()

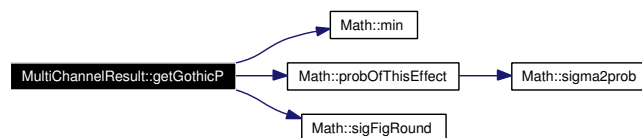
8.18.1 Constructor & Destructor Documentation

8.18.1.1 MultiChannelResult::MultiChannelResult (std::string *fileName*)

8.18.2 Member Function Documentation

8.18.2.1 double MultiChannelResult::getGothicP ()

Here is the call graph for this function:



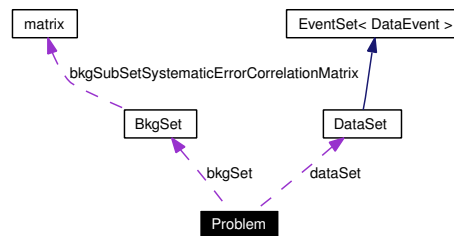
The documentation for this class was generated from the following files:

- [MultiChannelResult.hpp](#)
- [MultiChannelResult.cpp](#)

8.19 Problem Class Reference

```
#include <Problem.hpp>
```

Collaboration diagram for Problem:



Public Member Functions

- [Problem](#) ([DataSet](#) * _dataSet, [BkgSet](#) * _bkgSet, [RegionCriteria](#) _regionCriteria)
- [DataSet](#) * [getDataSet](#) () const
- [BkgSet](#) * [getBkgSet](#) () const
- [RegionCriteria](#) [getRegionCriteria](#) () const

8.19.1 Constructor & Destructor Documentation

8.19.1.1 [Problem::Problem](#) ([DataSet](#) * _dataSet, [BkgSet](#) * _bkgSet, [RegionCriteria](#) _regionCriteria)

8.19.2 Member Function Documentation

8.19.2.1 [BkgSet](#)* [Problem::getBkgSet](#) () const

8.19.2.2 [DataSet](#)* [Problem::getDataSet](#) () const

8.19.2.3 [RegionCriteria](#) [Problem::getRegionCriteria](#) () const

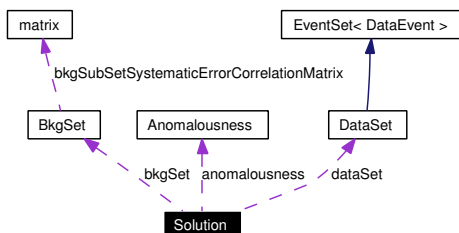
The documentation for this class was generated from the following file:

- [Problem.hpp](#)

8.20 Solution Class Reference

```
#include <Solution.hpp>
```

Collaboration diagram for Solution:



Public Member Functions

- [Solution](#) ([DataSet](#) *dataSet, [BkgSet](#) *bkgSet)
- void [solve](#) ()
- void [determineAnomalousness](#) ([HseSet](#) *hseSet)
- void [show](#) ()
- void [hseToFile](#) (std::ofstream &fhse)
- std::vector< double > [getp_N](#) ()

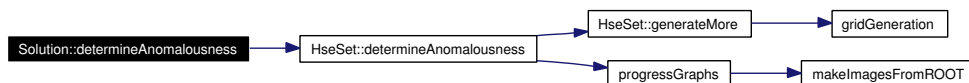
8.20.1 Constructor & Destructor Documentation

8.20.1.1 [Solution::Solution](#) ([DataSet](#) * *dataSet*, [BkgSet](#) * *bkgSet*)

8.20.2 Member Function Documentation

8.20.2.1 void [Solution::determineAnomalousness](#) ([HseSet](#) * *hseSet*)

Here is the call graph for this function:



8.20.2.2 [vector< double > Solution::getp_N](#) ()

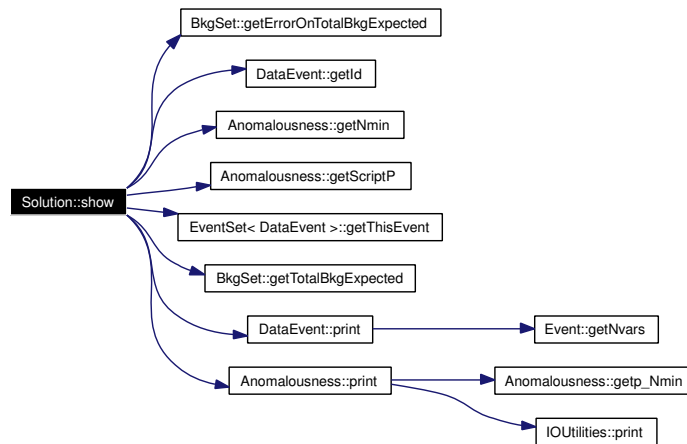
8.20.2.3 void Solution::hseToFile (std::ofstream & fhse)

Here is the call graph for this function:



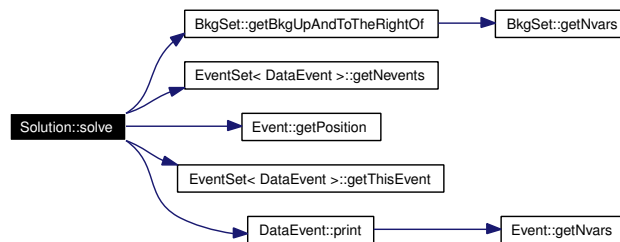
8.20.2.4 void Solution::show ()

Here is the call graph for this function:



8.20.2.5 void Solution::solve ()

Here is the call graph for this function:



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

- [Solution.hpp](#)
- [Solution.cpp](#)

9 CG_WP_1_3_2-SLEUTH File Documentation

9.1 addTwiddle.cpp File Reference

Namespaces

- namespace `std`

Functions

- int `main` ()

Variables

- const string `scriptPsFile` = "scriptPs.txt"

9.1.1 Function Documentation

9.1.1.1 int main ()

Here is the call graph for this function:



9.1.2 Variable Documentation

9.1.2.1 const string scriptPsFile = "scriptPs.txt"

9.2 Anomalousness.cpp File Reference

9.3 Anomalousness.hpp File Reference

Classes

- class [Anomalousness](#)

9.4 BkgEvent.cpp File Reference

9.5 BkgEvent.hpp File Reference

Classes

- class [BkgEvent](#)

9.6 BkgSet.cpp File Reference

9.7 BkgSet.hpp File Reference

Classes

- class [BkgSet](#)

9.8 BkgSubSet.cpp File Reference

Functions

- double [derf_](#) (double &)

9.8.1 Function Documentation

9.8.1.1 double [derf_](#) (double &)

9.9 BkgSubSet.hpp File Reference

Classes

- class [BkgSubSet](#)

9.10 calc.cpp File Reference

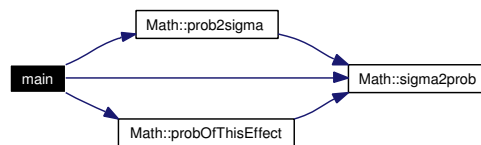
Functions

- int `main` (int argc, char *argv[])

9.10.1 Function Documentation

9.10.1.1 int `main` (int *argc*, char * *argv*[])

Here is the call graph for this function:



9.11 combineScriptPs.cpp File Reference

Functions

- void `execute` (string *command*)
- void `fixprecision` (ostream & *fout*, double *x*)
- void `combineScriptPs` (string *parentDirectory*, string *childDirectory*)
- int `main` ()

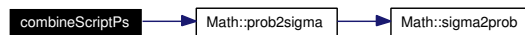
Variables

- ofstream `fex`

9.11.1 Function Documentation

9.11.1.1 void `combineScriptPs` (string *parentDirectory*, string *childDirectory*)

Here is the call graph for this function:

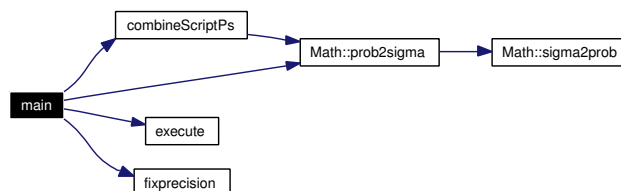


9.11.1.2 void `execute` (string *command*)

9.11.1.3 void `fixprecision` (ostream & *fout*, double *x*)

9.11.1.4 int `main` ()

Here is the call graph for this function:



9.11.2 Variable Documentation

9.11.2.1 ofstream [ftex](#)

9.12 DataEvent.cpp File Reference

9.13 DataEvent.hpp File Reference

Classes

- class [DataEvent](#)

9.14 DataSet.cpp File Reference

9.15 DataSet.hpp File Reference

Classes

- class [DataSet](#)

9.16 EmptyEventSet.hpp File Reference

Classes

- class [EmptyEventSet](#)

9.17 Event.cpp File Reference

9.18 Event.hpp File Reference

Classes

- class [Event](#)

9.19 EventSet.hpp File Reference

Classes

- class [EventSet](#)

9.20 GlobalVariables.cpp File Reference

Namespaces

- namespace [GlobalVariables](#)

9.21 GlobalVariables.hpp File Reference

Namespaces

- namespace [GlobalVariables](#)

9.22 grid.cpp File Reference

Functions

- void [makeImagesFromROOT](#) (TCanvas *c1, string name)
- void [progressGraphs](#) (int n, double *dynTimeVector, double *dynHSEVector, double *dynScriptPVector, double *dynScriptPHiVector, double *dynScriptPVector)
- int [SleuthMpiVisualize](#) (BkgSet *bkgSet, DataSet *dataSet)
- void [SleuthMpiEnd](#) (int np)
- void [gridGeneration](#) (int n, vector< Hse > &hseSet1, vector< Hse > &hseSet2)

9.22.1 Function Documentation

9.22.1.1 void gridGeneration (int n, vector< Hse > & hseSet1, vector< Hse > & hseSet2)

9.22.1.2 void makeImagesFromROOT (TCanvas * c1, string name)

9.22.1.3 void progressGraphs (int n, double * dynTimeVector, double * dynHSEVector, double * dynScriptPVector, double * dynScriptPHiVector, double * dynScriptPVector)

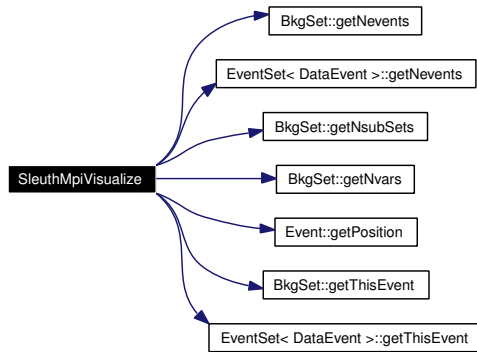
Here is the call graph for this function:



9.22.1.4 void SleuthMpiEnd (int np)

9.22.1.5 int SleuthMpiVisualize (BkgSet * bkgSet, DataSet * dataSet)

Here is the call graph for this function:



9.23 grid.hpp File Reference

Classes

- struct [generateRequest](#)

Functions

- void [progressGraphs](#) (int, double *, double *, double *, double *, double *)
- int [SleuthMpiVisualize](#) ([BkgSet](#) *bkgSet, [DataSet](#) *dataSet)
- void [SleuthMpiEnd](#) (int np)
- void [gridGeneration](#) (int, vector< [Hse](#) > &, vector< [Hse](#) > &)

9.23.1 Function Documentation

9.23.1.1 void gridGeneration (int, vector< [Hse](#) > &, vector< [Hse](#) > &)

9.23.1.2 void progressGraphs (int, double *, double *, double *, double *, double *)

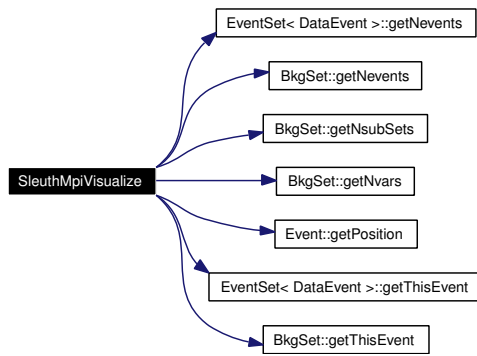
Here is the call graph for this function:



9.23.1.3 void SleuthMpiEnd (int np)

9.23.1.4 int SleuthMpiVisualize ([BkgSet](#) * bkgSet, [DataSet](#) * dataSet)

Here is the call graph for this function:



9.24 gSleuth-ocmg.cpp File Reference

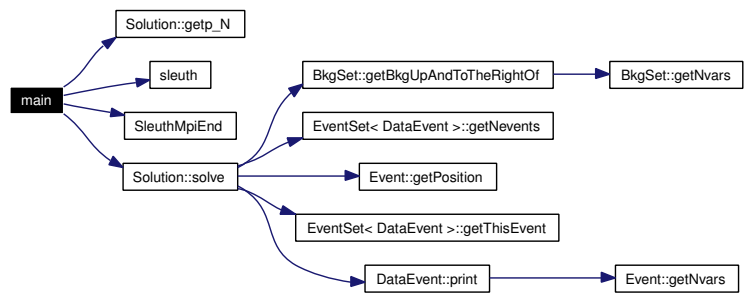
Functions

- int `main` (int argc, char *argv[])

9.24.1 Function Documentation

9.24.1.1 int main (int argc, char * argv[])

Here is the call graph for this function:



9.25 gSleuth.cpp File Reference

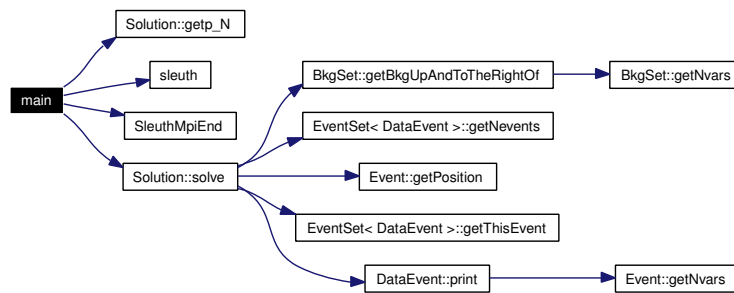
Functions

- int [main](#) (int argc, char *argv[])

9.25.1 Function Documentation

9.25.1.1 int main (int argc, char * argv[])

Here is the call graph for this function:



9.26 Hse.cpp File Reference

9.27 Hse.hpp File Reference

Classes

- class [Hse](#)

9.28 HseSet.cpp File Reference

9.29 HseSet.hpp File Reference

Classes

- class [HseSet](#)

9.30 IDNumber.cpp File Reference

Functions

- ostream & [operator<<](#) (ostream &os, const [IDNumber](#) &rhs)

9.30.1 Function Documentation

9.30.1.1 ostream& operator<< (std::ostream & os, const [IDNumber](#) & rhs)

9.31 IDNumber.hpp File Reference

Classes

- class [IDNumber](#)

9.32 IOUtilities.cpp File Reference

9.33 IOUtilities.hpp File Reference

Namespaces

- namespace [IOUtilities](#)

9.34 jacobi.cpp File Reference

Functions

- void `jacobi` (`matrix` &a, int `n`, vector< double > &d, `matrix` &v)

9.34.1 Function Documentation

9.34.1.1 void `jacobi` (`matrix` & *a*, int *n*, vector< double > & *d*, `matrix` & *v*)

9.35 lubksb.cpp File Reference

Functions

- void `lubksb` (float ***a*, int *n*, int **indx*, float *b*[])

9.35.1 Function Documentation

9.35.1.1 void `lubksb` (float ** *a*, int *n*, int * *indx*, float *b*[])

9.36 ludcmp.cpp File Reference

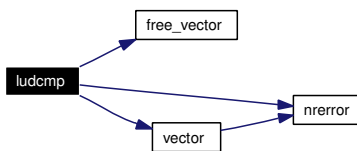
Functions

- void `ludcmp` (float **a, int n, int *indx, float *d)

9.36.1 Function Documentation

9.36.1.1 void ludcmp (float **a, int n, int *indx, float *d)

Here is the call graph for this function:



9.37 Math.cpp File Reference

Functions

- double [poissonConvolvedWithPoisson](#) (double x , void *params)
- double [poissonConvolvedWithGaussian](#) (double x , void *params)

9.37.1 Function Documentation

9.37.1.1 double `poissonConvolvedWithGaussian` (double x , void * *params*)

9.37.1.2 double `poissonConvolvedWithPoisson` (double x , void * *params*)

9.38 Math.hpp File Reference

Namespaces

- namespace [Math](#)

9.39 `matrix.cpp` File Reference

9.40 matrix.hpp File Reference

Classes

- class [matrix](#)

Functions

- void [jacobi](#) ([matrix](#) &a, int n, std::vector< double > &d, [matrix](#) &v)
- template<class T> void [print](#) (std::vector< T > vec, std::ostream &fout=std::cout)

9.40.1 Function Documentation

9.40.1.1 void [jacobi](#) ([matrix](#) & a, int n, std::vector< double > & d, [matrix](#) & v)

9.40.1.2 template<class T> void [print](#) (std::vector< T > vec, std::ostream & fout = std::cout)

9.41 MultiChannelResult.cpp File Reference

Functions

- double [derf_](#) (double &)

9.41.1 Function Documentation

9.41.1.1 double [derf_](#) (double &)

9.42 MultiChannelResult.hpp File Reference

Classes

- class [MultiChannelResult](#)

9.43 nrutil.cpp File Reference

Functions

- void `nrerror` (char error_text[])
- float * `vector` (long nl, long nh)
- int * `ivector` (long nl, long nh)
- unsigned char * `cvector` (long nl, long nh)
- unsigned long * `lvector` (long nl, long nh)
- double * `dvector` (long nl, long nh)
- float ** `matrix` (long nrl, long nrh, long ncl, long nch)
- double ** `dmatrix` (long nrl, long nrh, long ncl, long nch)
- int ** `imatrix` (long nrl, long nrh, long ncl, long nch)
- float ** `submatrix` (float **a, long oldrl, long oldrh, long oldcl, long oldch, long newrl, long newcl)
- float ** `convert_matrix` (float *a, long nrl, long nrh, long ncl, long nch)
- float *** `f3tensor` (long nrl, long nrh, long ncl, long nch, long ndl, long ndh)
- void `free_vector` (float *v, long nl, long nh)
- void `free_ivector` (int *v, long nl, long nh)
- void `free_cvector` (unsigned char *v, long nl, long nh)
- void `free_lvector` (unsigned long *v, long nl, long nh)
- void `free_dvector` (double *v, long nl, long nh)
- void `free_matrix` (float **m, long nrl, long nrh, long ncl, long nch)
- void `free_dmatrix` (double **m, long nrl, long nrh, long ncl, long nch)
- void `free_imatrix` (int **m, long nrl, long nrh, long ncl, long nch)
- void `free_submatrix` (float **b, long nrl, long nrh, long ncl, long nch)
- void `free_convert_matrix` (float **b, long nrl, long nrh, long ncl, long nch)
- void `free_f3tensor` (float ***t, long nrl, long nrh, long ncl, long nch, long ndl, long ndh)

9.43.1 Function Documentation

9.43.1.1 float** convert_matrix (float * a, long nrl, long nrh, long ncl, long nch)

Here is the call graph for this function:



9.43.1.2 unsigned char* cvector (long nl, long nh)

Here is the call graph for this function:



9.43.1.3 double dmatrix (long nrl, long nrh, long ncl, long nch)**

Here is the call graph for this function:

**9.43.1.4 double* dvector (long nl, long nh)**

Here is the call graph for this function:

**9.43.1.5 float*** f3tensor (long nrl, long nrh, long ncl, long nch, long ndl, long ndh)**

Here is the call graph for this function:

**9.43.1.6 void free_convert_matrix (float ** b, long nrl, long nrh, long ncl, long nch)****9.43.1.7 void free_cvector (unsigned char * v, long nl, long nh)****9.43.1.8 void free_dmatrix (double ** m, long nrl, long nrh, long ncl, long nch)****9.43.1.9 void free_dvector (double * v, long nl, long nh)****9.43.1.10 void free_f3tensor (float *** t, long nrl, long nrh, long ncl, long nch, long ndl, long ndh)**

9.43.1.11 void free_imatrix (int ** *m*, long *nrl*, long *nrh*, long *ncl*, long *nch*)

9.43.1.12 void free_ivector (int * *v*, long *nl*, long *nh*)

9.43.1.13 void free_lvector (unsigned long * *v*, long *nl*, long *nh*)

9.43.1.14 void free_matrix (float ** *m*, long *nrl*, long *nrh*, long *ncl*, long *nch*)

9.43.1.15 void free_submatrix (float ** *b*, long *nrl*, long *nrh*, long *ncl*, long *nch*)

9.43.1.16 void free_vector (float * *v*, long *nl*, long *nh*)

9.43.1.17 int** imatrix (long *nrl*, long *nrh*, long *ncl*, long *nch*)

Here is the call graph for this function:



9.43.1.18 int* ivector (long *nl*, long *nh*)

Here is the call graph for this function:



9.43.1.19 unsigned long* lvector (long *nl*, long *nh*)

Here is the call graph for this function:



9.43.1.20 float matrix (long nrl, long nrh, long ncl, long nch)**

Here is the call graph for this function:

**9.43.1.21 void nerror (char error_text[])****9.43.1.22 float** submatrix (float ** a, long oldrl, long oldrh, long oldcl, long oldch, long newrl, long newcl)**

Here is the call graph for this function:

**9.43.1.23 float* vector (long nl, long nh)**

Here is the call graph for this function:



9.44 nrutil.hpp File Reference

Functions

- void `nrerror` (char error_text[])
- float * `vector` (long nl, long nh)
- int * `ivector` (long nl, long nh)
- unsigned char * `cvector` (long nl, long nh)
- unsigned long * `lvector` (long nl, long nh)
- double * `dvector` (long nl, long nh)
- float ** `matrix` (long nrl, long nrh, long ncl, long nch)
- double ** `dmatrix` (long nrl, long nrh, long ncl, long nch)
- int ** `imatrix` (long nrl, long nrh, long ncl, long nch)
- float ** `submatrix` (float **a, long oldrl, long oldrh, long oldcl, long oldch, long newrl, long newcl)
- float ** `convert_matrix` (float *a, long nrl, long nrh, long ncl, long nch)
- float *** `f3tensor` (long nrl, long nrh, long ncl, long nch, long ndl, long ndh)
- void `free_vector` (float *v, long nl, long nh)
- void `free_ivector` (int *v, long nl, long nh)
- void `free_cvector` (unsigned char *v, long nl, long nh)
- void `free_lvector` (unsigned long *v, long nl, long nh)
- void `free_dvector` (double *v, long nl, long nh)
- void `free_matrix` (float **m, long nrl, long nrh, long ncl, long nch)
- void `free_dmatrix` (double **m, long nrl, long nrh, long ncl, long nch)
- void `free_imatrix` (int **m, long nrl, long nrh, long ncl, long nch)
- void `free_submatrix` (float **b, long nrl, long nrh, long ncl, long nch)
- void `free_convert_matrix` (float **b, long nrl, long nrh, long ncl, long nch)
- void `free_f3tensor` (float ***t, long nrl, long nrh, long ncl, long nch, long ndl, long ndh)

9.44.1 Function Documentation

9.44.1.1 float** convert_matrix (float * a, long nrl, long nrh, long ncl, long nch)

Here is the call graph for this function:



9.44.1.2 unsigned char* cvector (long nl, long nh)

Here is the call graph for this function:



9.44.1.3 double dmatrix (long nrl, long nrh, long ncl, long nch)**

Here is the call graph for this function:

**9.44.1.4 double* dvector (long nl, long nh)**

Here is the call graph for this function:

**9.44.1.5 float*** f3tensor (long nrl, long nrh, long ncl, long nch, long ndl, long ndh)**

Here is the call graph for this function:

**9.44.1.6 void free_convert_matrix (float ** b, long nrl, long nrh, long ncl, long nch)****9.44.1.7 void free_cvector (unsigned char * v, long nl, long nh)****9.44.1.8 void free_dmatrix (double ** m, long nrl, long nrh, long ncl, long nch)****9.44.1.9 void free_dvector (double * v, long nl, long nh)****9.44.1.10 void free_f3tensor (float *** t, long nrl, long nrh, long ncl, long nch, long ndl, long ndh)**

9.44.1.11 void free_imatrix (int ** *m*, long *nrl*, long *nrh*, long *ncl*, long *nch*)

9.44.1.12 void free_ivector (int * *v*, long *nl*, long *nh*)

9.44.1.13 void free_lvector (unsigned long * *v*, long *nl*, long *nh*)

9.44.1.14 void free_matrix (float ** *m*, long *nrl*, long *nrh*, long *ncl*, long *nch*)

9.44.1.15 void free_submatrix (float ** *b*, long *nrl*, long *nrh*, long *ncl*, long *nch*)

9.44.1.16 void free_vector (float * *v*, long *nl*, long *nh*)

9.44.1.17 int imatrix (long *nrl*, long *nrh*, long *ncl*, long *nch*)**

Here is the call graph for this function:



9.44.1.18 int* ivector (long *nl*, long *nh*)

Here is the call graph for this function:



9.44.1.19 unsigned long* lvector (long *nl*, long *nh*)

Here is the call graph for this function:



9.44.1.20 float matrix (long nrl, long nrh, long ncl, long nch)**

Here is the call graph for this function:

**9.44.1.21 void nerror (char error_text[])****9.44.1.22 float** submatrix (float ** a, long oldrl, long oldrh, long oldcl, long oldch, long newrl, long newcl)**

Here is the call graph for this function:

**9.44.1.23 float* vector (long nl, long nh)**

Here is the call graph for this function:



9.45 Problem.hpp File Reference

Classes

- class [Problem](#)

9.46 RCPPParameters.cpp File Reference

Namespaces

- namespace [RCPPParameters](#)

9.47 RCPPParameters.hpp File Reference

Namespaces

- namespace [RCPPParameters](#)

Enumerations

- enum [StoppingCriteriaType](#) { [timeConstraint](#), [scriptPabs](#), [scriptPfractional](#), [numberHse](#) }

9.47.1 Enumeration Type Documentation

9.47.1.1 enum [StoppingCriteriaType](#)

Enumeration values:

timeConstraint

scriptPabs

scriptPfractional

numberHse

9.48 simp1.cpp File Reference

Functions

- void [simp1](#) (double **a, int mm, int ll[], int nll, int iabf, int *kp, double *bmax)

9.48.1 Function Documentation

9.48.1.1 void `simp1` (double ** *a*, int *mm*, int *ll*[], int *nll*, int *iabf*, int * *kp*, double * *bmax*)

9.49 simp2.cpp File Reference

Functions

- void [simp2](#) (double **a, int n, int l2[], int nl2, int *ip, int kp, double *q1)

9.49.1 Function Documentation

9.49.1.1 void simp2 (double ** *a*, int *n*, int *l2*[], int *nl2*, int * *ip*, int *kp*, double * *q1*)

9.50 simp3.cpp File Reference

Functions

- void [simp3](#) (double **a, int i1, int k1, int ip, int kp)

9.50.1 Function Documentation

9.50.1.1 void `simp3` (double ** *a*, int *i1*, int *k1*, int *ip*, int *kp*)

9.51 `simplx.cpp` File Reference

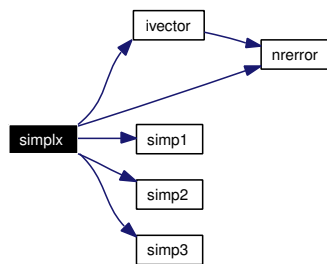
Functions

- void `simplx` (double ***a*, int *m*, int *n*, int *m1*, int *m2*, int *m3*, int **icase*, int *izrov*[], int *iposv*[])

9.51.1 Function Documentation

9.51.1.1 void `simplx` (double ***a*, int *m*, int *n*, int *m1*, int *m2*, int *m3*, int **icase*, int *izrov*[], int *iposv*[])

Here is the call graph for this function:



9.52 sleuth.cpp File Reference

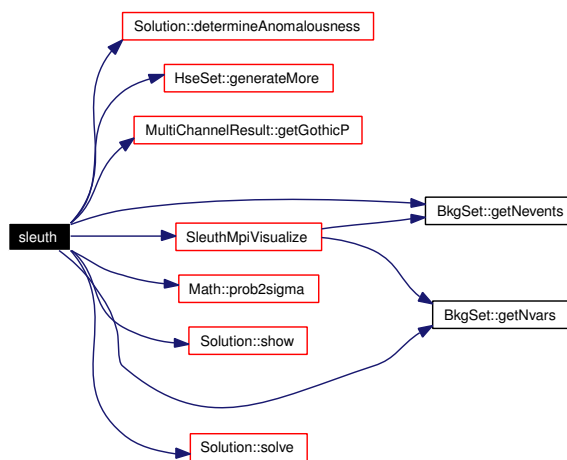
Functions

- int `sleuth` (int argc, char *argv[])

9.52.1 Function Documentation

9.52.1.1 int sleuth (int *argc*, char * *argv*[])

Here is the call graph for this function:



9.53 sleuth.hpp File Reference

Functions

- int [sleuth](#) (int, char **)

9.53.1 Function Documentation

9.53.1.1 int sleuth (int, char **)

9.54 Solution.cpp File Reference

9.55 Solution.hpp File Reference

Classes

- class [Solution](#)

9.56 zbrent.cpp File Reference

Functions

- double [zbrent](#) (double(*func)(double,...), double x1, double x2, double tol)

9.56.1 Function Documentation

9.56.1.1 double zbrent (double(* *func*)(double,...), double *x1*, double *x2*, double *tol*)

Here is the call graph for this function:



10 Product Testing

The MPI component have been initially tested using a set of specially design tests. Later, the whole application was tested using Marmot [6].

The Sleuth algorithm has also been tested on a large ensemble of HEP data. Two sets of tests were performed: scalability and load balancing studies. Both of them were performed on a cluster in A.Soltan Institute for Nuclear Studies (INS) in Warsaw. The cluster consists of 5 dual AMD Athlon 2000MP+ (1.7 GHz) nodes running Red Hat 7.3 Linux. Additionally, in load balancing tests one PII 300 MHz node was used.

Scalability tests show that the time of typical run with generation of 10000 HSEs can be reduced from about half an hour with one worker node to 7 minutes with 9 worker nodes (see tab. 10.1). Because the application is oriented on computation and uses relatively little communication, the speedup grows almost linearly with number of worker nodes (fig. 10.1). Tests of load balancing were performed in a very asymmetrical configuration of nodes - one

no. of worker nodes	time/HSE [s]	speedup
1	0.197	1
2	0.133	1.5
3	0.091	2.2
4	0.070	2.8
5	0.058	3.4
6	0.054	3.6
7	0.048	4.1
8	0.044	4.5
9	0.040	5.0

Table 10.1: Results of scalability tests

of the worker nodes is a PII 300 MHz machine which was 5.6 times slower than other worker nodes (running on Athlon 1.7 GHz machines). Results are shown in tab. 10.2. One can observe that in configuration with one worker node slowdown made by the slow node is equal to ratio between CPU power of the slower node and normal one, while in configurations with more worker nodes the slowdown decreases.

no. of worker nodes	time/HSE [s]	slowdown
1	1.068	5.4
3	0.259	2.8
4	0.186	2.7
6	0.126	2.3
9	0.095	2.4

Table 10.2: Results of load balancing tests with one slower worker node

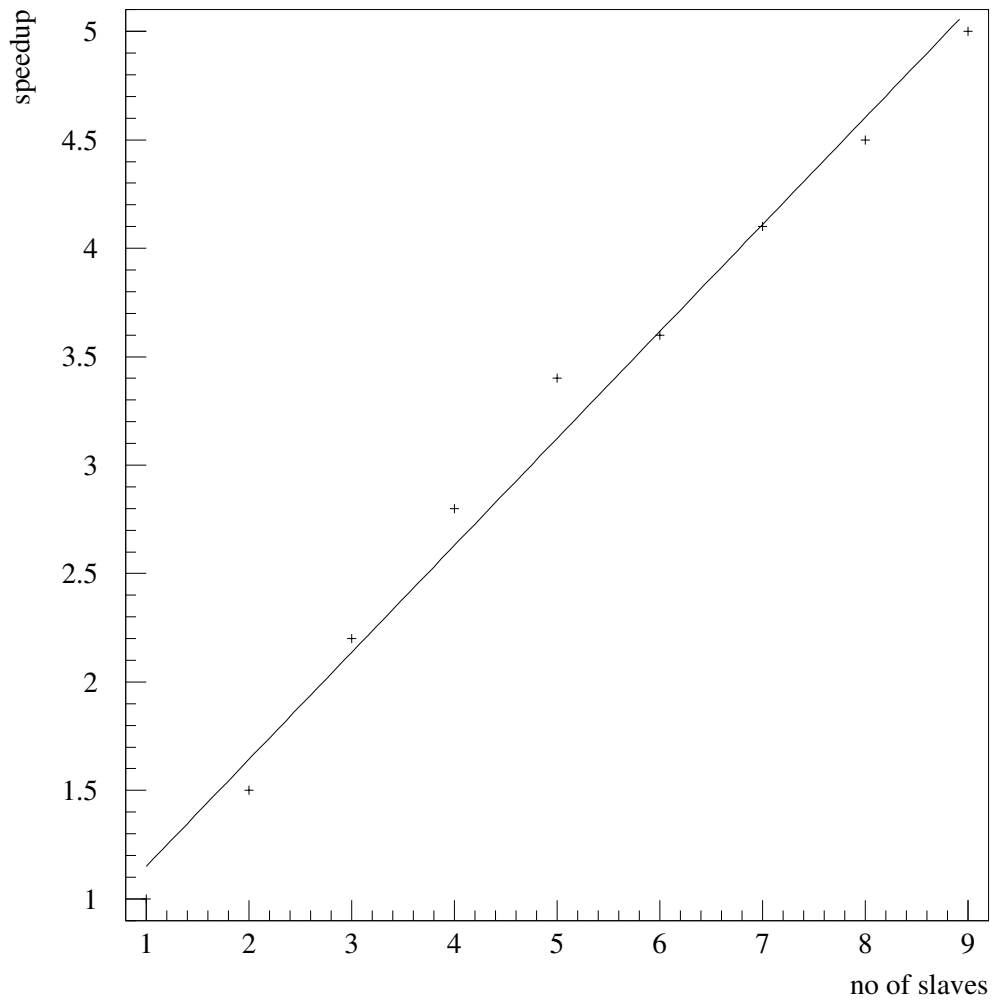


Figure 10.1: Speedup on a local cluster

11 EDG License Agreement

Copyright (c) 2005 CrossGrid. All rights reserved.

This software includes voluntary contributions made to the CrossGrid Project. For more information on CrossGrid, please see <http://www.eu-crossgrid.org>.

Installation, use, reproduction, display, modification and redistribution of this software, with or without modification, in source and binary forms, are permitted. Any exercise of rights under this license by you or your sub-licensees is subject to the following conditions:

1. Redistributions of this software, with or without modification, must reproduce the above copyright notice and the above license statement as well as this list of conditions, in the software, the user documentation and any other materials provided with the software.

2. The user documentation, if any, included with a redistribution, must include the following notice:

ŠaĚThis product includes software developed by the CrossGrid Project (<http://www.eu-crossgrid.org>).ŠaĚ

Alternatively, if that is where third-party acknowledgments normally appear, this acknowledgment must be reproduced in the software itself.

3. The names ŠaĚCrossGridŠaĚ and ŠaĚCGŠaĚ may not be used to endorse or promote software, or products derived therefrom, except with prior written permission by cgoffice@cyfronet.krakow.pl.

4. You are under no obligation to provide anyone with any bug fixes, patches, upgrades or other modifications, enhancements or derivatives of the features, functionality or performance of this software that you may develop. However, if you publish or distribute your modifications, enhancements or derivative works without contemporaneously requiring users to enter into a separate written license agreement, then you are deemed to have granted participants in the CrossGrid Project a worldwide, non-exclusive, royalty-free, perpetual license to install, use, reproduce, display, modify, redistribute and sub-license your modifications, enhancements or derivative works, whether in binary or source code form, under the license conditions stated in this list of conditions.

5. DISCLAIMER

THIS SOFTWARE IS PROVIDED BY THE CROSSGRID PROJECT AND CONTRIBUTORS ŠaĚAS ISŠaĚ AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, OF SATISFACTORY QUALITY, AND FITNESS FOR A PARTICULAR PURPOSE OR USE ARE DISCLAIMED. THE CROSSGRID PROJECT AND CONTRIBUTORS MAKE NO REPRESENTATION THAT THE SOFTWARE, MODIFICATIONS, ENHANCEMENTS OR DERIVATIVE WORKS THEREOF, WILL NOT INFRINGE ANY PATENT, COPYRIGHT, TRADE SECRET OR OTHER PROPRIETARY RIGHT.

6. LIMITATION OF LIABILITY

THE CROSSGRID PROJECT AND CONTRIBUTORS SHALL HAVE NO LIABILITY TO LICENSEE OR OTHER PERSONS FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, CONSEQUENTIAL, EXEMPLARY, OR PUNITIVE DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, LOSS OF USE, DATA OR PROFITS, OR BUSINESS INTERRUPTION, HOWEVER CAUSED AND ON ANY THEORY OF CONTRACT, WARRANTY, TORT (INCLUDING NEGLIGENCE), PRODUCT LIABILITY OR OTHERWISE, ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Bibliography

- [1] B. Abbott *et al.* [D0 Collaboration], “Search for new physics in e muon X data at D0 using SLEUTH: A quasi model independent search strategy for new physics,” *Phys. Rev. D* **62** (2000) [7](#)
- [2] MPI: A Message-Passing Interface Standard, Message Passing Interface Forum, 5 May 1994 <http://www-unix-mcs.anl.gov/mpi/standard.html> [7](#), [8](#)
- [3] MPICH. <http://www-unix-mcs.anl.gov/mpi/mpich> [7](#), [8](#)
- [4] Migrating Desktop. http://tulip.man.poznan.pl/pages/migrating_desktop.html [8](#)
- [5] N. Karonis, B. Toonen and I. Foster, “MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface”, *Journal of Parallel and Distributed Computing (JPDC)*, Vol. 63, No. 5, May 2003. [7](#), [9](#)
- [6] Tool for analysing and checking MPI-programs. <http://www.hlrs.de/people/mueller/projects/marmot/> [131](#)
- [7] The Globus Project. <http://www.globus.org> [9](#)
- [8] JDL (Job Description Language)
F. Pacini, “Job Attributes” DataGrid-01-TEN-0142-0_2, Oct 2003
http://www.infn.it/workload-grid/docs/DataGrid-01-TEN-0142-0_2.{doc,pdf}
- [9] E. Heymann, M. A. Senar, A. Fernandez, J. Salt, “Managing MPI Applications in Grid Environments”, *Proc. 2nd European Across Grids Conf. (AxGrid)*, Jan 2004, Nicosia, Cyprus. LNCS 3165
- [10] EU-DataGrid Data Management
<http://edg-wp2.web.cern.ch/edg-wp2>
- [11] M. Kosiedowski, M. Kupczyk, R. Lichwala, N. Meyer, B. Palak, M. Plociennik, P. Wolniewicz, S. Beco, “Mobile Work Environment for Grid Users. Grid Applications’ Framework”, *Computational Science - ICCS 2003, International Conference, Melbourne, Australia and St. Petersburg, Russia, June 2003. Proceedings, Part II.* LNCS 2658