FLARE Computing Library

Generated by Doxygen 1.8.11

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Module Documentation

3.1 Definitions

Constants defined in Flare.

Macros

- #define **CDEBUG** if (gsm_debug_on) (∗gsm_log) << "<GSM debug> "
- #define **CDEBUGC** if (gsm_debug_on) (∗gsm_log)
- #define **CINFO** if (gsm_info_on) (∗gsm_log) << "<GSM info> "
- #define **CINFOC** if (gsm_info_on) (∗gsm_log)
- #define **CWARN** if (gsm_warnings_on) cout << "<GSM WARNING> "
- #define **CERR** if (gsm_errors_on) cout << "<GSM ERROR> "

Variables

- ostream ∗ **gsm_log**
- const double **t_tol** = 1e-3
- const float **std_missing_value** = 9.9e20
- bool **gsm_info_on**
- bool **gsm_debug_on**
- bool **gsm_warnings_on**
- bool **gsm_errors_on**

3.1.1 Detailed Description

Constants defined in Flare.

Constants predefined in the library.

Author

Jaideep Joshi

Date

Sept 2018

3.2 Grid Operations

Classes

- class gVar

  Georeferenced Variable.

Functions

- vector< float > **createCoord** (float x0, float xf, int nx, float &dx)
- vector< float > **createCoord** (double x0, double xf, double dx, int &nx)
- vector< float > **createCoord_from_edges** (double x0, double xf, double dx, int &nx)
- void **printVar** (vector< float > &x, vector< float > &y, float *data)
- vector< int > **findGridBoxSW** (float x, float y, vector< float > &lons, vector< float > &lats)
- vector< int > **findGridBoxC** (float x, float y, vector< float > &lons, vector< float > &lats)
- vector< int > **billndices** (vector< float > &lons, vector< float > &lats, vector< float > &mlons, vector< float > &mlats)
- float **bilinear** (float x, float y, float iz, vector< float > &lons, vector< float > &lats, float *data, float missing↩Val=std_missing_value)
- float **bilinear** (int ilat, int ilon, int iz, vector< int > &indices, vector< float > &lons, vector< float > &lats, vector< float > &mlons, vector< float > &mlats, float *data, float missingVal=std_missing_value)
- float **cellVal** (float x, float y, float iz, vector< float > &lons, vector< float > &lats, float *data, float missing↩Val=std_missing_value)
- float **cellVal** (int ilat, int ilon, int iz, vector< int > &indices, vector< float > &lons, vector< float > &lats, vector< float > &mlons, vector< float > &mlats, float *data, float missingVal=std_missing_value)
- gVar **mask** (gVar &v, gVar &m, float val=0)
- gVar **Iterp** (gVar &v, vector< float > &xlons, vector< float > &xlats)
- int **IterpCube** (gVar &v, gVar &out, vector< int > &indices)
- int **cellRegridCube** (gVar &v, gVar &out, vector< int > &indices)
- gVar **coarseGrain_sum** (gVar &hires, vector< float > &xlons, vector< float > &xlats)
- gVar **coarseGrain_mean** (gVar &hires, vector< float > &xlons, vector< float > &xlats)
- gVar **binary** (gVar v, float thresh=0)

3.2.1 Detailed Description

3.3 Utilities

Various utility functions and classes, such as vector math, colour palettes, histograms, and date-time arithmatic.

Classes

- class Histogram

  A histogram class based on gsl_histogram.
- class Initializer

  A simple initializer that reads a parameter file and stores the values in a named map.
- class Colour_rgb

Functions

- string **int2str** (int n)
- float **str2float** (string s)
- int **str2int** (string s)
- int **IX3** (int ix, int iy, int iz, int nx, int ny)
- int **IX2** (int ix, int iy, int nx)
- void **printArray** (float v[ ], int n, ostream &lfout=cout)
- void **printArray** (vector< float > &v, ostream &lfout=cout, string send="", int n=0)
- void **printArray2d** (float v[ ], int rows, int columns)
- void **printArray2d** (vector< float > &v, int rows, int columns)
- void **printCube** (float v[ ], int nx, int ny, int nz=1, float ignoreVal=std_missing_value)
- void **reverseArray** (vector< float > &orig)
- void **reverseCube** (float v[ ], int nx, int ny, int nz=1, int n4=1, int n5=1)
- int ncIndexLo (vector< float > &v, float val)

  lower bound, edge for outliers
- int ncIndexHi (vector< float > &v, float val)

  upper bound, edge for outliers
- int lindexSW (vector< float > &v, float val)

  lower (S/W) bound, missing value for outliers
- int indexC (vector< float > &v, float val)

  cell index by center, missing value for outliers
- vector< float > max_vec (vector< float > &u, vector< float > &v)

  returns elementwise maximum
- float sum (vector< float > &v)

  Returns sum of vector.
- float **avg** (vector< float > &v)
- template<class T >
  void printSummary (T *data, int n, string s="")

  Data summaries.
- Colour_rgb **HSVtoRGB** (float h, float s, float v)
- vector< Colour_rgb > **createPalette_rainbow** (int N, float start, float end)
- vector< Colour_rgb > **createPalette_random** (int N, float start, float end)
- vector< Colour_rgb > **createPalette_grayscale** (int N, float start, float end)
- vector< Colour_rgb > **createPalette_ramp** (int N, Colour_rgb start, Colour_rgb end)
- void **printPalette** (vector< Colour_rgb > &p)

3.3.1 Detailed Description

Various utility functions and classes, such as vector math, colour palettes, histograms, and date-time arithmatic.

3.3.2 Function Documentation

3.3.2.1 template<class T > void printSummary (T * *data,* int *n,* string *s* = " ")

Data summaries.

Author

Jaideep Joshi

Date

11 May 2015

Print the summary of given data (min, max, mean, and histogram)

Parameters

data	Data array
n	Numeber of elements (array size)
s	Name of the array to prefix the printed output

Chapter 4

Class Documentation

4.1 Colour_rgb Class Reference

Public Member Functions

- **Colour_rgb** (float rr, float gg, float bb)

Public Attributes

- float **r**
- float **g**
- float **b**

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

- /home/jaideep/codes/Flare/include/palettes.h

4.2 gVar Class Reference

Georeferenced Variable.

`#include <gvar.h>`

Public Member Functions

- **gVar** (string name, string units, string tunits)
- int **initMetaFromFile** (string filename)
- int **_copyMeta** (const gVar &v)
- int **copyMeta** (const gVar &v)
- int **copyMeta** (const gVar &v, vector< float > &_lons, vector< float > &_lats, vector< float > &_levs)
- int **copyValues** (const gVar &v)
- int **setCoords** (vector< double > &t, vector< float > &le, vector< float > &la, vector< float > &lo)
- int **setTimeAtts** (int xntimes, double xtbase, float xtscale)
- int **printGrid** (ostream &lfout=std::cout)
- int **printGridIP** (ostream &lfout=std::cout)
- int **printValues** (ostream &lfout=std::cout)
- int **gt2ix** (double gt)
- double **ix2gt** (int ix)
- double **ix2gt_IST** (int ix)
- float **getValue** (float xlon, float xlat, float ilev=0)
- float **getCellValue** (float xlon, float xlat, float ilev=0)
- int **fill** (float f)
- int **sqrtVar** ()
- float & **operator()** (int ilon, int ilat, int ilev)
- float & **operator[ ]** (int i)
- gVar **operator+** (const gVar &v)
- gVar **operator+** (const float x)
- gVar **operator-** (const gVar &v)
- gVar **operator-** (const float x)
- gVar **operator*** (const gVar &v)
- gVar **operator*** (const float x)
- gVar **operator/** (const float x)
- gVar **operator/** (const gVar &v)
- void **setRegriddingMethod** (string m)
- int **createNcInputStream** (vector< string > files, vector< float > glim, string rm="bilinear")
- int **loadInputFileMeta** ()
- int **whichNextFile** (double gt)
- int **updateInputFile** (double gt)
- int **closeNcInputStream** ()
- int **readVar_gt** (double gt, int mode)
- int **readVar_it** (int tid)
- int **readVar_reduce_mean** (double gt1, double gt2)
- gVar **trend** (double gt1, double gt2)
- gVar **trend_gpu** (double gt1, double gt2)
- int **createOneShot** (string filename, vector< float > glim=vector< float >())
- int **readOneShot** (string filename, vector< float > glim=vector< float >())
- int **createNcOutputStream** (string filename)
- int **closeNcOutputStream** ()
- int **writeVar** (int itime)
- int **writeOneShot** (string filename)

Public Attributes

- int ntimes

  Number of timesteps that this variable references (note that at any point, only one timestep is held in the variable).
- int nlevs

  Number of levels in the data.
- int nlats

  Number of latitudes (rows) in the data.
- int nlons

  Number of longitudes (columns) in the data.
- vector< float > levs

  Levels.
- vector< float > lats

  Latitudes associated with data rows.
- vector< float > lons

  Longitudes associated with data columns.
- vector< double > times

  Time vector of the variable (This is useful when reading/writing data to files).
- double tbase

  Base time (values in the time vector are measured in units since this base time)
- float tscale

  Time unit in hours (hours/time-unit)
- float tstep

  Time-step in hours.
- double t

  The time for which values are currently held.
- string varname

  Variable name.
- string varunits

  Variable units.
- float **scale_factor**
- float **add_offset**
- int ncoords

  Number of coordinates in the associated inout file.
- int ivar1

  index of 1st data variable in the associated inout file
- float missing_value

  Missing value (what value to treat as missing data)
- vector< float > gridlimits

  Lat-Lon bounds.
- bool **lwrite**
- bool **lwriteSP**
- vector< float > values

  Data values. These are stored as a 1D array.

4.2.1 Detailed Description

Georeferenced Variable.

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

- /home/jaideep/codes/Flare/include/gvar.h

4.3 Histogram Class Reference

A histogram class based on gsl_histogram.

`#include <histogram.h>`

Public Member Functions

- Histogram ()
  Default constructor.
- Histogram (vector< float > &data, int nbins, float range_min=1e20, float range_max=1e20)
  Create histogram in one step by specifying number of breaks.
- Histogram (vector< float > &data, vector< double > &breaks)
  Create histogram in one step by specifying the breaks.
- Histogram (vector< float > &data, vector< float > &w, int nbins, float range_min=1e20, float range_↩ max=1e20)
  Create weighted histogram in one step by specifying number of breaks.
- Histogram (vector< float > &data, vector< float > &w, vector< double > &breaks)
  Create weighted histogram in one step by specifying breaks.
- int plot_console ()
  Plot the histogram to console.
- vector< float > getCounts ()
  Get the counts vector from the histogram.
- vector< float > getMids ()
  Get bin midpoints (Arithmatic mean of the bin ends)
- vector< float > getMids_log ()
  Get bin midpoints (Geometric mean of the bin ends)
- vector< float > getBreaks ()
  Get the breaks vector.
- int convertToPdf ()
  Normalize the counts to a probability distribution $ c = 1$.

Public Attributes

- gsl_histogram * h
  base GSL histogram

4.3.1 Detailed Description

A histogram class based on gsl_histogram.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Histogram::Histogram (vector< float > & *data*, int *nbins*, float *range_min* = `1e20`, float *range_max* = `1e20`)

Create histogram in one step by specifying number of breaks.

Parameters

data	Data from which to create histogram
nbins	Number of bins (Equally spaced bins will be created)
range_min	Min value (if not specified, will be calculated from the data)
range_max	Max value (if not specified, will be calculated from the data)

4.3.2.2 Histogram::Histogram (vector< float > & *data,* vector< double > & *breaks*)

Create histogram in one step by specifying the breaks.

Parameters

data	Data from which to create histogram
breaks	Breaks

4.3.2.3 Histogram::Histogram (vector< float > & *data,* vector< float > & *w,* int *nbins,* float *range_min* = `1e20`, float *range_max* = `1e20`)

Create weighted histogram in one step by specifying number of breaks.

Parameters

data	Data from which to create histogram
w	Weights (multiplied to the data before adding to counts)
nbins	Number of bins (Equally spaced bins will be created)
range_min	Min value (if not specified, will be calculated from the data)
range_max	Max value (if not specified, will be calculated from the data)

4.3.2.4 Histogram::Histogram (vector< float > & *data,* vector< float > & *w,* vector< double > & *breaks*)

Create weighted histogram in one step by specifying breaks.

Parameters

data	Data from which to create histogram
w	Weights (multiplied to the data before adding to counts)
breaks	Breaks

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

- /home/jaideep/codes/Flare/include/histogram.h

4.4 Initializer Class Reference

A simple initializer that reads a parameter file and stores the values in a named map.

`#include <initializer.h>`

Public Member Functions

- Initializer ()
- Initializer (string fname)
- void setInitFile (string fname)
- void readFile ()
- string getString (string s)
- float getScalar (string s)
- vector< float > getArray (string s, int size)
- void printVars ()

4.4.1 Detailed Description

A simple initializer that reads a parameter file and stores the values in a named map.

The parameter file must have three sections - STRINGS, SCALARS, ARRAYS. Sections start with '>'. Each section has name-value pairs separated by whitespace. Arrays have a name followed by values, ending in '-1'. Comments are allowed. Comments start with "# " (note the space) and can come either on a new line or on the same line after the name-value(s) pair.

Here is an example parameter file:

```
> STRINGS
sim_name      mySimution
output_file   ~/output/test.txt

> SCALARS
graphics      1           # Do we want graphics to be on?
timesteps     1000        # For how many timesteps do we run the simulation?
dt            0.1
# This is also a comment

> ARRAYS
parameter1    1 2 3 4 5 6 -1
```

4.4.2 Constructor & Destructor Documentation

4.4.2.1 Initializer::Initializer ( )

Default constructor

4.4.2.2 Initializer::Initializer (string *fname*)

The initializer can be created by specifying the parameter file name.

Parameters

fname	filename

4.4.3 Member Function Documentation

4.4.3.1 vector<float> Initializer::getArray (string *s*, int *size*)

Read an array defined in the parameter file's ARRAYS section.

Returns

vector of floating point numbers containing the array

Parameters

s	name of the array
size	length of the array

4.4.3.2 float Initializer::getScalar (string *s*)

Get a float variable defined in the parameter file's SCALARS section.

Parameters

s	Name of the variable

4.4.3.3 string Initializer::getString (string *s*)

Get string variable defined in the parameter file's STRINGS section.

Parameters

s	Name of the variable

4.4.3.4 void Initializer::printVars ( )

Print out the values that have been read from the maps.

4.4.3.5 void Initializer::readFile ( )

Read values from the parameters file and store them in maps.

4.4.3.6 void Initializer::setInitFile (string *fname*)

This function can be used to specify the parameter file. For example, if the initializer was created with the default constructor.

Parameters

fname	filename

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

- /home/jaideep/codes/Flare/include/initializer.h

4.5 NcFile_handle Class Reference

Public Member Functions

- void **setMapLimits** (float xwlon, float xelon, float xslat, float xnlat)
- int **open** (string s, string m, const float glimits[4])
- int **close** ()
- int **getMeta** ()
- int **readCoordData** (gVar &v)
- int **readTime** (gVar &v)
- int **readCoords** (gVar &v)
- int **getVarID** (string varname)
- int **readVarAtts** (gVar &v, int ivar=-1)
- int **readVar** (gVar &v, int itime, int iVar=-1)
- int **readVar_gt** (gVar &v, double gt, int mode, int iVar=-1)
- int **readVar_parallel** (gVar &v, int itime, int iVar=-1)
- int **writeCoords** (gVar &v, bool wr=true)
- NcVar ∗ **createVar** (gVar &v)
- int **writeVar** (gVar &v, NcVar ∗vVar, int itime)
- int **writeTimeValues** (gVar &v)

Public Attributes

- NcFile ∗ **dFile**
- string **fname**
- string **mode**
- NcVar ∗ **tVar**
- NcVar ∗ **levVar**
- NcVar ∗ **latVar**
- NcVar ∗ **lonVar**
- NcDim ∗ **tDim**
- NcDim ∗ **levDim**
- NcDim ∗ **latDim**
- NcDim ∗ **lonDim**
- int **ntimes**
- int **nlevs**
- int **nlats**

- int **nlons**
- int **ncoords**
- int **nvars**
- string **levname**
- string **levunits**
- bool **latSN**
- bool **lonPos**
- float **wlon**
- float **elon**
- float **slat**
- float **nlat**
- int **wlonix**
- int **elonix**
- int **slatix**
- int **nlatix**
- int **ilon0**
- int **ilat0**
- int **itime0**
- int **ilonf**
- int **ilatf**
- bool **mplimited**
- int **firstVarID**

Static Public Attributes

- static const int **NC_ERR** = 2

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

- /home/jaideep/codes/Flare/include/ncio.h

4.6 ResourceGrid Class Reference

Public Member Functions

- void **init** (Initializer &I)
- void **update** ()
- void **diffuse** ()
- void **grow** (float *ke_all_dev)
- float **sumResource** ()
- void **freeMemory** ()

Public Attributes

- int **nx**
- int **ny**
- float **L**
- float **dL**
- float **dt**
- bool **graphics**
- float **D**
- float * **r**
- float * **r_dev**
- float * **K**
- float * **K_dev**
- float * **res**
- float * **res_dev**
- float * **res_new_dev**
- float **totalRes**

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

- /home/jaideep/codes/Flare/include/resource.h

4.7 TurbulenceEngine Class Reference

Public Member Functions

- void **initRNG** ()
- void **init** (Initializer &I)
- void **generateSpectrum** ()
- void **calcEquilPsi** ()
- void **generateNoise_gpu** ()
- void **modifyPsi_gpu** ()
- void **transformPsi** ()
- void **calcVelocityField** ()
- void **normalize_psi** ()
- void **update** ()
- void **updateColorMap** ()
- void **printMap** (string mapname, ofstream &fout)
- void **freeMemory** ()

Public Attributes

- int **nx**
- int **ny**
- float **L**
- float **xmin**
- float **xmax**
- float **ymin**
- float **ymax**
- float **mu**
- float **xi**
- float **nu**
- float **lambda0**
- float **dt**
- cufftHandle **plan**
- int **nlevCol**
- float * **lambda**
- float * **lambda_dev**
- cufftComplex * **Psi**
- cufftComplex * **Psi_dev**
- cufftComplex * **psi**
- cufftComplex * **psi_dev**
- cufftComplex * **Z**
- cufftComplex * **Z_dev**
- float2 * **vel_field**
- float2 * **vel_field_dev**
- curandState * **te_dev_XWstates**
- int * **te_seeds_h**
- int * **te_seeds_dev**

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

- /home/jaideep/codes/Flare/include/turbulence.h

Index