

DopplerToolkit Reference Manual

Generated by Doxygen 1.4.1

Thu May 3 00:53:59 2007

Contents

1 DopplerToolkit Namespace Index	1
1.1 DopplerToolkit Namespace List	1
2 DopplerToolkit Hierarchical Index	3
2.1 DopplerToolkit Class Hierarchy	3
3 DopplerToolkit Class Index	5
3.1 DopplerToolkit Class List	5
4 DopplerToolkit File Index	7
4.1 DopplerToolkit File List	7
5 DopplerToolkit Namespace Documentation	9
5.1 APL Namespace Reference	9
5.2 LOAPEX Namespace Reference	10
6 DopplerToolkit Class Documentation	11
6.1 LOAPEX::CADFcompare Class Reference	11
6.2 LOAPEX::CADFFile Class Reference	12
6.3 LOAPEX::CADFFileAdapter Class Reference	15
6.4 LOAPEX::CADFInfo Class Reference	17
6.5 LOAPEX::CADFSource Class Reference	19
6.6 LOAPEX::CBase3DNavigationData Class Reference	22
6.7 LOAPEX::CBaseFileAdapter Class Reference	25
6.8 LOAPEX::CBaseFilter Class Reference	27

6.9	LOAPEX::CBaseMapSolver Class Reference	29
6.10	LOAPEX::CBrentMapSolver Class Reference	32
6.11	LOAPEX::CCompoundTime Class Reference	34
6.12	LOAPEX::CDemultiplexor Class Reference	36
6.13	LOAPEX::CFakeRXNavigationData Class Reference	38
6.14	LOAPEX::CFakeTXNavigationData Class Reference	41
6.15	LOAPEX::CFixedNavigationData Class Reference	44
6.16	LOAPEX::CGappyADFStream Class Reference	47
6.17	LOAPEX::CGenericSource Class Reference	49
6.18	LOAPEX::CGeoParameters Class Reference	51
6.19	LOAPEX::CInterpolatedTableFunction Class Reference	53
6.20	LOAPEX::CMultichannelFilter Class Reference	55
6.21	LOAPEX::CMultiplexor Class Reference	57
6.22	LOAPEX::CMUXFile Class Reference	59
6.23	LOAPEX::CMUXFileAdapter Class Reference	64
6.24	LOAPEX::CNetCDFNavigationData Class Reference	66
6.25	LOAPEX::CNewtonMapSolver Class Reference	69
6.26	LOAPEX::CRex3DNavigationData Class Reference	71
6.27	LOAPEX::CSingleChannelAllPassFilter Class Reference	74
6.28	LOAPEX::CSingleChannelAllStopFilter Class Reference	76
6.29	LOAPEX::CSingleChannelSincFilter Class Reference	78
6.30	LOAPEX::CTimeBaseMapEngine Class Reference	81
6.31	LOAPEX::CTimeTag Class Reference	84
6.32	LOAPEX::CZeroSource Class Reference	87
6.33	dEuclideanVector_t Struct Reference	89
6.34	APL::Error Class Reference	90
6.35	APL::FatalError Class Reference	91
6.36	tstream Class Reference	92
7	DopplerToolkit File Documentation	93
7.1	adfadapter.cpp File Reference	93

7.2	adfadapter.hpp File Reference	94
7.3	adffile.cpp File Reference	95
7.4	adffile.hpp File Reference	96
7.5	allpass.cpp File Reference	97
7.6	allpass.hpp File Reference	98
7.7	allstop.cpp File Reference	99
7.8	allstop.hpp File Reference	100
7.9	basefileadapter.hpp File Reference	101
7.10	basefilter.cpp File Reference	102
7.11	basefilter.hpp File Reference	103
7.12	basemapsolver.hpp File Reference	104
7.13	compoundtime.cpp File Reference	105
7.14	compoundtime.hpp File Reference	106
7.15	demultiplexor.cpp File Reference	107
7.16	demultiplexor.hpp File Reference	108
7.17	errors.hpp File Reference	109
7.18	gappystream.cpp File Reference	110
7.19	gappystream.hpp File Reference	111
7.20	mapengine.cpp File Reference	112
7.21	mapengine.hpp File Reference	113
7.22	mapsolvers.cpp File Reference	114
7.23	mapsolvers.hpp File Reference	115
7.24	multichannelfilter.cpp File Reference	116
7.25	multichannelfilter.hpp File Reference	117
7.26	multiplexor.cpp File Reference	118
7.27	multiplexor.hpp File Reference	119
7.28	muxadapter.cpp File Reference	120
7.29	muxadapter.hpp File Reference	121
7.30	muxfile.cpp File Reference	122
7.31	muxfile.hpp File Reference	123
7.32	navigation.cpp File Reference	124

7.33	navigation.hpp File Reference	125
7.34	nrutil.c File Reference	127
7.35	parameters.hpp File Reference	129
7.36	refman.dox File Reference	130
7.37	sincfilter.cpp File Reference	131
7.38	sincfilter.hpp File Reference	132
7.39	spline.c File Reference	133
7.40	splint.c File Reference	134
7.41	tablefunction.cpp File Reference	135
7.42	tablefunction.hpp File Reference	136
7.43	timetag.cpp File Reference	137
7.44	timetag.hpp File Reference	138
7.45	tstream.cpp File Reference	139
7.46	tstream.hpp File Reference	140

Chapter 1

DopplerToolkit Namespace Index

1.1 DopplerToolkit Namespace List

Here is a list of all namespaces with brief descriptions:

APL	9
LOAPEX	10

Chapter 2

DopplerToolkit Hierarchical Index

2.1 DopplerToolkit Class Hierarchy

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

LOAPEX::CADFcompare	11
LOAPEX::CADFFile	12
LOAPEX::CADFInfo	17
LOAPEX::CBase3DNavigationData	22
LOAPEX::CFakeRXNavigationData	38
LOAPEX::CFakeTXNavigationData	41
LOAPEX::CFixedNavigationData	44
LOAPEX::CNetCDFNavigationData	66
LOAPEX::CRex3DNavigationData	71
LOAPEX::CBaseFileAdapter	25
LOAPEX::CADFFileAdapter	15
LOAPEX::CMUXFileAdapter	64
LOAPEX::CBaseFilter	27
LOAPEX::CSingleChannelAllPassFilter	74
LOAPEX::CSingleChannelAllStopFilter	76
LOAPEX::CSingleChannelSincFilter	78
LOAPEX::CBaseMapSolver	29
LOAPEX::CBrentMapSolver	32
LOAPEX::CNewtonMapSolver	69
LOAPEX::CCompoundTime	34
LOAPEX::CDemultiplexor	36
LOAPEX::CGappyADFStream	47

LOAPEX::CGenericSource	49
LOAPEX::CADFSource	19
LOAPEX::CZeroSource	87
LOAPEX::CGeoParameters	51
LOAPEX::CInterpolatedTableFunction	53
LOAPEX::CMultichannelFilter	55
LOAPEX::CMultiplexor	57
LOAPEX::CMUXFile	59
LOAPEX::CTimeBaseMapEngine	81
LOAPEX::CTimeTag	84
dEuclideanVector_t	89
APL::Error	90
APL::FatalError	91
tstream	92

Chapter 3

DopplerToolkit Class Index

3.1 DopplerToolkit Class List

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

LOAPEX::CADFcompare	11
LOAPEX::CADFFile	12
LOAPEX::CADFFileAdapter	15
LOAPEX::CADFInfo	17
LOAPEX::CADFSource	19
LOAPEX::CBase3DNavigationData	22
LOAPEX::CBaseFileAdapter	25
LOAPEX::CBaseFilter	27
LOAPEX::CBaseMapSolver	29
LOAPEX::CBrentMapSolver	32
LOAPEX::CCompoundTime	34
LOAPEX::CDemultiplexor	36
LOAPEX::CFakeRXNavigationData	38
LOAPEX::CFakeTXNavigationData	41
LOAPEX::CFixedNavigationData	44
LOAPEX::CGappyADFStream	47
LOAPEX::CGenericSource	49
LOAPEX::CGeoParameters	51
LOAPEX::CInterpolatedTableFunction	53
LOAPEX::CMultichannelFilter	55
LOAPEX::CMultiplexor	57
LOAPEX::CMUXFile	59
LOAPEX::CMUXFileAdapter	64
LOAPEX::CNetCDFNavigationData	66
LOAPEX::CNewtonMapSolver	69

LOAPEX::CRex3DNavigationData	71
LOAPEX::CSingleChannelAllPassFilter	74
LOAPEX::CSingleChannelAllStopFilter	76
LOAPEX::CSingleChannelSincFilter	78
LOAPEX::CTimeBaseMapEngine	81
LOAPEX::CTimeTag	84
LOAPEX::CZeroSource	87
dEuclideanVector_t	89
APL::Error	90
APL::FatalError	91
tstream	92

Chapter 4

DopplerToolkit File Index

4.1 DopplerToolkit File List

Here is a list of all files with brief descriptions:

adfadapter.cpp	93
adfadapter.hpp	94
adfile.cpp	95
adfile.hpp	96
allpass.cpp	97
allpass.hpp	98
allstop.cpp	99
allstop.hpp	100
basefileadapter.hpp	101
basefilter.cpp	102
basefilter.hpp	103
basemapsolver.hpp	104
compoundtime.cpp	105
compoundtime.hpp	106
demultiplexor.cpp	107
demultiplexor.hpp	108
errors.hpp	109
gappystream.cpp	110
gappystream.hpp	111
mapengine.cpp	112
mapengine.hpp	113
mapsolvers.cpp	114
mapsolvers.hpp	115
multichannelfilter.cpp	116
multichannelfilter.hpp	117

multiplexor.cpp	118
multiplexor.hpp	119
muxadapter.cpp	120
muxadapter.hpp	121
muxfile.cpp	122
muxfile.hpp	123
navigation.cpp	124
navigation.hpp	125
nrutil.c	127
parameters.hpp	129
sincfilter.cpp	131
sincfilter.hpp	132
spline.c	133
splint.c	134
tablefunction.cpp	135
tablefunction.hpp	136
timetag.cpp	137
timetag.hpp	138
tstream.cpp	139
tstream.hpp	140

Chapter 5

DopplerToolkit Namespace Documentation

5.1 APL Namespace Reference

Classes

- class **Error**
- class **FatalError**

5.2 LOAPEX Namespace Reference

Classes

- class **CADFFileAdapter**
- class **CADFInfo**
- class **CADFFile**
- class **CSingleChannelAllPassFilter**
- class **CSingleChannelAllStopFilter**
- class **CBaseFileAdapter**
- class **CBaseFilter**
- class **CBaseMapSolver**
- class **CCompoundTime**
- class **CDemultiplexor**
- class **CGenericSource**
- class **CZeroSource**
- class **CADFSource**
- class **CADFcompare**
- class **CGappyADFStream**
- class **CTimeBaseMapEngine**
- class **CBrentMapSolver**
- class **CNewtonMapSolver**
- class **CMultichannelFilter**
- class **CMultiplexor**
- class **CMUXFileAdapter**
- class **CMUXFile**
- class **CBase3DNavigationData**
- class **CRex3DNavigationData**
- class **CNetCDFNavigationData**
- class **CFakeRXNavigationData**
- class **CFakeTXNavigationData**
- class **CFixedNavigationData**
- class **CGeoParameters**
- class **CSingleChannelSincFilter**
- class **CInterpolatedTableFunction**
- class **CTimeTag**

Chapter 6

DopplerToolkit Class Documentation

6.1 LOAPEX::CADFcompare Class Reference

```
#include <gappystream.hpp>
```

Public Member Functions

- `bool operator() (const CGenericSource *f1, const CGenericSource *f2) const`

6.1.1 Detailed Description

This is a function object, used for sorting a `std::list` of source objects using the `sort()` method.

6.1.2 Member Function Documentation

- ##### 6.1.2.1 `bool LOAPEX::CADFcompare::operator() (const CGenericSource * f1, const CGenericSource * f2) const`

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

- `gappystream.hpp`
- `gappystream.cpp`

6.2 LOAPEX::CADFFile Class Reference

```
#include <adffile.hpp>
```

Public Member Functions

- **CADFFile** (void)
- **CADFFile** (const char *filename)
- **~CADFFile** ()
- **bool eof** (void)
- **void echo_header** (std::string &s)
- **bool is_loaded** () const
- **void get_info_copy** (LOAPEX::CADFInfo &) const
- **void get_sample_rate** (int *fs) const
- **void get_start_second** (long *l) const
- **void get_total_scans** (long *t) const
- **void get_total_channels** (int *c) const
- **void get_scan** (std::vector< short > &V)
- **int read_entire_channel** (const int channel, char *buffer)
- **void print** (void) const

6.2.1 Detailed Description

ADFFile declaration – a class that interfaces the programmer with the .adf files produced by Scripps

So far, these are meant solely to model existing files on disk. The constructors are not engineered to construct themselves for .adf files that are to be created.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 LOAPEX::CADFFile::CADFFile (void)

Generic ctor. Why is this needed?

6.2.2.2 LOAPEX::CADFFile::CADFFile (const char * filename)

Typical ctor, takes C string filename.

6.2.2.3 LOAPEX::CADFFile::~~CADFFile ()

6.2.3 Member Function Documentation

6.2.3.1 void LOAPEX::CADFFile::echo_header (std::string & s)

Supplies the whole header in the string.

6.2.3.2 bool LOAPEX::CADFFile::eof (void)

Returns true if the next scan will fail.

6.2.3.3 void LOAPEX::CADFFile::get_info_copy (LOAPEX::CADFInfo &) const

Supplies a copy of the private info.

6.2.3.4 void LOAPEX::CADFFile::get_sample_rate (int * fs) const

Supplies the sample rate, from the file (header). Note that we assume an integer sample rate. This has generally held true for all ATOC hardware influenced by Kurt Metzger, but of course will not always be true in the future.

6.2.3.5 void LOAPEX::CADFFile::get_scan (std::vector< short > & V)

Read the next "scan" of samples, one from each channel. Note for .adf files that the data is not multiplexed with channel incrementing fastest. Instead, ALL of channel 1 comes first, then ALL of channel 2. This runs contrary to how hardware works, and cannot support really long acquisitions (unless one has a lot of memory) or, in particular, acquisitions where the collection time is not known before capture start. This routine does all the seeks necessary to find the next sample for each channel even though it may be located way down inside the file. Clears **std::vector**(p. 128) V first.

6.2.3.6 void LOAPEX::CADFFile::get_start_second (long * l) const

Supplies the start second (from the header).

6.2.3.7 void LOAPEX::CADFFile::get_total_channels (int * c)
const

Supplies the total number of sensors (from the header).

6.2.3.8 void LOAPEX::CADFFile::get_total_scans (long * t) const

Supplies the total number of scans (from the header).

6.2.3.9 bool LOAPEX::CADFFile::is_loaded () const

Returns true is the constructor finished without error. Typically this means you can pull scans from it.

6.2.3.10 void LOAPEX::CADFFile::print (void) const

Ubiquitous diagnostic routine.

6.2.3.11 int LOAPEX::CADFFile::read_entire_channel (const int
channel, char * *buffer*)

This is the converse of the above. Reads all the samples for the given channel. So this is reading consecutive 16 bit words off the disk. Buffer has to be the right size – user should be able to calculate that from header parameters available through this class interface.

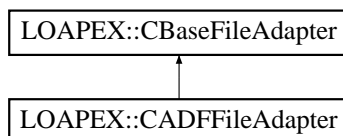
The documentation for this class was generated from the following files:

- **adffile.hpp**
- **adffile.cpp**

6.3 LOAPEX::CADFFileAdapter Class Reference

```
#include <adfadapter.hpp>
```

Inheritance diagram for LOAPEX::CADFFileAdapter::



Public Member Functions

- **CADFFileAdapter** (LOAPEX::CADFFile *p)
- **~CADFFileAdapter** (void)
- void **get_scan** (std::vector< short > &V)
- void **get_start_second** (long *s) const
- void **get_total_scans** (long *t) const
- void **get_total_channels** (int *c) const
- void **get_sample_rate** (int *fs) const
- void **get_header** (std::string &h) const

6.3.1 Detailed Description

An object adapter class for .adf files.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 LOAPEX::CADFFileAdapter::CADFFileAdapter (LOAPEX::CADFFile * p)

6.3.2.2 LOAPEX::CADFFileAdapter::~~CADFFileAdapter (void)

6.3.3 Member Function Documentation

6.3.3.1 void LOAPEX::CADFFileAdapter::get_header (std::string & h) const [virtual]

Supplies the whole header via a string.

Implements LOAPEX::CBaseFileAdapter (p. 25).

6.3.3.2 void LOAPEX::CADFFFileAdapter::get_sample_rate (int * fs) const

6.3.3.3 void LOAPEX::CADFFFileAdapter::get_scan (std::vector< short > & V) [virtual]

Read the next "scan" of samples, one from each channel. Clears `std::vector`(p. 128) V first.

Implements `LOAPEX::CBaseFileAdapter` (p. 26).

6.3.3.4 void LOAPEX::CADFFFileAdapter::get_start_second (long * s) const [virtual]

Supplies the "start second" from the file (header)

Implements `LOAPEX::CBaseFileAdapter` (p. 26).

6.3.3.5 void LOAPEX::CADFFFileAdapter::get_total_channels (int * c) const [virtual]

Supplies the total number of channels, from the file (header)

Implements `LOAPEX::CBaseFileAdapter` (p. 26).

6.3.3.6 void LOAPEX::CADFFFileAdapter::get_total_scans (long * t) const [virtual]

Supplies the total number of channels, from the file (header)

Implements `LOAPEX::CBaseFileAdapter` (p. 26).

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

- `adfadapter.hpp`
- `adfadapter.cpp`

6.4 LOAPEX::CADFInfo Class Reference

```
#include <adffile.hpp>
```

Public Member Functions

- **CADFInfo** (void)
- **CADFInfo** (std::ostream &os)
- **~CADFInfo** (void)
- void **get_id** (std::string &s) const
- void **get_sample_rate** (int *) const
- void **get_site** (std::string &s) const
- void **get_total_scans** (long *) const
- void **get_total_phones** (int *) const
- void **get_run_start** (std::string &t, long &epoch_seconds) const
- void **get_header** (std::string &h) const
- void **print** (void)

6.4.1 Detailed Description

CADFInfo(p. 17) encapsulates the parameters of the .adf file. Think of this as an interface to the header.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 LOAPEX::CADFInfo::CADFInfo (void)

Ctor for a plain vanilla object:

6.4.2.2 LOAPEX::CADFInfo::CADFInfo (std::ostream & os)

Ctor that accepts a sstream which should contain the header of a file.

6.4.2.3 LOAPEX::CADFInfo::~CADFInfo (void)

6.4.3 Member Function Documentation

6.4.3.1 void LOAPEX::CADFInfo::get_header (std::string & h) const

Supplies the entire header in one big string.

6.4.3.2 void LOAPEX::CADFInfo::get_id (std::string & s) const

Supplies the "ID" string used by these files

6.4.3.3 void LOAPEX::CADFInfo::get_run_start (std::string & t, long & epoch_seconds) const

Supplies the star time, in whole seconds, of the acquisition, from the file (header). The string is the human readable translation.

6.4.3.4 void LOAPEX::CADFInfo::get_sample_rate (int * s) const

Supplies the sample rate, from the file (header). Note that we assume an integer sample rate. This has generally held true for all ATOC hardware influenced by Kurt Metzger, but of course will not always be true in the future.

6.4.3.5 void LOAPEX::CADFInfo::get_site (std::string & s) const

Supplies the site string used by these files

6.4.3.6 void LOAPEX::CADFInfo::get_total_phones (int *) const

Supplies the total number of sensors, from the file (header)

6.4.3.7 void LOAPEX::CADFInfo::get_total_scans (long *) const

Supplies the total number of scans, from the file (header)

6.4.3.8 void LOAPEX::CADFInfo::print (void)

Diagnostic dump of member data.

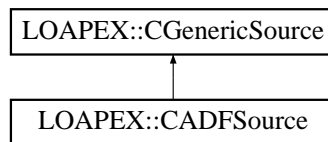
The documentation for this class was generated from the following files:

- **adffile.hpp**
- **adffile.cpp**

6.5 LOAPEX::CADFSource Class Reference

```
#include <gappystream.hpp>
```

Inheritance diagram for LOAPEX::CADFSource::



Public Member Functions

- **CADFSource** (const char *filename)
- **~CADFSource** (void)
- void **get_scan** (std::vector< short > &V)
- void **get_start_second** (long *s) const
- void **get_total_scans** (long *t) const
- void **get_total_channels** (int *c) const
- void **get_sample_rate** (int *fs) const
- void **get_type** (char *c) const

6.5.1 Detailed Description

Specialized class for ADF files. This is kind of a lite weight ADF file class, primarily for the gappystream application.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 LOAPEX::CADFSource::CADFSource (const char **filename*)

ctor takes a C string filename.

6.5.2.2 LOAPEX::CADFSource::~~CADFSource (void)

6.5.3 Member Function Documentation

6.5.3.1 void LOAPEX::CADFSource::get_sample_rate (int * fs) const [virtual]

Supplies the sample rate, from the file (header). Note that we assume an integer sample rate. This has generally held true for all ATOC hardware influenced by Kurt Metzger, but of course will not always be true in the future.

Implements **LOAPEX::CGenericSource** (p. 49).

6.5.3.2 void LOAPEX::CADFSource::get_scan (std::vector< short > & V) [virtual]

Read the next "scan" of samples, one from each channel. Note for .adf files that the data is not multiplexed with channel incrementing fastest. Instead, ALL of channel 1 comes first, then ALL of channel 2. This runs contrary to how hardware works, and cannot support really long acquisitions (unless one has a lot of memory) or, in particular, acquisitions where the collection time is not known before capture start. This routine does all the seeks necessary to find the next sample for each channel even though it may be located way down inside the file. Clears **std::vector**(p. 128) V first.

Implements **LOAPEX::CGenericSource** (p. 50).

6.5.3.3 void LOAPEX::CADFSource::get_start_second (long * s) const [virtual]

Supplies the "start second" from the file (header)

Implements **LOAPEX::CGenericSource** (p. 50).

6.5.3.4 void LOAPEX::CADFSource::get_total_channels (int * c) const [virtual]

Supplies the total number of channels, from the file (header)

Implements **LOAPEX::CGenericSource** (p. 50).

6.5.3.5 void LOAPEX::CADFSource::get_total_scans (long * t) const [virtual]

Supplies the total number of scans, from the file (header)

Implements **LOAPEX::CGenericSource** (p. 50).

6.5.3.6 `void LOAPEX::CADFSource::get_type (char * c) const`
[virtual]

Simple cosmetic function, returns character "A" to indicate and .adf file. Used when assembling a gappy stream to indicate a chunk of real (i.e., .adf) data.

Implements **LOAPEX::CGenericSource** (p. 50).

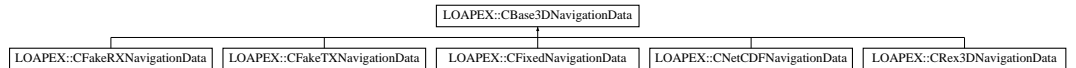
The documentation for this class was generated from the following files:

- **gappystream.hpp**
- **gappystream.cpp**

6.6 LOAPEX::CBase3DNavigationData Class Reference

```
#include <navigation.hpp>
```

Inheritance diagram for LOAPEX::CBase3DNavigationData::



Public Member Functions

- **CBase3DNavigationData** (void)
- **~CBase3DNavigationData** (void)
- virtual void **get_file_first_time** (time_t *pt) const =0
- virtual void **get_file_final_time** (time_t *pt) const =0
- virtual void **get_position** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *X) const =0
- virtual void **get_velocity** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *V) const =0

Protected Attributes

- time_t **file_time_first_**
- time_t **file_time_final_**

6.6.1 Detailed Description

Pure virtual base class, used primarily to specify the interface for "navigational" data. The intent here is to encapsulate data and present and interface that will yield the 3-D position or velocity for ANY time. Kind of like $f:R^1 \rightarrow R^3$. Derived classes that use tabulated data (which will be typical for real world data) will have to incorporate their own interpolators in order to provide positions (or velocities) for times in between tabulated times.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 LOAPEX::CBase3DNavigationData::CBase3DNavigationData (void)

Ctor will get overridden in derived classes.

6.6.2.2 LOAPEX::CBase3DNavigationData::~~CBase3DNavigationData (void)

6.6.3 Member Function Documentation

6.6.3.1 virtual void LOAPEX::CBase3DNavigationData::get_file_final_time (time_t * pt) const [pure virtual]

Supplies the time of the last valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implemented in [LOAPEX::CRex3DNavigationData](#) (p. 72), [LOAPEX::CNetCDFNavigationData](#) (p. 67), [LOAPEX::CFakeRXNavigationData](#) (p. 39), [LOAPEX::CFakeTXNavigationData](#) (p. 42), and [LOAPEX::CFixedNavigationData](#) (p. 45).

6.6.3.2 virtual void LOAPEX::CBase3DNavigationData::get_file_first_time (time_t * pt) const [pure virtual]

Supplies the time of the first valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implemented in [LOAPEX::CRex3DNavigationData](#) (p. 72), [LOAPEX::CNetCDFNavigationData](#) (p. 67), [LOAPEX::CFakeRXNavigationData](#) (p. 39), [LOAPEX::CFakeTXNavigationData](#) (p. 42), and [LOAPEX::CFixedNavigationData](#) (p. 45).

6.6.3.3 virtual void LOAPEX::CBase3DNavigationData::get_position (const LOAPEX::CCompoundTime t, dEuclideanVector_t * X) const [pure virtual]

Given a compound time, retrieves the position at that time. Derived classes will have to figure out what to do if there is no position at that exact time.

Implemented in [LOAPEX::CRex3DNavigationData](#) (p. 72), [LOAPEX::CNetCDFNavigationData](#) (p. 68), [LOAPEX::CFakeRXNavigationData](#) (p. 39), [LOAPEX::CFakeTXNavigationData](#) (p. 42), and [LOAPEX::CFixedNavigationData](#) (p. 45).

6.6.3.4 virtual void LOAPEX::CBase3DNavigationData::get_
velocity (const LOAPEX::CCompoundTime t,
dEuclideanVector _t * V) const [pure virtual]

Given a compound time, retrieves the velocity at that time. Derived classes will have to figure out what to do if there is no velocity at that exact time.

Implemented in LOAPEX::CRex3DNavigationData (p. 72), LOAPEX::CNetCDFNavigationData (p. 68), LOAPEX::CFakeRXNavigationData (p. 39), LOAPEX::CFakeTXNavigationData (p. 42), and LOAPEX::CFixedNavigationData (p. 45).

6.6.4 Member Data Documentation

6.6.4.1 time_t LOAPEX::CBase3DNavigationData::file_time_
final_ [protected]

6.6.4.2 time_t LOAPEX::CBase3DNavigationData::file_time_
first_ [protected]

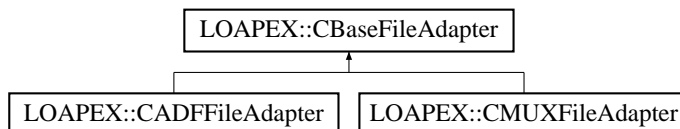
The documentation for this class was generated from the following files:

- navigation.hpp
- navigation.cpp

6.7 LOAPEX::CBaseFileAdapter Class Reference

```
#include <basefileadapter.hpp>
```

Inheritance diagram for LOAPEX::CBaseFileAdapter::



Public Member Functions

- virtual `~CBaseFileAdapter` (void)
- virtual void `get_scan` (std::vector< short > &V)=0
- virtual void `get_start_second` (long *s) const =0
- virtual void `get_total_scans` (long *t) const =0
- virtual void `get_total_channels` (int *c) const =0
- virtual void `get_header` (std::string &h) const =0

6.7.1 Detailed Description

A class that provides a pure virtual base class for input data files via an adaptor pattern.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 virtual LOAPEX::CBaseFileAdapter::~~CBaseFileAdapter (void) [inline, virtual]

6.7.3 Member Function Documentation

6.7.3.1 virtual void LOAPEX::CBaseFileAdapter::get_header (std::string & h) const [pure virtual]

Supplies the whole header via a string.

Implemented in `LOAPEX::CADFFFileAdapter` (p. 15), and `LOAPEX::CMUXFileAdapter` (p. 64).

6.7.3.2 virtual void LOAPEX::CBaseFileAdapter::get_scan
(std::vector< short > & V) [pure virtual]

Read the next "scan" of samples, one from each channel. Clears **std::vector**(p. 128) V first.

Implemented in **LOAPEX::CADFFFileAdapter** (p. 16), and **LOAPEX::CMUXFileAdapter** (p. 65).

6.7.3.3 virtual void LOAPEX::CBaseFileAdapter::get_start_second
(long * s) const [pure virtual]

Supplies the "start second" from the file (header)

Implemented in **LOAPEX::CADFFFileAdapter** (p. 16), and **LOAPEX::CMUXFileAdapter** (p. 65).

6.7.3.4 virtual void LOAPEX::CBaseFileAdapter::get_total_channels
(int * c) const [pure virtual]

Supplies the total number of channels, from the file (header)

Implemented in **LOAPEX::CADFFFileAdapter** (p. 16), and **LOAPEX::CMUXFileAdapter** (p. 65).

6.7.3.5 virtual void LOAPEX::CBaseFileAdapter::get_total_scans
(long * t) const [pure virtual]

Supplies the total number of channels, from the file (header)

Implemented in **LOAPEX::CADFFFileAdapter** (p. 16), and **LOAPEX::CMUXFileAdapter** (p. 65).

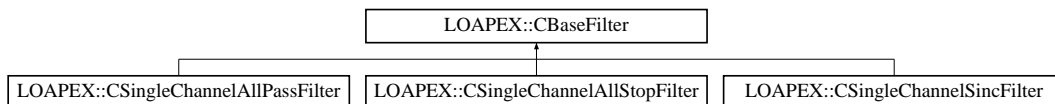
The documentation for this class was generated from the following file:

- **basefileadapter.hpp**

6.8 LOAPEX::CBaseFilter Class Reference

```
#include <basefilter.hpp>
```

Inheritance diagram for LOAPEX::CBaseFilter::



Public Member Functions

- **CBaseFilter** (const int channel_id)
- **~CBaseFilter** (void)
- virtual void **clock** (void)

Public Attributes

- int **channel_id_**
- LOAPEX::CDemultiplexor * **ptheDemux_**
- LOAPEX::CMultiplexor * **ptheMux_**

6.8.1 Detailed Description

A base class for filters that know about the demultiplexor (that's where they get their data) and the multiplexor (that's where they put their data). All derived classes must override the **clock()**(p. 28) method: this is the mechanism which causes the class to perform one cycle of its filter algorithm.

Turns out I needed to retain the notion of "channel" with each filter, so that in multichannel filters, each filter will know its own channel, in the (likely) case that the coefficients for each channel will be different. So for convenience, I made channel ID self knowledge available throughout all derived classes by putting it here.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 `LOAPEX::CBaseFilter::CBaseFilter (const int channel_id)`

6.8.2.2 `LOAPEX::CBaseFilter::~~CBaseFilter (void)`

6.8.3 Member Function Documentation

6.8.3.1 `void LOAPEX::CBaseFilter::clock (void) [virtual]`

Reimplemented in `LOAPEX::CSingleChannelAllPassFilter` (p. 74), `LOAPEX::CSingleChannelAllStopFilter` (p. 76), and `LOAPEX::CSingleChannelSincFilter` (p. 79).

6.8.4 Member Data Documentation

6.8.4.1 `int LOAPEX::CBaseFilter::channel_id_`

6.8.4.2 `LOAPEX::CDemultiplexor* LOAPEX::CBaseFilter::ptheDemux_`

6.8.4.3 `LOAPEX::CMultiplexor* LOAPEX::CBaseFilter::ptheMux_`

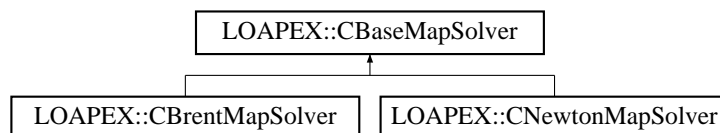
The documentation for this class was generated from the following files:

- `basefilter.hpp`
- `basefilter.cpp`

6.9 LOAPEX::CBaseMapSolver Class Reference

```
#include <basemapsolver.hpp>
```

Inheritance diagram for LOAPEX::CBaseMapSolver::



Public Member Functions

- `~CBaseMapSolver` (void)
- virtual void `load_parameters` (const `LOAPEX::CGeoParameters &P`)=0
- virtual void `load_TX` (`LOAPEX::CBase3DNavigationData *p`)=0
- virtual void `load_RX` (`LOAPEX::CBase3DNavigationData *p`)=0
- virtual void `compute_map` (const `LOAPEX::CCompoundTime &tin`, `LOAPEX::CCompoundTime &tout`)=0

Public Attributes

- `LOAPEX::CGeoParameters params_`
- `LOAPEX::CBase3DNavigationData * pTX_`
- `LOAPEX::CBase3DNavigationData * pRX_`

6.9.1 Detailed Description

`CBaseMapSolver`(p. 29) is intended to be the abstract base class for algorithms that solve the problem of computing $T(t)$, given t . The solver is abstracted because the workhorse routine is likely to use a line search method, but for testing a more accurate and computationally efficient Newton-Raphson scheme would be more appropriate. Hence, code that uses the solver(s) should be invariant to the choice of algorithm. (This is the "strategy pattern".)

6.9.2 Constructor & Destructor Documentation

6.9.2.1 LOAPEX::CBaseMapSolver::~~CBaseMapSolver (void)

6.9.3 Member Function Documentation

6.9.3.1 virtual void LOAPEX::CBaseMapSolver::compute_map (const LOAPEX::CCompoundTime & *tin*, LOAPEX::CCompoundTime & *tout*) [pure virtual]

Once the parameters and the TX and RX interfaces have been registered, this routine is the one that does one single T(t) computation.

Implemented in `LOAPEX::CBrentMapSolver` (p. 32), and `LOAPEX::CNewtonMapSolver` (p. 70).

6.9.3.2 virtual void LOAPEX::CBaseMapSolver::load_parameters (const LOAPEX::CGeoParameters & *P*) [pure virtual]

Provide an interface for copying parameter values into private storage.

Implemented in `LOAPEX::CBrentMapSolver` (p. 33), and `LOAPEX::CNewtonMapSolver` (p. 70).

6.9.3.3 virtual void LOAPEX::CBaseMapSolver::load_RX (LOAPEX::CBase3DNavigationData * *p*) [pure virtual]

Provide an interface for copying an interface for receiver position and velocity into private storage.

Implemented in `LOAPEX::CBrentMapSolver` (p. 33), and `LOAPEX::CNewtonMapSolver` (p. 70).

6.9.3.4 virtual void LOAPEX::CBaseMapSolver::load_TX (LOAPEX::CBase3DNavigationData * *p*) [pure virtual]

Provide an interface for copying an interface for transmitter position and velocity into private storage.

Implemented in `LOAPEX::CBrentMapSolver` (p. 33), and `LOAPEX::CNewtonMapSolver` (p. 70).

6.9.4 Member Data Documentation

6.9.4.1 LOAPEX::CGeoParameters LOAPEX::CBaseMapSolver::params_

6.9.4.2 LOAPEX::CBase3DNavigationData* LOAPEX::CBaseMapSolver::pRX_

6.9.4.3 LOAPEX::CBase3DNavigationData* LOAPEX::CBaseMapSolver::pTX_

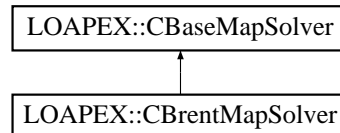
The documentation for this class was generated from the following files:

- **basemapsolver.hpp**
- **mapsolvers.cpp**

6.10 LOAPEX::CBrentMapSolver Class Reference

```
#include <mapsolvers.hpp>
```

Inheritance diagram for LOAPEX::CBrentMapSolver::



Public Member Functions

- **CBrentMapSolver** (void)
- **~CBrentMapSolver** (void)
- void **load_parameters** (const LOAPEX::CGeoParameters &P)
- void **load_TX** (LOAPEX::CBase3DNavigationData *p)
- void **load_RX** (LOAPEX::CBase3DNavigationData *p)
- void **compute_map** (const LOAPEX::CCompoundTime &tin, LOAPEX::CCompoundTime &tout)

6.10.1 Detailed Description

Uses the zbrent routine from Numerical Recipes as the root finder.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 LOAPEX::CBrentMapSolver::CBrentMapSolver (void)

6.10.2.2 LOAPEX::CBrentMapSolver::~~CBrentMapSolver (void)

6.10.3 Member Function Documentation

6.10.3.1 void LOAPEX::CBrentMapSolver::compute_map (const LOAPEX::CCompoundTime & tin, LOAPEX::CCompoundTime & tout) [virtual]

Once the parameters and the TX and RX interfaces have been registered, this routine is the one that does one single T(t) computation.

Implements LOAPEX::CBaseMapSolver (p. 30).

6.10.3.2 void LOAPEX::CBrentMapSolver::load_parameters
(const LOAPEX::CGeoParameters & *P*) [virtual]

Provide an interface for copying parameter values into private storage.

Implements LOAPEX::CBaseMapSolver (p. 30).

6.10.3.3 void LOAPEX::CBrentMapSolver::load_RX
(LOAPEX::CBase3DNavigationData * *p*) [virtual]

Provide an interface for copying an interface for receiver position and velocity into private storage.

Implements LOAPEX::CBaseMapSolver (p. 30).

6.10.3.4 void LOAPEX::CBrentMapSolver::load_TX
(LOAPEX::CBase3DNavigationData * *p*) [virtual]

Provide an interface for copying an interface for transmitter position and velocity into private storage.

Implements LOAPEX::CBaseMapSolver (p. 30).

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

- mapsolvers.hpp
- mapsolvers.cpp

6.11 LOAPEX::CCompoundTime Class Reference

```
#include <compoundtime.hpp>
```

Public Member Functions

- **CCompoundTime** (const long seconds, const double fraction)
- **CCompoundTime** (void)
- **~CCompoundTime** (void)
- void **normalize** (void)
- **CCompoundTime** & **operator+=** (**CCompoundTime** &)
- **CCompoundTime** & **operator-=** (**CCompoundTime** &)

Public Attributes

- long **seconds** _
- double **fraction** _

6.11.1 Detailed Description

Class to model a high resolution time object, comprised of a long, for unix seconds, and a double, for fractional seconds. Huge precision!

6.11.2 Constructor & Destructor Documentation

6.11.2.1 LOAPEX::CCompoundTime::CCompoundTime (const long *seconds*, const double *fraction*)

The ctor takes integer seconds, plus so fractional part, which can be greater than one or less than -1. The seconds are supposed to be Unix time (but I'm not sure the class cares.)

6.11.2.2 LOAPEX::CCompoundTime::CCompoundTime (void)

6.11.2.3 LOAPEX::CCompoundTime::~~CCompoundTime (void)

6.11.3 Member Function Documentation

6.11.3.1 void LOAPEX::CCompoundTime::normalize (void)

Readjust so that the fraction is in [0,1)

6.11.3.2 LOAPEX::CCompoundTime & LOAPEX::CCompoundTime::operator+= (CCompoundTime &)

6.11.3.3 LOAPEX::CCompoundTime & LOAPEX::CCompoundTime::operator-= (CCompoundTime &)

6.11.4 Member Data Documentation

6.11.4.1 double LOAPEX::CCompoundTime::fraction_

6.11.4.2 long LOAPEX::CCompoundTime::seconds_

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

- **compoundtime.hpp**
- **compoundtime.cpp**

6.12 LOAPEX::CDemultiplexor Class Reference

```
#include <demultiplexor.hpp>
```

Public Member Functions

- **CDemultiplexor** (**LOAPEX::CBaseFileAdapter** *A)
- **~CDemultiplexor** (void)
- void **send_sample** (const int channel, short *sample) const
- void **clock** (void)

6.12.1 Detailed Description

The demultiplexor class: contains an interface to a generic input device (base class **LOAPEX::CBaseFileAdapter**(p. 25)) and upon **clock()**(p. 36) reads a scan from the file and holds the scan data in internal storage. Classes holding the handle of this object can then get out a particular value by calling **send_sample()**(p. 36) which *sends* the sample to the calling routine.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 LOAPEX::CDemultiplexor::CDemultiplexor (LOAPEX::CBaseFileAdapter * A)

Ctor takes the pointer and stores it internally. Caller is responsible for deallocating it (AFTER the demultiplexor is done with it!)

6.12.2.2 LOAPEX::CDemultiplexor::~~CDemultiplexor (void)

6.12.3 Member Function Documentation

6.12.3.1 void LOAPEX::CDemultiplexor::clock (void)

This method is the one for the synchronization mechanism to call. Causes a new read from the input file.

6.12.3.2 void LOAPEX::CDemultiplexor::send_sample (const int *channel*, short * *sample*) const

This is the routine for external uses of the demux to use. Not sure if there is checking on the channel value. Supplies a 16 bit dataword.

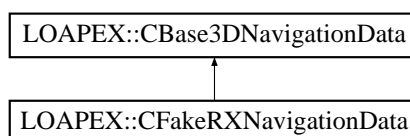
The documentation for this class was generated from the following files:

- **demultiplexor.hpp**
- **demultiplexor.cpp**

6.13 LOAPEX::CFakeRXNavigationData Class Reference

```
#include <navigation.hpp>
```

Inheritance diagram for LOAPEX::CFakeRXNavigationData::



Public Member Functions

- **CFakeRXNavigationData** (const int channel)
- **~CFakeRXNavigationData** (void)
- bool **is_loaded** (void) const
- void **get_file_first_time** (time_t *pt) const
- void **get_file_final_time** (time_t *pt) const
- void **get_position** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *X) const
- void **get_velocity** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *V) const

6.13.1 Detailed Description

Subclass for a fake class that outputs the position of a fake 3 channel receiver. The receiver is modeled as 3 sensors revolving in horizontal "orbits" one above the other. They all revolve synchronously. Think of a 3-phone vertical line array leaning over at some tilt angle and revolving around the vertical axis. All geometrical parameters hard coded in the include file. The start and stop times of the data (even though it is fake data) are hard coded here as well.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 LOAPEX::CFakeRXNavigationData::CFakeRXNavigationData (const int *channel*)

6.13.2.2 LOAPEX::CFakeRXNavigationData::~~CFakeRXNavigationData (void)

6.13.3 Member Function Documentation

6.13.3.1 void LOAPEX::CFakeRXNavigationData::get_file_final_time (time_t * *pt*) const [virtual]

Supplies the time of the last valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.13.3.2 void LOAPEX::CFakeRXNavigationData::get_file_first_time (time_t * *pt*) const [virtual]

Supplies the time of the first valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.13.3.3 void LOAPEX::CFakeRXNavigationData::get_position (const LOAPEX::CCompoundTime *t*, dEuclideanVector_t * *X*) const [virtual]

The evaluation time is the base_time (which is a time_t) plus and additional_seconds, which is a real, and can contain a fractional part.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.13.3.4 void LOAPEX::CFakeRXNavigationData::get_velocity (const LOAPEX::CCompoundTime *t*, dEuclideanVector_t * *V*) const [virtual]

The evaluation time is the base_time (which is a time_t) plus and additional_seconds, which is a real, and can contain a fractional part.

Implements **LOAPEX::CBase3DNavigationData** (p. 24).

6.13.3.5 `bool LOAPEX::CFakeRXNavigationData::is_loaded
(void) const [inline]`

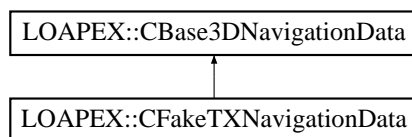
The documentation for this class was generated from the following files:

- `navigation.hpp`
- `navigation.cpp`

6.14 LOAPEX::CFakeTXNavigationData Class Reference

```
#include <navigation.hpp>
```

Inheritance diagram for LOAPEX::CFakeTXNavigationData:



Public Member Functions

- **CFakeTXNavigationData** (void)
- **~CFakeTXNavigationData** (void)
- **bool is_loaded** (void) const
- **void get_file_first_time** (time_t *pt) const
- **void get_file_final_time** (time_t *pt) const
- **void get_position** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *X) const
- **void get_velocity** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *V) const

6.14.1 Detailed Description

Subclass for a fake class that outputs the position of a fake single channel transmitter. The transmitter is modeled as revolving in a horizontal "orbit". All geometrical parameters hard coded in the include file. The start and stop times of the data (even though it is fake data) are hard coded here as well.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 `LOAPEX::CFakeTXNavigationData::CFakeTXNavigationData (void)`

6.14.2.2 `LOAPEX::CFakeTXNavigationData::~~CFakeTXNavigationData (void)`

6.14.3 Member Function Documentation

6.14.3.1 `void LOAPEX::CFakeTXNavigationData::get_file_final_time (time_t * pt) const [virtual]`

Supplies the time of the last valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements `LOAPEX::CBase3DNavigationData` (p. 23).

6.14.3.2 `void LOAPEX::CFakeTXNavigationData::get_file_first_time (time_t * pt) const [virtual]`

Supplies the time of the first valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements `LOAPEX::CBase3DNavigationData` (p. 23).

6.14.3.3 `void LOAPEX::CFakeTXNavigationData::get_position (const LOAPEX::CCompoundTime t, dEuclideanVector_t * X) const [virtual]`

The evaluation time is the base_time (which is a time_t) plus and additional_seconds, which is a real, and can contain a fractional part.

Implements `LOAPEX::CBase3DNavigationData` (p. 23).

6.14.3.4 `void LOAPEX::CFakeTXNavigationData::get_velocity (const LOAPEX::CCompoundTime t, dEuclideanVector_t * V) const [virtual]`

The evaluation time is the base_time (which is a time_t) plus and additional_seconds, which is a real, and can contain a fractional part.

Implements **LOAPEX::CBase3DNavigationData** (p. 24).

6.14.3.5 bool LOAPEX::CFakeTXNavigationData::is_loaded (void) const [inline]

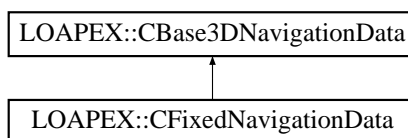
The documentation for this class was generated from the following files:

- **navigation.hpp**
- **navigation.cpp**

6.15 LOAPEX::CFixedNavigationData Class Reference

```
#include <navigation.hpp>
```

Inheritance diagram for LOAPEX::CFixedNavigationData:



Public Member Functions

- **CFixedNavigationData** (const int channel)
- **~CFixedNavigationData** (void)
- bool **is_loaded** (void) const
- void **get_file_first_time** (time_t *pt) const
- void **get_file_final_time** (time_t *pt) const
- void **get_position** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *X) const
- void **get_velocity** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *V) const

6.15.1 Detailed Description

Subclass for a fake class that is stationary File start and stop times are the same as the fake RX and TX classes,

6.15.2 Constructor & Destructor Documentation

6.15.2.1 LOAPEX::CFixedNavigationData::CFixedNavigationData (const int *channel*)

6.15.2.2 LOAPEX::CFixedNavigationData::~~CFixedNavigationData (void)

6.15.3 Member Function Documentation

6.15.3.1 void LOAPEX::CFixedNavigationData::get_
file_final_time (time_t * *pt*) const
[virtual]

Supplies the time of the last valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.15.3.2 void LOAPEX::CFixedNavigationData::get_
file_first_time (time_t * *pt*) const
[virtual]

Supplies the time of the first valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.15.3.3 void LOAPEX::CFixedNavigationData::get_position
(const LOAPEX::CCompoundTime *t*, dEuclideanVector_t
* *X*) const [virtual]

The evaluation time is the base_time (which is a time_t) plus and additional_seconds, which is a real, and can contain a fractional part.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.15.3.4 void LOAPEX::CFixedNavigationData::get_velocity
(const LOAPEX::CCompoundTime *t*, dEuclideanVector_t
* *V*) const [virtual]

The evaluation time is the base_time (which is a time_t) plus and additional_seconds, which is a real, and can contain a fractional part.

Implements **LOAPEX::CBase3DNavigationData** (p. 24).

6.15.3.5 `bool LOAPEX::CFixedNavigationData::is_loaded (void)`
`const [inline]`

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

- **navigation.hpp**
- **navigation.cpp**

6.16 LOAPEX::CGappyADFStream Class Reference

```
#include <gappystream.hpp>
```

Public Member Functions

- **CGappyADFStream** (const long *sstart*, const long *sstop*, std::list< std::string > &*L*)
- **~CGappyADFStream** (void)
- void **get_scan** (std::vector< short > &*V*)
- void **get_start_second** (long **lstart*) const
- void **get_total_channels** (int **tc*) const
- void **get_total_scans** (long **ts*) const
- void **get_sample_rate** (int **fs*) const
- void **print** (void)

6.16.1 Detailed Description

This is a class that contains a list of files and pretends that they comprise, with possible zero padding in between, one big long file. Pulling a scan from the class gets the next scan: this may contain actual data if there is a file in the list whos time span includes the next scan, otherwise one gets a scan of zeros. This class was devised to "glue" Scripps burst collections together in way that retained the temporal relationship between them through intervening gaps. (This is made easy because the sample rate was a whole integer, meaning there are a whole number of zero pad scans each second in between real data.)

The public interface is meant to mimic a real .adf file, probably so that the .adf file adapter could be used on this class as well as real .adf files.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 LOAPEX::CGappyADFStream::CGappyADFStream (const long *sstart*, const long *sstop*, std::list< std::string > & *L*)

Takes a list of .adf file names, and user selected start and stop times. Makes zero source objects to fill the gaps in between the supplied files. After construction, one can pull scans from the object.

6.16.2.2 LOAPEX::CGappyADFStream::~~CGappyADFStream
(void)

6.16.3 Member Function Documentation

6.16.3.1 void LOAPEX::CGappyADFStream::get_sample_rate
(int * fs) const

Supplies the sample rate, from the file (header). Note that we assume an integer sample rate. This has generally held true for all ATOC hardware influenced by Kurt Metzger, but of course will not always be true in the future.

6.16.3.2 void LOAPEX::CGappyADFStream::get_scan
(std::vector< short > & V)

6.16.3.3 void LOAPEX::CGappyADFStream::get_start_second
(long * lstart) const

6.16.3.4 void LOAPEX::CGappyADFStream::get_total_channels
(int * tc) const

Supplies the total number of sensors, from the file (header)

6.16.3.5 void LOAPEX::CGappyADFStream::get_total_scans
(long * ts) const

Supplies the total number of scans, from the file (header)

6.16.3.6 void LOAPEX::CGappyADFStream::print (void)

Diagnostic dump of member data.

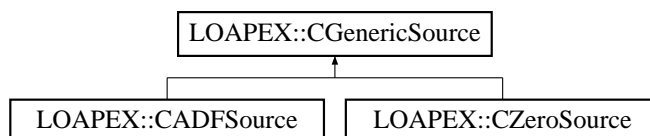
The documentation for this class was generated from the following files:

- gappystream.hpp
- gappystream.cpp

6.17 LOAPEX::CGenericSource Class Reference

```
#include <gappystream.hpp>
```

Inheritance diagram for LOAPEX::CGenericSource::



Public Member Functions

- `~CGenericSource` (void)
- virtual void `get_scan` (std::vector< short > &V)=0
- virtual void `get_start_second` (long *s) const =0
- virtual void `get_total_scans` (long *t) const =0
- virtual void `get_total_channels` (int *c) const =0
- virtual void `get_sample_rate` (int *fs) const =0
- virtual void `get_type` (char *c) const =0

6.17.1 Detailed Description

Base class for the types of sources we could have. These are lightweight classes, just doing the bare minimum of modeling of actual source files (and being really bare if modeling a source of zeros.) These are primarily for sticking into a std::list().

6.17.2 Constructor & Destructor Documentation

6.17.2.1 LOAPEX::CGenericSource::~~CGenericSource (void)

6.17.3 Member Function Documentation

6.17.3.1 virtual void LOAPEX::CGenericSource::get_sample_rate (int * fs) const [pure virtual]

Implemented in `LOAPEX::CZeroSource` (p. 87), and `LOAPEX::CADFSource` (p. 20).

6.17.3.2 virtual void LOAPEX::CGenericSource::get_scan
(std::vector< short > & V) [pure virtual]

Implemented in LOAPEX::CZeroSource (p. 88), and
LOAPEX::CADFSource (p. 20).

6.17.3.3 virtual void LOAPEX::CGenericSource::get_start_second
(long * s) const [pure virtual]

Implemented in LOAPEX::CZeroSource (p. 88), and
LOAPEX::CADFSource (p. 20).

6.17.3.4 virtual void LOAPEX::CGenericSource::get_-
total_channels (int * c) const [pure
virtual]

Implemented in LOAPEX::CZeroSource (p. 88), and
LOAPEX::CADFSource (p. 20).

6.17.3.5 virtual void LOAPEX::CGenericSource::get_total_scans
(long * t) const [pure virtual]

Implemented in LOAPEX::CZeroSource (p. 88), and
LOAPEX::CADFSource (p. 20).

6.17.3.6 virtual void LOAPEX::CGenericSource::get_type (char *
c) const [pure virtual]

Implemented in LOAPEX::CZeroSource (p. 88), and
LOAPEX::CADFSource (p. 21).

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

- gappystream.hpp
- gappystream.cpp

6.18 LOAPEX::CGeoParameters Class Reference

```
#include <parameters.hpp>
```

Public Attributes

- double `mean_sound_speed`
- double `range_m`
- double `forward_bearing2VLA`
- double `forward_bearing2APL`
- int `channel`
- `dEuclideanVector_t` `kTX`
- `dEuclideanVector_t` `kRX`

6.18.1 Detailed Description

Simple encapsulation of doppler context user parameters. This is essentially a C structure with an initializer.

6.18.2 Member Data Documentation

6.18.2.1 int LOAPEX::CGeoParameters::channel

Channel identifier. I think this is 0 to N-1.

6.18.2.2 double LOAPEX::CGeoParameters::forward_bearing2APL

Compass bearing from the receiver to the transmitter, in degrees.

6.18.2.3 double LOAPEX::CGeoParameters::forward_bearing2VLA

Compass bearing from the transmitter to the receiver, in degrees.

6.18.2.4 `dEuclideanVector_t` LOAPEX::CGeoParameters::kRX

This is the incident wavenumber vector at the receiver. I think it is the unit vector.

6.18.2.5 `dEuclideanVector_t LOAPEX::CGeoParameters::kTX`

This is the wavenumber launch vector at the transmitter. I think it is the unit vector.

6.18.2.6 `double LOAPEX::CGeoParameters::mean_sound_speed`

Sound speed, meters per second.

6.18.2.7 `double LOAPEX::CGeoParameters::range_m`

Range in metres.

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

- `parameters.hpp`

6.19 LOAPEX::CInterpolatedTableFunction Class Reference

```
#include <tablefunction.hpp>
```

Public Member Functions

- **CInterpolatedTableFunction** (const std::vector< double > &x, const std::vector< double > &y)
- **~CInterpolatedTableFunction** (void)
- void **evaluate** (const double x, double *y) const
- void **get_table_length** (long *L) const

6.19.1 Detailed Description

These classes interface spline code to a file containing a Nx2 table of values, with the intent of making the tableness of the data transparent to the caller.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 LOAPEX::CInterpolatedTableFunction::CInterpolatedTableFunction (const std::vector< double > & x, const std::vector< double > & y)

The ctor builds the spline coefficients. Uses the spline routine from Numerical Recipes in C, 2nd ed. The table data are (x[0],y[0]), (x[1],y[1]), etc.....

6.19.2.2 LOAPEX::CInterpolatedTableFunction::~~CInterpolatedTableFunction (void)

6.19.3 Member Function Documentation

6.19.3.1 void LOAPEX::CInterpolatedTableFunction::evaluate (const double x, double * y) const

Method provides $y(x)$ for arbitrary x . You're on your own if you ask for x outside the domain of the supplied vector x .

6.19.3.2 void LOAPEX::CInterpolatedTableFunction::get_table_length (long * *L*) const

Gets the length of the underlying table. I have no idea why I exposed this.

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

- **tablefunction.hpp**
- **tablefunction.cpp**

6.20 LOAPEX::CMultichannelFilter Class Reference

```
#include <multichannelfilter.hpp>
```

Public Member Functions

- **CMultichannelFilter** (const int total_channels)
- **~CMultichannelFilter** (void)
- void **clock** (void)
- void **add_filter** (LOAPEX::CBaseFilter *p)

6.20.1 Detailed Description

Concrete class that simply is a container for an array of of single channel filter objects. Note that the handles to the demultiplexor and the multiplexor reside in the filters: this class does not know about the demux and mux. Design decision: could have gone the other way too.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 LOAPEX::CMultichannelFilter::CMultichannelFilter (const int *total_channels*)

Ctor simply allocates storage for the single channel filter pointers.

6.20.2.2 LOAPEX::CMultichannelFilter::~~CMultichannelFilter (void)

6.20.3 Member Function Documentation

6.20.3.1 void LOAPEX::CMultichannelFilter::add_filter (LOAPEX::CBaseFilter * *p*)

Helper function that copies the pointer into internal storage. I don't know what happens if the user adds too many, or not enough. Don't do that. User responsible for deallocating the filter objects at end of use.

6.20.3.2 void LOAPEX::CMultichannelFilter::clock (void)

This method simply applies **clock()**(p. 56) to each element of the internal array of filters.

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

- **multichannelfilter.hpp**
- **multichannelfilter.cpp**

6.21 LOAPEX::CMultiplexor Class Reference

```
#include <multiplexor.hpp>
```

Public Member Functions

- **CMultiplexor** (LOAPEX::CMUXFile *F)
- **~CMultiplexor** (void)
- void **receive_sample** (const int channel, const short sample)
- void **clock** (void)

6.21.1 Detailed Description

The multiplexor class: contains an interface to a MUX file and upon **clock()**(p. 57) writes a scan from internal storage to the file. Classes holding the handle of this object write data into the internal storage via the **receive_sample()**(p. 57) method, which means that this object is "receiving" something from the caller.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 LOAPEX::CMultiplexor::CMultiplexor (LOAPEX::CMUXFile * F)

Ctor takes a pointer to an existing MUX file and stores it internally. Caller is responsible for deallocating it (AFTER the multiplexor is done with it!)

6.21.2.2 LOAPEX::CMultiplexor::~~CMultiplexor (void)

6.21.3 Member Function Documentation

6.21.3.1 void LOAPEX::CMultiplexor::clock (void)

This method is the one for the synchronization mechanism to call. Causes a new write to the output file.

6.21.3.2 void LOAPEX::CMultiplexor::receive_sample (const int *channel*, const short *sample*)

This is the routine for external users of the mux to use. Not sure if there is checking on the channel value. Puts a 16-bit dataword into the object.

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

- `multiplexor.hpp`
- `multiplexor.cpp`

6.22 LOAPEX::CMUXFile Class Reference

```
#include <muxfile.hpp>
```

Public Types

- enum `mode_type` { `ReadOnly`, `Write` }

Public Member Functions

- `CMUXFile` (const char *name, `mode_type` m)
- `~CMUXFile` (void)
- void `close` (void)
- void `print` (void)
- bool `is_open` (void) const
- void `put_sample_rate` (int fs)
- void `put_total_scans` (long L)
- void `put_total_channels` (int c)
- void `put_start_second` (long s)
- void `put_description` (std::string &s)
- void `write_header` (void)
- void `put_scan` (std::vector< short > &V)
- bool `is_loaded` (void) const
- void `get_sample_rate` (int *fs) const
- void `get_total_scans` (long *L) const
- void `get_total_channels` (int *c) const
- void `get_start_second` (long *s) const
- void `get_descriptions` (std::vector< std::string const * > &V)
- void `get_scan` (std::vector< short > &V)
- void `get_header` (std::string &h) const

6.22.1 Detailed Description

Class `CMUXFile`(p. 59) describes a very simple scan-oriented file for use with `LOAPEX`(p. 10) data. Such a file has the format

```
muxfile
{
    header
    binary data
}
```

This class interfaces either to an existing file (using mode `LOAPEX::CMUXFile::ReadOnly`) or open a new file for writing (using `LOAPEX::CMUXFile::Write`). Methods are provided to read the basic parameters from the header, or to write the basic parameters into the header. These basic parameters are:

```

sample rate (integer)
start-second (long)
total scans (long)
total channels (integer)
an arbitrary amount of line-oriented description material

```

The header is written in XML.

There are also the methods `get_scan()`(p. 61) for reading one scan at a time, or `put_scan()`(p. 62) for writing one scan at a time. The binary data is treated like a stream: no seeking implemented.

6.22.2 Member Enumeration Documentation

6.22.2.1 enum `LOAPEX::CMUXFile::mode_type`

Enumeration values:

ReadOnly

Write

6.22.3 Constructor & Destructor Documentation

6.22.3.1 `LOAPEX::CMUXFile::CMUXFile (const char * name, mode_type m)`

Ctor takes a C string name and a mode indicating whether we are creating a new file for writing or opening an existing file reading.

6.22.3.2 `LOAPEX::CMUXFile::~~CMUXFile (void)`

6.22.4 Member Function Documentation

6.22.4.1 `void LOAPEX::CMUXFile::close (void)`

Simple pass through to C lib `close()`(p. 60).

6.22.4.2 void LOAPEX::CMUXFile::get_descriptions (std::vector< std::string const * > & V)

Gets the descriptions from internal storage. Looks like there could be more than one, so they come back in a vector of strings.

6.22.4.3 void LOAPEX::CMUXFile::get_header (std::string & h) const

Gets the entire XML header as a single string.

6.22.4.4 void LOAPEX::CMUXFile::get_sample_rate (int * fs) const

Gets the sample rate from internal storage.

6.22.4.5 void LOAPEX::CMUXFile::get_scan (std::vector< short > & V)

Gets a vector of 16 bit datawords.

6.22.4.6 void LOAPEX::CMUXFile::get_start_second (long * s) const

Gets the start second from internal storage.

6.22.4.7 void LOAPEX::CMUXFile::get_total_channels (int * c) const

Gets the total number of channels from internal storage.

6.22.4.8 void LOAPEX::CMUXFile::get_total_scans (long * L) const

Gets the total number of scans from internal storage.

6.22.4.9 bool LOAPEX::CMUXFile::is_loaded (void) const

Returns true if the file was opened and the header parsed successfully.

6.22.4.10 `bool LOAPEX::CMUXFile::is_open (void) const`

Returns true if the file open was successful, false if not.

6.22.4.11 `void LOAPEX::CMUXFile::print (void)`

Diagnostic: dumps the header (I think.)

6.22.4.12 `void LOAPEX::CMUXFile::put_description (std::string & s)`

Put the given string into the header. (This does not write anything to file. See `writer_header()`.) XML tag is `<description>`. I think this can be called multiple times, which would result in multiple `<description></description>` pairs in the file header.

6.22.4.13 `void LOAPEX::CMUXFile::put_sample_rate (int fs)`

Put the given sample rate into the header. (This does not write anything to file. See `writer_header()`.) XML tag is `<sample-rate>`.

6.22.4.14 `void LOAPEX::CMUXFile::put_scan (std::vector< short > & V)`

This writes a vector of 16 bit data words to the file.

6.22.4.15 `void LOAPEX::CMUXFile::put_start_second (long s)`

Put the given starting second into the header. (This does not write anything to file. See `writer_header()`.) XML tag is `<start-second>`.

6.22.4.16 `void LOAPEX::CMUXFile::put_total_channels (int c)`

Put the given total number of channels into the header. (This does not write anything to file. See `writer_header()`.) XML tag is `<total-channels>`.

6.22.4.17 `void LOAPEX::CMUXFile::put_total_scans (long L)`

Put the given total number of scans into the header. (This does not write anything to file. See `writer_header()`.) XML tag is `<total-scans>`

6.22.4.18 void LOAPEX::CMUXFile::write_header (void)

This causes a bunch of XML to be written to the file on disk, in principle forming the file header. Should be called once before any data is written to the disk.

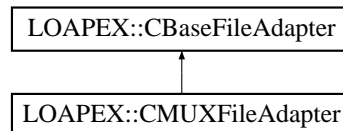
The documentation for this class was generated from the following files:

- **muxfile.hpp**
- **muxfile.cpp**

6.23 LOAPEX::CMUXFileAdapter Class Reference

```
#include <muxadapter.hpp>
```

Inheritance diagram for LOAPEX::CMUXFileAdapter::



Public Member Functions

- **CMUXFileAdapter** (LOAPEX::CMUXFile *p)
- **~CMUXFileAdapter** (void)
- void **get_scan** (std::vector< short > &V)
- void **get_start_second** (long *s) const
- void **get_total_scans** (long *t) const
- void **get_total_channels** (int *c) const
- void **get_sample_rate** (int *fs) const
- void **get_header** (std::string &h) const

6.23.1 Detailed Description

Object adaptor class for my .mux files

6.23.2 Constructor & Destructor Documentation

6.23.2.1 LOAPEX::CMUXFileAdapter::CMUXFileAdapter (LOAPEX::CMUXFile * p)

6.23.2.2 LOAPEX::CMUXFileAdapter::~~CMUXFileAdapter (void)

6.23.3 Member Function Documentation

6.23.3.1 void LOAPEX::CMUXFileAdapter::get_header (std::string & h) const [virtual]

Supplies the whole header via a string.

Implements `LOAPEX::CBaseFileAdapter` (p. 25).

6.23.3.2 `void LOAPEX::CMUXFileAdapter::get_sample_rate (int * fs) const`

6.23.3.3 `void LOAPEX::CMUXFileAdapter::get_scan (std::vector< short > & V) [virtual]`

Read the next "scan" of samples, one from each channel. Clears `std::vector`(p. 128) V first.

Implements `LOAPEX::CBaseFileAdapter` (p. 26).

6.23.3.4 `void LOAPEX::CMUXFileAdapter::get_start_second (long * s) const [virtual]`

Supplies the "start second" from the file (header)

Implements `LOAPEX::CBaseFileAdapter` (p. 26).

6.23.3.5 `void LOAPEX::CMUXFileAdapter::get_total_channels (int * c) const [virtual]`

Supplies the total number of channels, from the file (header)

Implements `LOAPEX::CBaseFileAdapter` (p. 26).

6.23.3.6 `void LOAPEX::CMUXFileAdapter::get_total_scans (long * t) const [virtual]`

Supplies the total number of channels, from the file (header)

Implements `LOAPEX::CBaseFileAdapter` (p. 26).

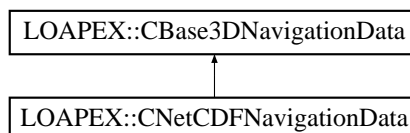
The documentation for this class was generated from the following files:

- `muxadapter.hpp`
- `muxadapter.cpp`

6.24 LOAPEX::CNetCDFNavigationData Class Reference

```
#include <navigation.hpp>
```

Inheritance diagram for LOAPEX::CNetCDFNavigationData:



Public Member Functions

- **CNetCDFNavigationData** (const char *filename, const int channel)
- **~CNetCDFNavigationData** (void)
- **bool is_loaded** (void) const
- **void get_file_first_time** (time_t *pt) const
- **void get_file_final_time** (time_t *pt) const
- **void get_position** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *X) const
- **void get_velocity** (const LOAPEX::CCompoundTime t, d-EuclideanVector_t *V) const

6.24.1 Detailed Description

Subclass for a netCDF file format. I made such files based on Mike Zarnetske's .mat files, but I needed something I could read beyond Matlab. The CDL for an example file is:

```
netcdf t1000z350 {
dimensions:
    time = 128213 ;
    singleton = 1 ;
variables:
    double depth(singleton) ;
        depth:units = "metres" ;
    double x(time) ;
        x:units = "metres, east/west" ;
    double y(time) ;
        y:units = "metres, north/south" ;
    double z(time) ;
        z:units = "metres, up/down (up is positive)" ;
    double u(time) ;
        u:units = "metres/second, east/west" ;
```

```

double v(time) ;
    v:units = "metres/second, north/south" ;
double w(time) ;
    w:units = "metres/second, up/down (up is positive)" ;
int t(time) ;
    t:units = "seconds since Jan 1 1970 00:00:00" ;

// global attributes:
    :station = "T1000" ;
    :description = "VLFP position solution, 2004 LOAPEX Cruise" ;
    :note = "Converted from file t1000z350f.mat" ;
    :conversiondate = "2006/10/28" ;
    :author = "RKAndrew, APL/UW, randrew@apl.washington.edu" ;
}

```

6.24.2 Constructor & Destructor Documentation

6.24.2.1 LOAPEX::CNetCDFNavigationData::CNetCDFNavigationData (const char * *filename*, const int *channel*)

Ctor takes the filename as a C string, and a channel. This class only handles ne navigable object. If there are multiple channels, (i.e., multiple tables of position and velocity) I suppose one would need multiple instantiations of this class.

6.24.2.2 LOAPEX::CNetCDFNavigationData::~~CNetCDFNavigationData (void)

6.24.3 Member Function Documentation

6.24.3.1 void LOAPEX::CNetCDFNavigationData::get_file_final_time (time_t * *pt*) const [virtual]

Supplies the time of the last valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements `LOAPEX::CBase3DNavigationData` (p. 23).

6.24.3.2 void LOAPEX::CNetCDFNavigationData::get_file_first_time (time_t * *pt*) const [virtual]

Supplies the time of the first valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements **LOAPEX::CBase3DNavigationData** (p. 23).

6.24.3.3 `void LOAPEX::CNetCDFNavigationData::get_position
(const LOAPEX::CCompoundTime t, dEuclideanVector_t
* X) const [virtual]`

The evaluation time is the base_time (which is a time_t) plus and additional_seconds, which is a real, and can contain a fractional part.

Implements **LOAPEX::CBase3DNavigationData** (p. 23).

6.24.3.4 `void LOAPEX::CNetCDFNavigationData::get_velocity
(const LOAPEX::CCompoundTime t, dEuclideanVector_t
* V) const [virtual]`

The evaluation time is the base_time (which is a time_t) plus and additional_seconds, which is a real, and can contain a fractional part.

Implements **LOAPEX::CBase3DNavigationData** (p. 24).

6.24.3.5 `bool LOAPEX::CNetCDFNavigationData::is_loaded
(void) const [inline]`

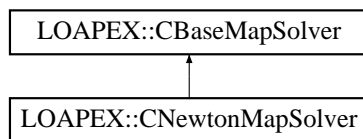
The documentation for this class was generated from the following files:

- **navigation.hpp**
- **navigation.cpp**

6.25 LOAPEX::CNewtonMapSolver Class Reference

```
#include <mapsolvers.hpp>
```

Inheritance diagram for LOAPEX::CNewtonMapSolver:



Public Member Functions

- **CNewtonMapSolver** (void)
- **~CNewtonMapSolver** (void)
- void **load_parameters** (const **LOAPEX::CGeoParameters** &P)
- void **load_TX** (**LOAPEX::CBase3DNavigationData** *p)
- void **load_RX** (**LOAPEX::CBase3DNavigationData** *p)
- void **compute_map** (const **LOAPEX::CCompoundTime** &tin, **LOAPEX::CCompoundTime** &tout)

6.25.1 Detailed Description

Uses a Newton-Raphson solver, from Numerical Recipes, for solving the root finding problem by optimizing to the minimum of the squared error. Requires an analytic expression for position and velocity versus time, which is only the case for theoretical test scenarios. Hence, this class is only used for testing, and comparing results to the **CBrentMapSolver**(p. 32) class.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 `LOAPEX::CNewtonMapSolver::CNewtonMapSolver`
(void)

6.25.2.2 `LOAPEX::CNewtonMapSolver::~~CNewtonMapSolver`
(void)

6.25.3 Member Function Documentation

6.25.3.1 `void LOAPEX::CNewtonMapSolver::compute_map`
(const LOAPEX::CCompoundTime & *tin*,
LOAPEX::CCompoundTime & *tout*) [virtual]

Once the parameters and the TX and RX interfaces have been registered, this routine is the one that does one single T(t) computation.

Implements `LOAPEX::CBaseMapSolver` (p. 30).

6.25.3.2 `void LOAPEX::CNewtonMapSolver::load_parameters`
(const LOAPEX::CGeoParameters & *P*) [virtual]

Provide an interface for copying parameter values into private storage.

Implements `LOAPEX::CBaseMapSolver` (p. 30).

6.25.3.3 `void LOAPEX::CNewtonMapSolver::load_RX`
(LOAPEX::CBase3DNavigationData * *p*) [virtual]

Provide an interface for copying an interface for receiver position and velocity into private storage.

Implements `LOAPEX::CBaseMapSolver` (p. 30).

6.25.3.4 `void LOAPEX::CNewtonMapSolver::load_TX`
(LOAPEX::CBase3DNavigationData * *p*) [virtual]

Provide an interface for copying an interface for transmitter position and velocity into private storage.

Implements `LOAPEX::CBaseMapSolver` (p. 30).

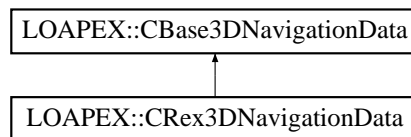
The documentation for this class was generated from the following files:

- `mapsolvers.hpp`
- `mapsolvers.cpp`

6.26 LOAPEX::CRex3DNavigationData Class Reference

```
#include <navigation.hpp>
```

Inheritance diagram for LOAPEX::CRex3DNavigationData:



Public Member Functions

- **CRex3DNavigationData** (const char *filename)
- **~CRex3DNavigationData** (void)
- bool **is_loaded** (void) const
- void **get_file_first_time** (time_t *pt) const
- void **get_file_final_time** (time_t *pt) const
- void **get_position** (const **LOAPEX::CCompoundTime** t, **d-EuclideanVector_t** *X) const
- void **get_velocity** (const **LOAPEX::CCompoundTime** t, **d-EuclideanVector_t** *V) const

6.26.1 Detailed Description

This was total testing stuff. Ignore.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 LOAPEX::CRex3DNavigationData::CRex3DNavigationData (const char * *filename*)

6.26.2.2 LOAPEX::CRex3DNavigationData::~~CRex3DNavigationData (void)

6.26.3 Member Function Documentation

6.26.3.1 void LOAPEX::CRex3DNavigationData::get_file_final_time (time_t * *pt*) const [virtual]

Supplies the time of the last valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.26.3.2 void LOAPEX::CRex3DNavigationData::get_file_first_time (time_t * *pt*) const [virtual]

Supplies the time of the first valid position (or vecocity) sample in the file. Looks like this is a time_t, so we are assuming Unix time, and the first sample must be tied to a unique whole second.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.26.3.3 void LOAPEX::CRex3DNavigationData::get_position (const LOAPEX::CCompoundTime *t*, dEuclideanVector_t * *X*) const [virtual]

Given a compound time, retrieves the position at that time. Derived classes will have to figure out what to do it there is no position at that exact time.

Implements LOAPEX::CBase3DNavigationData (p. 23).

6.26.3.4 void LOAPEX::CRex3DNavigationData::get_velocity (const LOAPEX::CCompoundTime *t*, dEuclideanVector_t * *V*) const [virtual]

Given a compound time, retrieves the velocity at that time. Derived classes will have to figure out what to do it there is no velocity at that exact time.

Implements **LOAPEX::CBase3DNavigationData** (p. 24).

6.26.3.5 `bool LOAPEX::CRex3DNavigationData::is_loaded (void)`
`const [inline]`

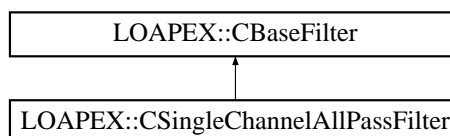
The documentation for this class was generated from the following file:

- **navigation.hpp**

6.27 LOAPEX::CSingleChannelAllPassFilter Class Reference

```
#include <allpass.hpp>
```

Inheritance diagram for LOAPEX::CSingleChannelAllPassFilter::



Public Member Functions

- **CSingleChannelAllPassFilter** (const int channel)
- **~CSingleChannelAllPassFilter** (void)
- void **clock** (void)
- void **set_demultiplexor** (LOAPEX::CDemultiplexor *p)
- void **set_multiplexor** (LOAPEX::CMultiplexor *p)

6.27.1 Detailed Description

Implements a filter that simply clocks the input data to the output. I.e., all pass. Largely for testing.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 LOAPEX::CSingleChannelAllPassFilter::CSingleChannelAllPassFilter (const int *channel*)

6.27.2.2 LOAPEX::CSingleChannelAllPassFilter::~~CSingleChannelAllPassFilter (void)

6.27.3 Member Function Documentation

6.27.3.1 void LOAPEX::CSingleChannelAllPassFilter::clock (void) [virtual]

Causes the filter to step once.

Reimplemented from LOAPEX::CBaseFilter (p. 28).

6.27.3.2 void LOAPEX::CSingleChannelAllPassFilter::set_
demultiplexor (LOAPEX::CDemultiplexor *
p)

Copies the handle to the demux into internal storage. It is the user's responsibility to free the demux after the class is destroyed.

6.27.3.3 void LOAPEX::CSingleChannelAllPass-
Filter::set_
multiplexor (LOAPEX::CMultiplexor *
p)

Copies the handle to the mux into internal storage. It is the user's responsibility to free the emux after the class is destroyed.

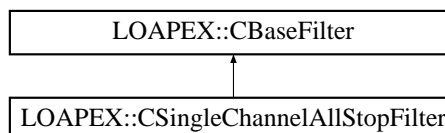
The documentation for this class was generated from the following files:

- **allpass.hpp**
- **allpass.cpp**

6.28 LOAPEX::CSingleChannelAllStopFilter Class Reference

```
#include <allstop.hpp>
```

Inheritance diagram for LOAPEX::CSingleChannelAllStopFilter::



Public Member Functions

- **CSingleChannelAllStopFilter** (const int channel)
- **~CSingleChannelAllStopFilter** (void)
- void **clock** (void)
- void **set_demultiplexor** (LOAPEX::CDemultiplexor *p)
- void **set_multiplexor** (LOAPEX::CMultiplexor *p)

6.28.1 Detailed Description

Implements a filter that simply clocks out zeros regardless of input. I.e., all stop. Largely for testing.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 LOAPEX::CSingleChannelAllStopFilter::CSingleChannelAllStopFilter (const int *channel*)

6.28.2.2 LOAPEX::CSingleChannelAllStopFilter::~~CSingleChannelAllStopFilter (void)

6.28.3 Member Function Documentation

6.28.3.1 void LOAPEX::CSingleChannelAllStopFilter::clock (void) [virtual]

Causes the filter to step once.

Reimplemented from LOAPEX::CBaseFilter (p. 28).

6.28.3.2 void LOAPEX::CSingleChannelAllStopFilter::set_
demultiplexor (LOAPEX::CDemultiplexor *
p)

Copies the handle to the demux into internal storage. It is the user's responsibility to free the demux after the class is destroyed.

6.28.3.3 void LOAPEX::CSingleChannelAllStop-
Filter::set_
multiplexor (LOAPEX::CMultiplexor *
p)

Copies the handle to the mux into internal storage. It is the user's responsibility to free the emux after the class is destroyed.

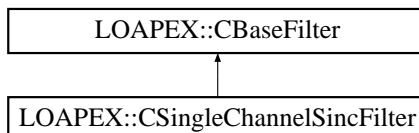
The documentation for this class was generated from the following files:

- **allstop.hpp**
- **allstop.cpp**

6.29 LOAPEX::CSingleChannelSincFilter Class Reference

```
#include < sincfilter.hpp >
```

Inheritance diagram for LOAPEX::CSingleChannelSincFilter::



Public Member Functions

- **CSingleChannelSincFilter** (const int channel, const int order, const double sample_time)
- **~CSingleChannelSincFilter** (void)
- void **clock** (void)
- void **set_demultiplexor** (LOAPEX::CDemultiplexor *p)
- void **set_multiplexor** (LOAPEX::CMultiplexor *p)
- void **set_mapper** (LOAPEX::CTimeBaseMapEngine *p)
- void **set_rx_start_time** (const time_t t)
- void **enable_logging** (void)
- void **disable_logging** (void)

6.29.1 Constructor & Destructor Documentation

6.29.1.1 LOAPEX::CSingleChannelSincFilter::CSingleChannelSincFilter (const int *channel*, const int *order*, const double *sample_time*)

Ctor takes a channel, an order and a sample period. It uses the channel to interface with the receiver navigation interface. (There is assumed to be only one transmitter, so that channel is always 1.) The order is the order of the filter you want. I prefer this be odd: never tested for even order.

6.29.1.2 LOAPEX::CSingleChannelSincFilter::~~CSingleChannelSincFilter (void)

6.29.2 Member Function Documentation

6.29.2.1 void LOAPEX::CSingleChannelSincFilter::clock (void)
[virtual]

Causes the filter to step once.

Reimplemented from LOAPEX::CBaseFilter (p. 28).

6.29.2.2 void LOAPEX::CSingleChannelSincFilter::disable_logging (void)

Help utility that stops the dumping of coefficients to a log file.

6.29.2.3 void LOAPEX::CSingleChannelSincFilter::enable_logging (void)

Help utility that causes coefficients to be dumped to a log file.

6.29.2.4 void LOAPEX::CSingleChannelSincFilter::set_demultiplexor (LOAPEX::CDemultiplexor * *p*)

Copies the handle to the demux into internal storage. It is the user's responsibility to free the demux after the class is destroyed.

6.29.2.5 void LOAPEX::CSingleChannelSincFilter::set_mapper (LOAPEX::CTimeBaseMapEngine * *p*)

Copies the mapper pointer into internal storage.

6.29.2.6 void LOAPEX::CSingleChannelSincFilter::set_multiplexor (LOAPEX::CMultiplexor * *p*)

Copies the handle to the mux into internal storage. It is the user's responsibility to free the emux after the class is destroyed.

6.29.2.7 void LOAPEX::CSingleChannelSincFilter::set_rx_start_time (const time_t t)

This method tells the filter the time of the first sample to be processed. Usually this will be taken from the receiver file (which knows when it started taking data.)

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

- **sincfilter.hpp**
- **sincfilter.cpp**

6.30 LOAPEX::CTimeBaseMapEngine Class Reference

```
#include <mapengine.hpp>
```

Public Member Functions

- **CTimeBaseMapEngine** (const time_t tstart, const time_t tstop)
- **~CTimeBaseMapEngine** (void)
- void **load_parameters** (const LOAPEX::CGeoParameters &P)
- void **load_TX** (const char *name, const int channel)
- void **load_RX** (const char *name, const int channel)
- void **load_TX** (LOAPEX::CBase3DNavigationData *p)
- void **load_RX** (LOAPEX::CBase3DNavigationData *p)
- void **arm** (void)
- bool **is_valid** (void)
- void **get_forward_time** (const CCompoundTime &Tin, CCompoundTime *pTout)
- void **get_inverse_time** (const CCompoundTime &Tin, CCompoundTime *pTout)
- void **get_first_time** (time_t *pt) const
- void **get_last_time** (time_t *pt) const
- void **dump** (const int howmany)

6.30.1 Detailed Description

A class that describes the mapping between two timebases.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 LOAPEX::CTimeBaseMapEngine::CTimeBaseMapEngine (const time_t tstart, const time_t tstop)

The Ctor initializes a few data members, and copies the two supplied times to internal member. These two times are intended to be inside the times of the navigation files.

6.30.2.2 LOAPEX::CTimeBaseMapEngine::~~CTimeBaseMapEngine (void)

Dtor deallocates all the instantiated internal objects.

6.30.3 Member Function Documentation

6.30.3.1 void LOAPEX::CTimeBaseMapEngine::arm (void)

Gets start and stop times of TX and RX nav data, compares them to the requested start and stop times. Aborts with an error message if incompatible .

6.30.3.2 void LOAPEX::CTimeBaseMapEngine::dump (const int *howmany*)

Diagnostic utility.

6.30.3.3 void LOAPEX::CTimeBaseMapEngine::get_first_time (time_t * *pt*) const

Convenience function to retrieve the time supplied during construction.

6.30.3.4 void LOAPEX::CTimeBaseMapEngine::get_forward_time (const CCompoundTime & *Tin*, CCompoundTime * *pTout*)

this is the main forward interpolation engine:

6.30.3.5 void LOAPEX::CTimeBaseMapEngine::get_inverse_time (const CCompoundTime & *Tin*, CCompoundTime * *pTout*)

this is the main inverse interpolation engine:

6.30.3.6 void LOAPEX::CTimeBaseMapEngine::get_last_time (time_t * *pt*) const

Convenience function to retrieve the time supplied during construction.

6.30.3.7 bool LOAPEX::CTimeBaseMapEngine::is_valid (void)

True if the mapper armed correctly. False if the mapped aborted during arming.

**6.30.3.8 void LOAPEX::CTimeBaseMapEngine::load_parameters
(const LOAPEX::CGeoParameters & *P*)**

Copies the parameters to internal storage. This is also the routine where the wavenumber vectors are computed (for internal use only.)

**6.30.3.9 void LOAPEX::CTimeBaseMapEngine::load_RX
(LOAPEX::CBase3DNavigationData * *p*)**

Overloaded member, copies a pointer from a navigation interface that has been previously instantiated into internal storage.

**6.30.3.10 void LOAPEX::CTimeBaseMapEngine::load_RX (const
char * *name*, const int *channel*)**

Overloaded member, loads a netCDF file for the receiver.

**6.30.3.11 void LOAPEX::CTimeBaseMapEngine::load_TX
(LOAPEX::CBase3DNavigationData * *p*)**

Overloaded member, copies a pointer from a navigation interface that has been previously instantiated into internal storage.

**6.30.3.12 void LOAPEX::CTimeBaseMapEngine::load_TX (const
char * *name*, const int *channel*)**

Overloaded member, loads a netCDF file for the transmitter.

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

- **mapengine.hpp**
- **mapengine.cpp**

6.31 LOAPEX::CTimeTag Class Reference

```
#include <timetag.hpp>
```

Public Member Functions

- **CTimeTag** ()
- **CTimeTag** (char *string)
- **CTimeTag** (CTimeTag &)
- ~**CTimeTag** ()
- **CTimeTag** & **operator=** (const **CTimeTag** &)
- std::string **get_tag** (void)
- void **get_components** (int *yyyy, int *ddd, int *hr, int *min, float *ss)
- long double **get_decimal_day** (void)
- long **get_year_seconds** (void)
- void **KM_to_tm** (struct tm &TM)
- void **tm_to_KM** (struct tm &TM, std::string &KMtime)
- int **compute_mday** (void)

Public Attributes

- char * **str_time_**
- int **year_**
- int **month_**
- int **day_**
- int **hour_**
- int **minute_**
- float **second_**
- long double **decYD_**
- long double **decJD_**

Friends

- std::ostream & **operator<<** (std::ostream &, **CTimeTag** &)

6.31.1 Detailed Description

One of my first big classes! Handles time conversions from Kurt's format to linear time. Everything is exposed. Didn't understand data hiding (or couldn't make it work...)

6.31.2 Constructor & Destructor Documentation

6.31.2.1 CTimeTag::CTimeTag ()

6.31.2.2 CTimeTag::CTimeTag (char * *string*)

6.31.2.3 CTimeTag::CTimeTag (CTimeTag &)

6.31.2.4 CTimeTag::~~CTimeTag ()

6.31.3 Member Function Documentation

6.31.3.1 int CTimeTag::compute_mday (void)

6.31.3.2 void CTimeTag::get_components (int * *yyyy*, int * *ddd*,
int * *hr*, int * *min*, float * *ss*)

6.31.3.3 long double LOAPEX::CTimeTag::get_decimal_day
(void) [inline]

6.31.3.4 std::string LOAPEX::CTimeTag::get_tag (void) [inline]

6.31.3.5 long CTimeTag::get_year_seconds (void)

6.31.3.6 void CTimeTag::KM_to_tm (struct tm & *TM*)

6.31.3.7 CTimeTag & CTimeTag::operator= (const CTimeTag &)

6.31.3.8 void CTimeTag::tm_to_KM (struct tm & *TM*, std::string
& *KMtime*)

6.31.4 Friends And Related Function Documentation

6.31.4.1 std::ostream& operator<< (std::ostream &, CTimeTag &)
[friend]

6.31.5 Member Data Documentation

6.31.5.1 int LOAPEX::CTimeTag::day_

6.31.5.2 long double LOAPEX::CTimeTag::decJD_

6.31.5.3 long double LOAPEX::CTimeTag::decYD_

6.31.5.4 int LOAPEX::CTimeTag::hour_

6.31.5.5 int LOAPEX::CTimeTag::minute_

6.31.5.6 int LOAPEX::CTimeTag::month_

6.31.5.7 float LOAPEX::CTimeTag::second_

6.31.5.8 char* LOAPEX::CTimeTag::str_time_

6.31.5.9 int LOAPEX::CTimeTag::year_

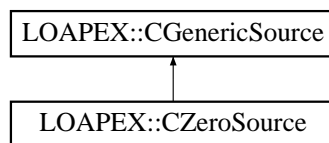
The documentation for this class was generated from the following files:

- `timetag.hpp`
- `timetag.cpp`

6.32 LOAPEX::CZeroSource Class Reference

```
#include <gappystream.hpp>
```

Inheritance diagram for LOAPEX::CZeroSource::



Public Member Functions

- **CZeroSource** (const long start_second, const long total_scans, const int total_channels, const int sample_rate)
- **~CZeroSource** (void)
- void **get_scan** (std::vector< short > &V)
- void **get_start_second** (long *s) const
- void **get_total_scans** (long *t) const
- void **get_total_channels** (int *c) const
- void **get_sample_rate** (int *fs) const
- void **get_type** (char *c) const

6.32.1 Detailed Description

Specialized class for sources of zeroes. Interface mimics a .adf file source.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 LOAPEX::CZeroSource::CZeroSource (const long start_second, const long total_scans, const int total_channels, const int sample_rate)

6.32.2.2 LOAPEX::CZeroSource::~~CZeroSource (void)

6.32.3 Member Function Documentation

6.32.3.1 void LOAPEX::CZeroSource::get_sample_rate (int * fs) const [virtual]

Implements LOAPEX::CGenericSource (p. 49).

6.32.3.2 void LOAPEX::CZeroSource::get_scan (std::vector< short > & V) [virtual]

Implements LOAPEX::CGenericSource (p. 50).

6.32.3.3 void LOAPEX::CZeroSource::get_start_second (long * s) const [virtual]

Implements LOAPEX::CGenericSource (p. 50).

6.32.3.4 void LOAPEX::CZeroSource::get_total_channels (int * c) const [virtual]

Implements LOAPEX::CGenericSource (p. 50).

6.32.3.5 void LOAPEX::CZeroSource::get_total_scans (long * t) const [virtual]

Implements LOAPEX::CGenericSource (p. 50).

6.32.3.6 void LOAPEX::CZeroSource::get_type (char * c) const [virtual]

Supplies the character "Z", for cosmetic output, indicating a zero source class.

Implements LOAPEX::CGenericSource (p. 50).

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

- gappystream.hpp
- gappystream.cpp

6.33 dEuclideanVector_ t Struct Reference

```
#include <navigation.hpp>
```

Public Attributes

- double **x**
- double **y**
- double **z**

6.33.1 Member Data Documentation

6.33.1.1 double dEuclideanVector_ t::x

6.33.1.2 double dEuclideanVector_ t::y

6.33.1.3 double dEuclideanVector_ t::z

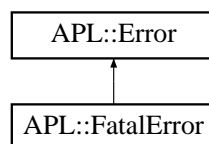
The documentation for this struct was generated from the following file:

- **navigation.hpp**

6.34 APL::Error Class Reference

```
#include <errors.hpp>
```

Inheritance diagram for APL::Error::



Public Member Functions

- **Error** (std::string s)
- **Error** (const char *c)

Public Attributes

- std::string **error_msg_**

6.34.1 Constructor & Destructor Documentation

6.34.1.1 APL::Error::Error (std::string s) [inline]

6.34.1.2 APL::Error::Error (const char * c) [inline]

6.34.2 Member Data Documentation

6.34.2.1 std::string APL::Error::error_msg_

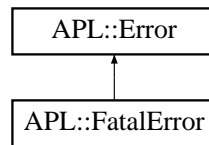
The documentation for this class was generated from the following file:

- **errors.hpp**

6.35 APL::FatalError Class Reference

```
#include <errors.hpp>
```

Inheritance diagram for APL::FatalError::



Public Member Functions

- **FatalError** (std::string *s*)
- **FatalError** (const char **c*)

6.35.1 Constructor & Destructor Documentation

6.35.1.1 APL::FatalError::FatalError (std::string *s*) [inline]

6.35.1.2 APL::FatalError::FatalError (const char * *c*) [inline]

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

- **errors.hpp**

6.36 tstream Class Reference

```
#include <tstream.hpp>
```

Public Member Functions

- **tstream** (std::ostream &os)
- **~tstream** (void)
- void **operator>>** (char *)
- void **operator>>** (std::string &s)
- int **seekt** (long, std::ios_base::seekdir)
- bool **eos** (void)

6.36.1 Detailed Description

A class for a stream of tokens, derived from the simple list class. New functionality over the base class includes the extraction operator >> , the ability to seek to any particular token, and an "end of stream" function.

6.36.2 Constructor & Destructor Documentation

6.36.2.1 **tstream::tstream** (std::ostream & *os*)

6.36.2.2 **tstream::~~tstream** (void)

6.36.3 Member Function Documentation

6.36.3.1 **bool tstream::eos** (void) [inline]

6.36.3.2 **void tstream::operator>>** (std::string & *s*)

6.36.3.3 **void tstream::operator>>** (char *)

6.36.3.4 **int tstream::seekt** (long, std::ios_base::seekdir)

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

- **tstream.hpp**
- **tstream.cpp**

Chapter 7

DopplerToolkit File Documentation

7.1 adfadapter.cpp File Reference

```
#include <adfadapter.hpp>
```

```
#include <adffile.hpp>
```

```
#include <iostream>
```

7.2 adfadapter.hpp File Reference

```
#include <vector>
#include <string>
#include <basefileadapter.hpp>
#include <adffile.hpp>
```

Namespaces

- namespace **LOAPEX**

7.3 adffile.cpp File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <fstream>
#include <iostream>
#include <sstream>
#include <adffile.hpp>
#include <tstream.hpp>
#include <errors.hpp>
#include <timetag.hpp>
#include <time.h>
```

Defines

- #define **SBUFFERSIZE** 256
- #define **TOKEN_BUFFER_SIZE** 100
- #define **ECHOINDENT** " "

Functions

- int **FindLine** (std::ifstream &instream, const char *key)
- int **FindLine** (std::fstream &instream, const char *key)

7.3.1 Define Documentation

7.3.1.1 #define **ECHOINDENT** " "

7.3.1.2 #define **SBUFFERSIZE** 256

7.3.1.3 #define **TOKEN_BUFFER_SIZE** 100

7.3.2 Function Documentation

7.3.2.1 int **FindLine** (std::fstream & *instream*, const char * *key*)

7.3.2.2 int **FindLine** (std::ifstream & *instream*, const char * *key*)

7.4 adffile.hpp File Reference

```
#include <fstream>
#include <vector>
#include <map>
#include <string>
```

Namespaces

- namespace **LOAPEX**

Defines

- #define **DEFAULT_SAMPLE_RATE** 300
- #define **DEFAULT_SITE** "no-site"
- #define **DEFAULT_ID** "No ID"
- #define **BYTE** unsigned char

7.4.1 Define Documentation

7.4.1.1 #define **BYTE** unsigned char

7.4.1.2 #define **DEFAULT_ID** "No ID"

7.4.1.3 #define **DEFAULT_SAMPLE_RATE** 300

7.4.1.4 #define **DEFAULT_SITE** "no-site"

7.5 allpass.cpp File Reference

```
#include <iostream>
#include <allpass.hpp>
```

7.6 allpass.hpp File Reference

```
#include <basefilter.hpp>
```

Namespaces

- namespace **LOAPEX**

7.7 allstop.cpp File Reference

```
#include <allstop.hpp>
```

7.8 allstop.hpp File Reference

```
#include <basefilter.hpp>
```

Namespaces

- namespace **LOAPEX**

7.9 basefileadapter.hpp File Reference

```
#include <vector>
```

```
#include <string>
```

Namespaces

- namespace **LOAPEX**

7.10 basefilter.cpp File Reference

```
#include <basefilter.hpp>  
#include <iostream>
```

7.11 basefilter.hpp File Reference

```
#include <multiplexor.hpp>  
#include <demultiplexor.hpp>
```

Namespaces

- namespace **LOAPEX**

7.12 basemapsolver.hpp File Reference

```
#include <compoundtime.hpp>
```

```
#include <parameters.hpp>
```

```
#include <navigation.hpp>
```

Namespaces

- namespace **LOAPEX**

7.13 compoundtime.cpp File Reference

```
#include <stdlib.h>
#include <math.h>
#include <iostream>
#include <compoundtime.hpp>
```

Functions

- **LOAPEX::CCompoundTime** **operator+**
(LOAPEX::CCompoundTime &a, LOAPEX::CCompoundTime &b)
- **LOAPEX::CCompoundTime** **operator-**
(LOAPEX::CCompoundTime &a, LOAPEX::CCompoundTime &b)

7.13.1 Function Documentation

7.13.1.1 **LOAPEX::CCompoundTime operator+**
(LOAPEX::CCompoundTime & *a*,
LOAPEX::CCompoundTime & *b*)

7.13.1.2 **LOAPEX::CCompoundTime operator-**
(LOAPEX::CCompoundTime & *a*,
LOAPEX::CCompoundTime & *b*)

7.14 compoundtime.hpp File Reference

Namespaces

- namespace **LOAPEX**

Functions

- **LOAPEX::CCompoundTime** **operator+**
(**LOAPEX::CCompoundTime** &a, **LOAPEX::CCompoundTime** &b)
- **LOAPEX::CCompoundTime** **operator-**
(**LOAPEX::CCompoundTime** &a, **LOAPEX::CCompoundTime** &b)

7.14.1 Function Documentation

7.14.1.1 **LOAPEX::CCompoundTime** **operator+**
(**LOAPEX::CCompoundTime** & *a*,
LOAPEX::CCompoundTime & *b*)

7.14.1.2 **LOAPEX::CCompoundTime** **operator-**
(**LOAPEX::CCompoundTime** & *a*,
LOAPEX::CCompoundTime & *b*)

7.15 demultiplexor.cpp File Reference

```
#include <demultiplexor.hpp>  
#include <iostream>
```

7.16 demultiplexor.hpp File Reference

```
#include <basefileadapter.hpp>
```

Namespaces

- namespace **LOAPEX**

7.17 errors.hpp File Reference

```
#include <string>
```

Namespaces

- namespace **APL**

7.18 gappystream.cpp File Reference

```
#include <stdio.h>
#include <iostream>
#include <errors.hpp>
#include <gappystream.hpp>
```

7.19 gappystream.hpp File Reference

```
#include <iostream>
#include <list>
#include <vector>
#include <adffile.hpp>
```

Namespaces

- namespace **LOAPEX**

7.20 mapengine.cpp File Reference

```
#include <math.h>
#include <time.h>
#include <iostream>
#include <errors.hpp>
#include <mapengine.hpp>
```

Defines

- #define **GRID_SAMPLE_RATE** 1.0
- #define **NRANSI**
- #define **ITMAX** 100
- #define **EPS** 3.0e-8
- #define **SIGN(a, b)** ((b)>0.0 ? fabs(a) : -fabs(a))

7.20.1 Define Documentation

7.20.1.1 #define **EPS** 3.0e-8

7.20.1.2 #define **GRID_SAMPLE_RATE** 1.0

7.20.1.3 #define **ITMAX** 100

7.20.1.4 #define **NRANSI**

7.20.1.5 #define **SIGN(a, b)** ((b)>0.0 ? fabs(a) : -fabs(a))

7.21 mapengine.hpp File Reference

```
#include <time.h>
#include <vector>
#include <mapsolvers.hpp>
#include <navigation.hpp>
#include <compoundtime.hpp>
#include <parameters.hpp>
#include <tablefunction.hpp>
```

Namespaces

- namespace **LOAPEX**

7.22 mapsolvers.cpp File Reference

```
#include <math.h>
#include <errors.hpp>
#include <mapsolvers.hpp>
#include <compoundtime.hpp>
#include <parameters.hpp>
#include <navigation.hpp>
```

Defines

- `#define BRENTTOL 0.00001`
- `#define NEWTONTOL 0.00001`
- `#define NRANSI`
- `#define ITMAX 100`
- `#define EPS 3.0e-8`
- `#define SIGN(a, b) ((b)>0.0 ? fabs(a) : -fabs(a))`
- `#define JMAX 20`

7.22.1 Define Documentation

7.22.1.1 `#define BRENTTOL 0.00001`

7.22.1.2 `#define EPS 3.0e-8`

7.22.1.3 `#define ITMAX 100`

7.22.1.4 `#define JMAX 20`

7.22.1.5 `#define NEWTONTOL 0.00001`

7.22.1.6 `#define NRANSI`

7.22.1.7 `#define SIGN(a, b) ((b)>0.0 ? fabs(a) : -fabs(a))`

7.23 mapsolvers.hpp File Reference

```
#include <math.h>
#include <basemapsolver.hpp>
#include <compoundtime.hpp>
#include <parameters.hpp>
```

Namespaces

- namespace **LOAPEX**

7.24 multichannelfilter.cpp File Reference

```
#include <multichannelfilter.hpp>
```

7.25 multichannelfilter.hpp File Reference

```
#include <basefilter.hpp>
```

```
#include <vector>
```

Namespaces

- namespace **LOAPEX**

7.26 multiplexor.cpp File Reference

```
#include <multiplexor.hpp>
```

```
#include <iostream>
```

7.27 multiplexor.hpp File Reference

```
#include <muxfile.hpp>
```

```
#include <vector>
```

Namespaces

- namespace **LOAPEX**

7.28 muxadapter.cpp File Reference

```
#include <muxadapter.hpp>
```

```
#include <muxfile.hpp>
```

```
#include <iostream>
```

7.29 muxadapter.hpp File Reference

```
#include <vector>
#include <string>
#include <basefileadapter.hpp>
#include <muxfile.hpp>
```

Namespaces

- namespace **LOAPEX**

7.30 muxfile.cpp File Reference

```
#include <errors.hpp>
#include <iostream>
#include <muxfile.hpp>
#include <string>
#include <list>
#include <sstream>
#include <stdio.h>
#include <time.h>
#include <string.h>
#include <stdlib.h>
```

Functions

- void `clean_line_list` (std::list< std::string * > &L)

7.30.1 Function Documentation

7.30.1.1 void `clean_line_list` (std::list< std::string * > & *L*)

7.31 muxfile.hpp File Reference

```
#include <fstream>
#include <vector>
#include <string>
#include <list>
```

Namespaces

- namespace **LOAPEX**

7.32 navigation.cpp File Reference

```
#include <math.h>
#include <string.h>
#include <fstream>
#include <iostream>
#include <sstream>
#include <navigation.hpp>
#include <compoundtime.hpp>
#include <tablefunction.hpp>
```

7.33 navigation.hpp File Reference

```
#include <time.h>
#include <netcdf.hh>
#include <compoundtime.hpp>
#include <tablefunction.hpp>
```

Namespaces

- namespace **LOAPEX**

Defines

- #define **RXRADIUS** 50.0
- #define **RXRADIUSINCREMENT** 50.0
- #define **RXDEPTH** 1000.0
- #define **RXDEPTHINCREMENT** 500.0
- #define **RXPERIOD** 4320.0
- #define **RXTIME_START** 1096392840
- #define **RXTIME_STOP** 1099439999
- #define **TXRADIUS** 10.0
- #define **TXPERIOD** 50.0
- #define **TXDEPTH** 0.0
- #define **TXTIME_START** 1099180800
- #define **TXTIME_STOP** 1099439999

7.33.1 Define Documentation

7.33.1.1 `#define RXDEPTH 1000.0`

7.33.1.2 `#define RXDEPTHINCREMENT 500.0`

7.33.1.3 `#define RXPERIOD 4320.0`

7.33.1.4 `#define RXRADIUS 50.0`

7.33.1.5 `#define RXRADIUSINCREMENT 50.0`

7.33.1.6 `#define RXTIME_START 1096392840`

7.33.1.7 `#define RXTIME_STOP 1099439999`

7.33.1.8 `#define TXDEPTH 0.0`

7.33.1.9 `#define TXPERIOD 50.0`

7.33.1.10 `#define TXRADIUS 10.0`

7.33.1.11 `#define TXTIME_START 1099180800`

7.33.1.12 `#define TXTIME_STOP 1099439999`

7.34 nrutil.c File Reference

```
#include <stdio.h>
#include <stddef.h>
#include <stdlib.h>
```

Defines

- #define **NR_END** 1
- #define **FREE_ARG** char*

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)

7.34.1 Define Documentation

7.34.1.1 `#define FREE_ARG char*`

7.34.1.2 `#define NR_END 1`

7.34.2 Function Documentation

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

7.34.2.2 `unsigned char* cvector (long nl, long nh)`

7.34.2.3 `double** dmatrix (long nrl, long nrh, long ncl, long nch)`

7.34.2.4 `double* dvector (long nl, long nh)`

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

7.34.2.6 `void free_convert_matrix (float ** b, long nrl, long nrh, long ncl, long nch)`

7.34.2.7 `void free_cvector (unsigned char * v, long nl, long nh)`

7.34.2.8 `void free_dmatrix (double ** m, long nrl, long nrh, long ncl, long nch)`

7.34.2.9 `void free_dvector (double * v, long nl, long nh)`

7.34.2.10 `void free_f3tensor (float *** t, long nrl, long nrh, long ncl, long nch, long ndl, long ndh)`

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

7.34.2.12 `void free_ivector (int * v, long nl, long nh)`

7.34.2.13 `void free_lvector (unsigned long * v, long nl, long nh)`

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

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

~~7.34.2.16 `void free_vector (float * v, long nl, long nh)`~~

Generated on Thu May 3 00:53:59 2007 for DopplerToolkit by Doxygen

7.34.2.17 `int** imatrix (long nrl, long nrh, long ncl, long nch)`

7.34.2.18 `int* ivector (long nl, long nh)`

7.34.2.19 `unsigned long* lvector (long nl, long nh)`

7.34.2.20 `float** matrix (long nrl, long nrh, long ncl, long nch)`

7.34.2.21 `void nrerror (char error_text[])`

7.34.2.22 `float** submatrix (float ** a, long oldrl, long oldrh, long oldcl, long oldch, long newrl, long newcl)`

7.35 parameters.hpp File Reference

```
#include <navigation.hpp>
```

Namespaces

- namespace **LOAPEX**

7.36 refman.dox File Reference

7.37 sincfilter.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <stdio.h>
#include <sincfilter.hpp>
```

Defines

- `#define LOGNAME "sincfilter.log"`

7.37.1 Define Documentation

7.37.1.1 `#define LOGNAME "sincfilter.log"`

7.38 sincfilter.hpp File Reference

```
#include <basefilter.hpp>
#include <mapengine.hpp>
#include <stdio.h>
#include <iostream>
#include <fstream>
```

Namespaces

- namespace **LOAPEX**

7.39 spline.c File Reference

Defines

- `#define NRANSI`

Functions

- void **nerror** (char *error_text*[])
- double * **dvector** (long *nl*, long *nh*)
- void **free_dvector** (double **v*, long *nl*, long *nh*)
- void **spline** (double *x*[], double *y*[], int *n*, double *yp1*, double *ypn*, double *y2*[])

7.39.1 Define Documentation

7.39.1.1 `#define NRANSI`

7.39.2 Function Documentation

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

7.39.2.2 void **free_dvector** (double * *v*, long *nl*, long *nh*)

7.39.2.3 void **nerror** (char *error_text*[])

7.39.2.4 void **spline** (double *x*[], double *y*[], int *n*, double *yp1*, double *ypn*, double *y2*[])

7.40 splint.c File Reference

Functions

- void **nerror** (char error_text[])
- void **splint** (double xa[], double ya[], double y2a[], int n, double x, double *y)

7.40.1 Function Documentation

7.40.1.1 void **nerror** (char *error_text*[])

7.40.1.2 void **splint** (double *xa*[], double *ya*[], double *y2a*[], int *n*, double *x*, double * *y*)

7.41 tablefunction.cpp File Reference

```
#include <string>
#include <fstream>
#include <iostream>
#include <vector>
#include <stdio.h>
#include <tablefunction.hpp>
```

Functions

- void **spline** (double *x*[], double *y*[], int *n*, double *yp1*, double *ypn*, double *y2*[])
- void **splint** (double *xa*[], double *ya*[], double *y2a*[], int *n*, double *x*, double **y*)

7.41.1 Function Documentation

7.41.1.1 void **spline** (double *x*[], double *y*[], int *n*, double *yp1*, double *ypn*, double *y2*[])

7.41.1.2 void **splint** (double *xa*[], double *ya*[], double *y2a*[], int *n*, double *x*, double **y*)

7.42 tablefunction.hpp File Reference

```
#include <string>
#include <fstream>
#include <iostream>
#include <vector>
#include <stdio.h>
```

Namespaces

- namespace **LOAPEX**

7.43 timetag.cpp File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <iostream>
#include <sstream>
#include <timetag.hpp>
#include <time.h>
```

Functions

- int **decYD_to_month** (float decYD, int year)
- int **isleap** (int year)

7.43.1 Function Documentation

7.43.1.1 int **decYD_to_month** (float *decYD*, int *year*)

7.43.1.2 int **isleap** (int *year*)

7.44 timetag.hpp File Reference

```
#include <string>
```

```
#include <time.h>
```

Namespaces

- namespace **LOAPEX**

7.45 tstream.cpp File Reference

```
#include <string.h>
#include <stdio.h>
#include <iostream>
#include <fstream>
#include <sstream>
#include <tstream.hpp>
```

Defines

- #define **CR** 0x0D
- #define **SPACE** 0x20
- #define **TAB** 0x09
- #define **LINEFEED** 0x0A

7.45.1 Define Documentation

7.45.1.1 #define **CR** 0x0D

7.45.1.2 #define **LINEFEED** 0x0A

7.45.1.3 #define **SPACE** 0x20

7.45.1.4 #define **TAB** 0x09

7.46 tstream.hpp File Reference

```
#include <fstream>
#include <sstream>
#include <string>
#include <list>
```

Index

- ~CADFFFile
 - LOAPEX::CADFFFile, 12
- ~CADFFFileAdapter
 - LOAPEX::CADFFFileAdapter, 15
- ~CADFInfo
 - LOAPEX::CADFInfo, 17
- ~CADFSource
 - LOAPEX::CADFSource, 19
- ~CBase3DNavigationData
 - LOAPEX::CBase3DNavigation-Data, 22
- ~CBaseFileAdapter
 - LOAPEX::CBaseFileAdapter, 25
- ~CBaseFilter
 - LOAPEX::CBaseFilter, 28
- ~CBaseMapSolver
 - LOAPEX::CBaseMapSolver, 30
- ~CBrentMapSolver
 - LOAPEX::CBrentMapSolver, 32
- ~CCompoundTime
 - LOAPEX::CCompoundTime, 35
- ~CDemultiplexor
 - LOAPEX::CDemultiplexor, 36
- ~CFakeRXNavigationData
 - LOAPEX::CFakeRXNavigation-Data, 39
- ~CFakeTXNavigationData
 - LOAPEX::CFakeTXNavigation-Data, 42
- ~CFixedNavigationData
 - LOAPEX::CFixedNavigation-Data, 45
- ~CGappyADFStream
 - LOAPEX::CGappyADFStream, 47
- ~CGenericSource
 - LOAPEX::CGenericSource, 49
- ~CInterpolatedTableFunction
 - LOAPEX::CInterpolatedTable-Function, 53
- ~CMUXFile
 - LOAPEX::CMUXFile, 60
- ~CMUXFileAdapter
 - LOAPEX::CMUXFileAdapter, 64
- ~CMultichannelFilter
 - LOAPEX::CMultichannelFilter, 55
- ~CMultiplexor
 - LOAPEX::CMultiplexor, 57
- ~CNetCDFNavigationData
 - LOAPEX::CNetCDFNavigation-Data, 67
- ~CNewtonMapSolver
 - LOAPEX::CNewtonMapSolver, 70
- ~CRex3DNavigationData
 - LOAPEX::CRex3DNavigation-Data, 72
- ~CSingleChannelAllPassFilter
 - LOAPEX::CSingleChannelAll-PassFilter, 74
- ~CSingleChannelAllStopFilter
 - LOAPEX::CSingleChannelAll-StopFilter, 76
- ~CSingleChannelSincFilter
 - LOAPEX::CSingleChannelSinc-Filter, 78
- ~CTimeBaseMapEngine
 - LOAPEX::CTimeBaseMap-Engine, 81
- ~CTimeTag
 - LOAPEX::CTimeTag, 85
- ~CZeroSource
 - LOAPEX::CZeroSource, 87

- ~tstream
 - tstream, 92
- add_filter
 - LOAPEX::CMultichannelFilter, 55
- adfadapter.cpp, 93
- adfadapter.hpp, 94
- adffile.cpp, 95
 - ECHOINDENT, 95
 - FindLine, 95
 - SBUFFERSIZE, 95
 - TOKEN_BUFFER_SIZE, 95
- adffile.hpp, 96
 - BYTE, 96
 - DEFAULT_ID, 96
 - DEFAULT_SAMPLE_RATE, 96
 - DEFAULT_SITE, 96
- allpass.cpp, 97
- allpass.hpp, 98
- allstop.cpp, 99
- allstop.hpp, 100
- APL, 9
 - APL::Error, 90
 - Error, 90
 - error_msg_, 90
 - APL::FatalError, 91
 - APL::FatalError
 - FatalError, 91
- arm
 - LOAPEX::CTimeBaseMapEngine, 82
- basefileadapter.hpp, 101
- basefilter.cpp, 102
- basefilter.hpp, 103
- basemapsolver.hpp, 104
- BRENTTOL
 - mapsolvers.cpp, 114
- BYTE
 - adffile.hpp, 96
- CADFFile
 - LOAPEX::CADFFile, 12
- CADFFileAdapter
 - LOAPEX::CADFFileAdapter, 15
- CADFFInfo
 - LOAPEX::CADFFInfo, 17
- CADFSOURCE
 - LOAPEX::CADFSOURCE, 19
- CBase3DNavigationData
 - LOAPEX::CBase3DNavigationData, 22
- CBaseFilter
 - LOAPEX::CBaseFilter, 28
- CBrentMapSolver
 - LOAPEX::CBrentMapSolver, 32
- CCompoundTime
 - LOAPEX::CCompoundTime, 34
- CDemultiplexor
 - LOAPEX::CDemultiplexor, 36
- CFakeRXNavigationData
 - LOAPEX::CFakeRXNavigationData, 39
- CFakeTXNavigationData
 - LOAPEX::CFakeTXNavigationData, 42
- CFixedNavigationData
 - LOAPEX::CFixedNavigationData, 45
- CGappyADFStream
 - LOAPEX::CGappyADFStream, 47
- channel
 - LOAPEX::CGeoParameters, 51
- channel_id_
 - LOAPEX::CBaseFilter, 28
- CInterpolatedTableFunction
 - LOAPEX::CInterpolatedTableFunction, 53
- clean_line_list
 - muxfile.cpp, 122
- clock
 - LOAPEX::CBaseFilter, 28
 - LOAPEX::CDemultiplexor, 36
 - LOAPEX::CMultichannelFilter, 55
 - LOAPEX::CMultiplexor, 57
 - LOAPEX::CSingleChannelAllPassFilter, 74

-
- LOAPEX::CSingleChannelAll-StopFilter, 76
 - LOAPEX::CSingleChannelSincFilter, 79
 - close
 - LOAPEX::CMUXFile, 60
 - CMultichannelFilter
 - LOAPEX::CMultichannelFilter, 55
 - CMultiplexor
 - LOAPEX::CMultiplexor, 57
 - CMUXFile
 - LOAPEX::CMUXFile, 60
 - CMUXFileAdapter
 - LOAPEX::CMUXFileAdapter, 64
 - CNetCDFNavigationData
 - LOAPEX::CNetCDFNavigationData, 67
 - CNewtonMapSolver
 - LOAPEX::CNewtonMapSolver, 70
 - compoundtime.cpp, 105
 - operator+, 105
 - operator-, 105
 - compoundtime.hpp, 106
 - operator+, 106
 - operator-, 106
 - compute_map
 - LOAPEX::CBaseMapSolver, 30
 - LOAPEX::CBrentMapSolver, 32
 - LOAPEX::CNewtonMapSolver, 70
 - compute_mday
 - LOAPEX::CTimeTag, 85
 - convert_matrix
 - nrutil.c, 128
 - CR
 - tstream.cpp, 139
 - CRex3DNavigationData
 - LOAPEX::CRex3DNavigationData, 72
 - CSingleChannelAllPassFilter
 - LOAPEX::CSingleChannelAllPassFilter, 74
 - CSingleChannelAllStopFilter
 - LOAPEX::CSingleChannelAllStopFilter, 76
 - CSingleChannelSincFilter
 - LOAPEX::CSingleChannelSincFilter, 78
 - CTimeBaseMapEngine
 - LOAPEX::CTimeBaseMapEngine, 81
 - CTimeTag
 - LOAPEX::CTimeTag, 85
 - cvector
 - nrutil.c, 128
 - CZeroSource
 - LOAPEX::CZeroSource, 87
 - day_
 - LOAPEX::CTimeTag, 85
 - decJD_
 - LOAPEX::CTimeTag, 85
 - decYD_
 - LOAPEX::CTimeTag, 85
 - decYD_to_month
 - timetag.cpp, 137
 - DEFAULT_ID
 - adffile.hpp, 96
 - DEFAULT_SAMPLE_RATE
 - adffile.hpp, 96
 - DEFAULT_SITE
 - adffile.hpp, 96
 - demultiplexor.cpp, 107
 - demultiplexor.hpp, 108
 - dEuclideanVector_t, 89
 - dEuclideanVector_t
 - x, 89
 - y, 89
 - z, 89
 - disable_logging
 - LOAPEX::CSingleChannelSincFilter, 79
 - dmatrix
 - nrutil.c, 128
 - dump
 - LOAPEX::CTimeBaseMapEngine, 82
 - dvector
 - nrutil.c, 128
-

-
- spline.c, 133
 - echo_header
 - LOAPEX::CADFFFile, 13
 - ECHOINDENT
 - adffile.cpp, 95
 - enable_logging
 - LOAPEX::CSingleChannelSincFilter, 79
 - eof
 - LOAPEX::CADFFFile, 13
 - eos
 - tstream, 92
 - EPS
 - mapengine.cpp, 112
 - mapsolvers.cpp, 114
 - Error
 - APL::Error, 90
 - error_msg_
 - APL::Error, 90
 - errors.hpp, 109
 - evaluate
 - LOAPEX::CInterpolatedTableFunction, 53
 - f3tensor
 - nrutil.c, 128
 - FatalError
 - APL::FatalError, 91
 - file_time_final_
 - LOAPEX::CBase3DNavigationData, 24
 - file_time_first_
 - LOAPEX::CBase3DNavigationData, 24
 - FindLine
 - adffile.cpp, 95
 - forward_bearing2APL
 - LOAPEX::CGeoParameters, 51
 - forward_bearing2VLA
 - LOAPEX::CGeoParameters, 51
 - fraction_
 - LOAPEX::CCompoundTime, 35
 - FREE_ARG
 - nrutil.c, 128
 - free_convert_matrix
 - nrutil.c, 128
 - free_cvector
 - nrutil.c, 128
 - free_dmatrix
 - nrutil.c, 128
 - free_dvector
 - nrutil.c, 128
 - free_f3tensor
 - nrutil.c, 128
 - free_imatrix
 - nrutil.c, 128
 - free_ivector
 - nrutil.c, 128
 - free_lvector
 - nrutil.c, 128
 - free_matrix
 - nrutil.c, 128
 - free_submatrix
 - nrutil.c, 128
 - free_vector
 - nrutil.c, 128
 - gappystream.cpp, 110
 - gappystream.hpp, 111
 - get_components
 - LOAPEX::CTimeTag, 85
 - get_decimal_day
 - LOAPEX::CTimeTag, 85
 - get_descriptions
 - LOAPEX::CMUXFile, 60
 - get_file_final_time
 - LOAPEX::CBase3DNavigationData, 23
 - LOAPEX::CFakeRXNavigationData, 39
 - LOAPEX::CFakeTXNavigationData, 42
 - LOAPEX::CFixedNavigationData, 45
 - LOAPEX::CNetCDFNavigationData, 67
 - LOAPEX::CReX3DNavigationData, 72
 - get_file_first_time
-

-
- LOAPEX::CBase3DNavigation-Data, 23
 - LOAPEX::CFakeRXNavigation-Data, 39
 - LOAPEX::CFakeTXNavigation-Data, 42
 - LOAPEX::CFixedNavigation-Data, 45
 - LOAPEX::CNetCDFNavigation-Data, 67
 - LOAPEX::CRex3DNavigation-Data, 72
 - get_first_time
 - LOAPEX::CTimeBaseMap-Engine, 82
 - get_forward_time
 - LOAPEX::CTimeBaseMap-Engine, 82
 - get_header
 - LOAPEX::CADFFFileAdapter, 15
 - LOAPEX::CADFInfo, 17
 - LOAPEX::CBaseFileAdapter, 25
 - LOAPEX::CMUXFile, 61
 - LOAPEX::CMUXFileAdapter, 64
 - get_id
 - LOAPEX::CADFInfo, 17
 - get_info_copy
 - LOAPEX::CADFFFile, 13
 - get_inverse_time
 - LOAPEX::CTimeBaseMap-Engine, 82
 - get_last_time
 - LOAPEX::CTimeBaseMap-Engine, 82
 - get_position
 - LOAPEX::CBase3DNavigation-Data, 23
 - LOAPEX::CFakeRXNavigation-Data, 39
 - LOAPEX::CFakeTXNavigation-Data, 42
 - LOAPEX::CFixedNavigation-Data, 45
 - LOAPEX::CNetCDFNavigation-Data, 68
 - LOAPEX::CRex3DNavigation-Data, 72
 - get_run_start
 - LOAPEX::CADFInfo, 18
 - get_sample_rate
 - LOAPEX::CADFFFile, 13
 - LOAPEX::CADFFFileAdapter, 15
 - LOAPEX::CADFInfo, 18
 - LOAPEX::CADFSource, 20
 - LOAPEX::CGappyADFStream, 48
 - LOAPEX::CGenericSource, 49
 - LOAPEX::CMUXFile, 61
 - LOAPEX::CMUXFileAdapter, 65
 - LOAPEX::CZeroSource, 87
 - get_scan
 - LOAPEX::CADFFFile, 13
 - LOAPEX::CADFFFileAdapter, 16
 - LOAPEX::CADFSource, 20
 - LOAPEX::CBaseFileAdapter, 25
 - LOAPEX::CGappyADFStream, 48
 - LOAPEX::CGenericSource, 49
 - LOAPEX::CMUXFile, 61
 - LOAPEX::CMUXFileAdapter, 65
 - LOAPEX::CZeroSource, 87
 - get_site
 - LOAPEX::CADFInfo, 18
 - get_start_second
 - LOAPEX::CADFFFile, 13
 - LOAPEX::CADFFFileAdapter, 16
 - LOAPEX::CADFSource, 20
 - LOAPEX::CBaseFileAdapter, 26
 - LOAPEX::CGappyADFStream, 48
 - LOAPEX::CGenericSource, 50
 - LOAPEX::CMUXFile, 61
 - LOAPEX::CMUXFileAdapter, 65
 - LOAPEX::CZeroSource, 88
 - get_table_length
 - LOAPEX::CInterpolatedTable-Function, 53
 - get_tag
 - LOAPEX::CTimeTag, 85
 - get_total_channels
 - LOAPEX::CADFFFile, 13
-

- LOAPEX::CADFFFileAdapter, 16
- LOAPEX::CADFSource, 20
- LOAPEX::CBaseFileAdapter, 26
- LOAPEX::CGappyADFStream, 48
- LOAPEX::CGenericSource, 50
- LOAPEX::CMUXFile, 61
- LOAPEX::CMUXFileAdapter, 65
- LOAPEX::CZeroSource, 88
- get_total_phones
 - LOAPEX::CADFInfo, 18
- get_total_scans
 - LOAPEX::CADFFFile, 14
 - LOAPEX::CADFFFileAdapter, 16
 - LOAPEX::CADFInfo, 18
 - LOAPEX::CADFSource, 20
 - LOAPEX::CBaseFileAdapter, 26
 - LOAPEX::CGappyADFStream, 48
 - LOAPEX::CGenericSource, 50
 - LOAPEX::CMUXFile, 61
 - LOAPEX::CMUXFileAdapter, 65
 - LOAPEX::CZeroSource, 88
- get_type
 - LOAPEX::CADFSource, 21
 - LOAPEX::CGenericSource, 50
 - LOAPEX::CZeroSource, 88
- get_velocity
 - LOAPEX::CBase3DNavigation-Data, 23
 - LOAPEX::CFakeRXNavigation-Data, 39
 - LOAPEX::CFakeTXNavigation-Data, 42
 - LOAPEX::CFixedNavigation-Data, 45
 - LOAPEX::CNetCDFNavigation-Data, 68
 - LOAPEX::CRex3DNavigation-Data, 72
- get_year_seconds
 - LOAPEX::CTimeTag, 85
- GRID_SAMPLE_RATE
 - mapengine.cpp, 112
- hour_
 - LOAPEX::CTimeTag, 85
- imatrix
 - nrrutil.c, 128
- is_loaded
 - LOAPEX::CADFFFile, 14
 - LOAPEX::CFakeRXNavigation-Data, 40
 - LOAPEX::CFakeTXNavigation-Data, 43
 - LOAPEX::CFixedNavigation-Data, 46
 - LOAPEX::CMUXFile, 61
 - LOAPEX::CNetCDFNavigation-Data, 68
 - LOAPEX::CRex3DNavigation-Data, 73
- is_open
 - LOAPEX::CMUXFile, 61
- is_valid
 - LOAPEX::CTimeBaseMap-Engine, 82
- isleap
 - timetag.cpp, 137
- ITMAX
 - mapengine.cpp, 112
 - mapsolvers.cpp, 114
- ivector
 - nrrutil.c, 128
- JMAX
 - mapsolvers.cpp, 114
- KM_to_tm
 - LOAPEX::CTimeTag, 85
- kRX
 - LOAPEX::CGeoParameters, 51
- kTX
 - LOAPEX::CGeoParameters, 51
- LINEFEED
 - tstream.cpp, 139
- load_parameters
 - LOAPEX::CBaseMapSolver, 30
 - LOAPEX::CBrentMapSolver, 32
 - LOAPEX::CNewtonMapSolver, 70

-
- LOAPEX::CTimeBaseMap-Engine, 82
 - load_RX
 - LOAPEX::CBaseMapSolver, 30
 - LOAPEX::CBrentMapSolver, 33
 - LOAPEX::CNewtonMapSolver, 70
 - LOAPEX::CTimeBaseMap-Engine, 83
 - load_TX
 - LOAPEX::CBaseMapSolver, 30
 - LOAPEX::CBrentMapSolver, 33
 - LOAPEX::CNewtonMapSolver, 70
 - LOAPEX::CTimeBaseMap-Engine, 83
 - LOAPEX, 10
 - LOAPEX::CADFcompare, 11
 - operator(), 11
 - LOAPEX::CADFFile, 12
 - ~CADFFile, 12
 - CADFFile, 12
 - echo_header, 13
 - eof, 13
 - get_info_copy, 13
 - get_sample_rate, 13
 - get_scan, 13
 - get_start_second, 13
 - get_total_channels, 13
 - get_total_scans, 14
 - is_loaded, 14
 - print, 14
 - read_entire_channel, 14
 - LOAPEX::CADFFileAdapter, 15
 - LOAPEX::CADFFileAdapter
 - ~CADFFileAdapter, 15
 - CADFFileAdapter, 15
 - get_header, 15
 - get_sample_rate, 15
 - get_scan, 16
 - get_start_second, 16
 - get_total_channels, 16
 - get_total_scans, 16
 - LOAPEX::CADFInfo, 17
 - ~CADFInfo, 17
 - CADFInfo, 17
 - get_header, 17
 - get_id, 17
 - get_run_start, 18
 - get_sample_rate, 18
 - get_site, 18
 - get_total_phones, 18
 - get_total_scans, 18
 - print, 18
 - LOAPEX::CADFSource, 19
 - ~CADFSource, 19
 - CADFSource, 19
 - get_sample_rate, 20
 - get_scan, 20
 - get_start_second, 20
 - get_total_channels, 20
 - get_total_scans, 20
 - get_type, 21
 - LOAPEX::CBase3DNavigationData, 22
 - LOAPEX::CBase3DNavigationData
 - ~CBase3DNavigationData, 22
 - CBase3DNavigationData, 22
 - file_time_final_, 24
 - file_time_first_, 24
 - get_file_final_time, 23
 - get_file_first_time, 23
 - get_position, 23
 - get_velocity, 23
 - LOAPEX::CBaseFileAdapter, 25
 - LOAPEX::CBaseFileAdapter
 - ~CBaseFileAdapter, 25
 - get_header, 25
 - get_scan, 25
 - get_start_second, 26
 - get_total_channels, 26
 - get_total_scans, 26
 - LOAPEX::CBaseFilter, 27
 - LOAPEX::CBaseFilter
 - ~CBaseFilter, 28
 - CBaseFilter, 28
 - channel_id_, 28
 - clock, 28
 - ptheDemux_, 28
 - ptheMux_, 28
 - LOAPEX::CBaseMapSolver, 29
 - LOAPEX::CBaseMapSolver
-

- ~CBaseMapSolver, 30
- compute_map, 30
- load_parameters, 30
- load_RX, 30
- load_TX, 30
- params_, 31
- pRX_, 31
- pTX_, 31
- LOAPEX::CBrentMapSolver, 32
- LOAPEX::CBrentMapSolver
 - ~CBrentMapSolver, 32
 - CBrentMapSolver, 32
 - compute_map, 32
 - load_parameters, 32
 - load_RX, 33
 - load_TX, 33
- LOAPEX::CCompoundTime, 34
- LOAPEX::CCompoundTime
 - ~CCompoundTime, 35
 - CCompoundTime, 34
 - fraction_, 35
 - normalize, 35
 - operator+=", 35
 - operator-=", 35
 - seconds_, 35
- LOAPEX::CDemultiplexor, 36
 - ~CDemultiplexor, 36
 - CDemultiplexor, 36
 - clock, 36
 - send_sample, 36
- LOAPEX::CFakeRXNavigationData, 38
- LOAPEX::CFakeRXNavigationData
 - ~CFakeRXNavigationData, 39
 - CFakeRXNavigationData, 39
 - get_file_final_time, 39
 - get_file_first_time, 39
 - get_position, 39
 - get_velocity, 39
 - is_loaded, 40
- LOAPEX::CFakeTXNavigationData, 41
- LOAPEX::CFakeTXNavigationData
 - ~CFakeTXNavigationData, 42
 - CFakeTXNavigationData, 42
 - get_file_final_time, 42
 - get_file_first_time, 42
 - get_position, 42
 - get_velocity, 42
 - is_loaded, 43
- LOAPEX::CFixedNavigationData, 44
- LOAPEX::CFixedNavigationData
 - ~CFixedNavigationData, 45
 - CFixedNavigationData, 45
 - get_file_final_time, 45
 - get_file_first_time, 45
 - get_position, 45
 - get_velocity, 45
 - is_loaded, 46
- LOAPEX::CGappyADFStream, 47
- LOAPEX::CGappyADFStream
 - ~CGappyADFStream, 47
 - CGappyADFStream, 47
 - get_sample_rate, 48
 - get_scan, 48
 - get_start_second, 48
 - get_total_channels, 48
 - get_total_scans, 48
 - print, 48
- LOAPEX::CGenericSource, 49
- LOAPEX::CGenericSource
 - ~CGenericSource, 49
 - get_sample_rate, 49
 - get_scan, 49
 - get_start_second, 50
 - get_total_channels, 50
 - get_total_scans, 50
 - get_type, 50
- LOAPEX::CGeoParameters, 51
- LOAPEX::CGeoParameters
 - channel, 51
 - forward_bearing2APL, 51
 - forward_bearing2VLA, 51
 - kRX, 51
 - kTX, 51
 - mean_sound_speed, 52
 - range_m, 52
- LOAPEX::CInterpolatedTableFunction, 53
- LOAPEX::CInterpolatedTableFunction
 - ~CInterpolatedTableFunction, 53

- CInterpolatedTableFunction, 53
 - evaluate, 53
 - get_table_length, 53
- LOAPEX::CMultichannelFilter, 55
- LOAPEX::CMultichannelFilter
 - ~CMultichannelFilter, 55
 - add_filter, 55
 - clock, 55
 - CMultichannelFilter, 55
- LOAPEX::CMultiplexor, 57
 - ~CMultiplexor, 57
 - clock, 57
 - CMultiplexor, 57
 - receive_sample, 57
- LOAPEX::CMUXFile, 59
 - ~CMUXFile, 60
 - close, 60
 - CMUXFile, 60
 - get_descriptions, 60
 - get_header, 61
 - get_sample_rate, 61
 - get_scan, 61
 - get_start_second, 61
 - get_total_channels, 61
 - get_total_scans, 61
 - is_loaded, 61
 - is_open, 61
 - mode_type, 60
 - print, 62
 - put_description, 62
 - put_sample_rate, 62
 - put_scan, 62
 - put_start_second, 62
 - put_total_channels, 62
 - put_total_scans, 62
 - ReadOnly, 60
 - Write, 60
 - write_header, 62
- LOAPEX::CMUXFileAdapter, 64
- LOAPEX::CMUXFileAdapter
 - ~CMUXFileAdapter, 64
 - CMUXFileAdapter, 64
 - get_header, 64
 - get_sample_rate, 65
 - get_scan, 65
 - get_start_second, 65
 - get_total_channels, 65
 - get_total_scans, 65
- LOAPEX::CNetCDFNavigationData, 66
- LOAPEX::CNetCDFNavigationData
 - ~CNetCDFNavigationData, 67
 - CNetCDFNavigationData, 67
 - get_file_final_time, 67
 - get_file_first_time, 67
 - get_position, 68
 - get_velocity, 68
 - is_loaded, 68
- LOAPEX::CNewtonMapSolver, 69
- LOAPEX::CNewtonMapSolver
 - ~CNewtonMapSolver, 70
 - CNewtonMapSolver, 70
 - compute_map, 70
 - load_parameters, 70
 - load_RX, 70
 - load_TX, 70
- LOAPEX::CRex3DNavigationData, 71
- LOAPEX::CRex3DNavigationData
 - ~CRex3DNavigationData, 72
 - CRex3DNavigationData, 72
 - get_file_final_time, 72
 - get_file_first_time, 72
 - get_position, 72
 - get_velocity, 72
 - is_loaded, 73
- LOAPEX::CSingleChannelAllPassFilter, 74
- LOAPEX::CSingleChannelAllPassFilter
 - ~CSingleChannelAllPassFilter, 74
 - clock, 74
 - CSingleChannelAllPassFilter, 74
 - set_demultiplexor, 74
 - set_multiplexor, 75
- LOAPEX::CSingleChannelAllStopFilter, 76
- LOAPEX::CSingleChannelAllStopFilter
 - ~CSingleChannelAllStopFilter, 76

- clock, 76
- CSingleChannelAllStopFilter, 76
- set_demultiplexor, 76
- set_multiplexor, 77
- LOAPEX::CSingleChannelSincFilter, 78
- LOAPEX::CSingleChannelSincFilter
 - ~CSingleChannelSincFilter, 78
 - clock, 79
 - CSingleChannelSincFilter, 78
 - disable_logging, 79
 - enable_logging, 79
 - set_demultiplexor, 79
 - set_mapper, 79
 - set_multiplexor, 79
 - set_rx_start_time, 79
- LOAPEX::CTimeBaseMapEngine, 81
- LOAPEX::CTimeBaseMapEngine
 - ~CTimeBaseMapEngine, 81
 - arm, 82
 - CTimeBaseMapEngine, 81
 - dump, 82
 - get_first_time, 82
 - get_forward_time, 82
 - get_inverse_time, 82
 - get_last_time, 82
 - is_valid, 82
 - load_parameters, 82
 - load_RX, 83
 - load_TX, 83
- LOAPEX::CTimeTag, 84
- LOAPEX::CTimeTag
 - ~CTimeTag, 85
 - compute_mday, 85
 - CTimeTag, 85
 - day_, 85
 - decJD_, 85
 - decYD_, 85
 - get_components, 85
 - get_decimal_day, 85
 - get_tag, 85
 - get_year_seconds, 85
 - hour_, 85
 - KM_to_tm, 85
 - minute_, 85
 - month_, 85
 - operator<<, 85
 - operator=, 85
 - second_, 85
 - str_time_, 85
 - tm_to_KM, 85
 - year_, 85
- LOAPEX::CZeroSource, 87
- LOAPEX::CZeroSource
 - ~CZeroSource, 87
 - CZeroSource, 87
 - get_sample_rate, 87
 - get_scan, 87
 - get_start_second, 88
 - get_total_channels, 88
 - get_total_scans, 88
 - get_type, 88
- LOGNAME
 - sincfilter.cpp, 131
- lvector
 - nrutil.c, 128
- mapengine.cpp, 112
 - EPS, 112
 - GRID_SAMPLE_RATE, 112
 - ITMAX, 112
 - NRANSI, 112
 - SIGN, 112
- mapengine.hpp, 113
- mapsolvers.cpp, 114
 - BRENTTOL, 114
 - EPS, 114
 - ITMAX, 114
 - JMAX, 114
 - NEWTONTOL, 114
 - NRANSI, 114
 - SIGN, 114
- mapsolvers.hpp, 115
- matrix
 - nrutil.c, 128
- mean_sound_speed
 - LOAPEX::CGeoParameters, 52
- minute_
 - LOAPEX::CTimeTag, 85
- mode_type
 - LOAPEX::CMUXFile, 60
- month_

- LOAPEX::CTimeTag, 85
- multichannelfilter.cpp, 116
- multichannelfilter.hpp, 117
- multiplexor.cpp, 118
- multiplexor.hpp, 119
- muxadapter.cpp, 120
- muxadapter.hpp, 121
- muxfile.cpp, 122
 - clean_line_list, 122
- muxfile.hpp, 123
- navigation.cpp, 124
- navigation.hpp, 125
 - RXDEPTH, 126
 - RXDEPTHINCREMENT, 126
 - RXPERIOD, 126
 - RXRADIUS, 126
 - RXRADIUSINCREMENT, 126
 - RXTIME_START, 126
 - RXTIME_STOP, 126
 - TXDEPTH, 126
 - TXPERIOD, 126
 - TXRADIUS, 126
 - TXTIME_START, 126
 - TXTIME_STOP, 126
- NEWTONTOL
 - mapsolvers.cpp, 114
- normalize
 - LOAPEX::CCompoundTime, 35
- NR_END
 - nrutil.c, 128
- NRANSI
 - mapengine.cpp, 112
 - mapsolvers.cpp, 114
 - spline.c, 133
- nrerror
 - nrutil.c, 128
 - spline.c, 133
 - splint.c, 134
- nrutil.c, 127
 - convert_matrix, 128
 - cvector, 128
 - dmatrix, 128
 - dvector, 128
 - f3tensor, 128
 - FREE_ARG, 128
 - free_convert_matrix, 128
 - free_cvector, 128
 - free_dmatrix, 128
 - free_dvector, 128
 - free_f3tensor, 128
 - free_imatrix, 128
 - free_ivector, 128
 - free_lvector, 128
 - free_matrix, 128
 - free_submatrix, 128
 - free_vector, 128
 - imatrix, 128
 - ivector, 128
 - lvector, 128
 - matrix, 128
 - NR_END, 128
 - nrerror, 128
 - submatrix, 128
 - vector, 128
- operator()
 - LOAPEX::CADFcompare, 11
- operator+
 - compoundtime.cpp, 105
 - compoundtime.hpp, 106
- operator+=
 - LOAPEX::CCompoundTime, 35
- operator-
 - compoundtime.cpp, 105
 - compoundtime.hpp, 106
- operator-=
 - LOAPEX::CCompoundTime, 35
- operator<<
 - LOAPEX::CTimeTag, 85
- operator=
 - LOAPEX::CTimeTag, 85
- operator>>
 - tstream, 92
- parameters.hpp, 129
- params_
 - LOAPEX::CBaseMapSolver, 31
- print
 - LOAPEX::CADFFile, 14
 - LOAPEX::CADFInfo, 18

- LOAPEX::CGappyADFStream, 48
- LOAPEX::CMUXFile, 62
- pRX_
 - LOAPEX::CBaseMapSolver, 31
- ptheDemux_
 - LOAPEX::CBaseFilter, 28
- ptheMux_
 - LOAPEX::CBaseFilter, 28
- pTX_
 - LOAPEX::CBaseMapSolver, 31
- put_description
 - LOAPEX::CMUXFile, 62
- put_sample_rate
 - LOAPEX::CMUXFile, 62
- put_scan
 - LOAPEX::CMUXFile, 62
- put_start_second
 - LOAPEX::CMUXFile, 62
- put_total_channels
 - LOAPEX::CMUXFile, 62
- put_total_scans
 - LOAPEX::CMUXFile, 62
- range_m
 - LOAPEX::CGeoParameters, 52
- read_entire_channel
 - LOAPEX::CADFFFile, 14
- ReadOnly
 - LOAPEX::CMUXFile, 60
- receive_sample
 - LOAPEX::CMultiplexor, 57
- refman.dox, 130
- RXDEPTH
 - navigation.hpp, 126
- RXDEPTHINCREMENT
 - navigation.hpp, 126
- RXPERIOD
 - navigation.hpp, 126
- RXRADIUS
 - navigation.hpp, 126
- RXRADIUSINCREMENT
 - navigation.hpp, 126
- RXTIME_START
 - navigation.hpp, 126
- RXTIME_STOP
 - navigation.hpp, 126
- navigation.hpp, 126
- SBUFFERSIZE
 - adffile.cpp, 95
- second_
 - LOAPEX::CTimeTag, 85
- seconds_
 - LOAPEX::CCompoundTime, 35
- seekt
 - tstream, 92
- send_sample
 - LOAPEX::CDemultiplexor, 36
- set_demultiplexor
 - LOAPEX::CSingleChannelAllPassFilter, 74
 - LOAPEX::CSingleChannelAllStopFilter, 76
 - LOAPEX::CSingleChannelSincFilter, 79
- set_mapper
 - LOAPEX::CSingleChannelSincFilter, 79
- set_multiplexor
 - LOAPEX::CSingleChannelAllPassFilter, 75
 - LOAPEX::CSingleChannelAllStopFilter, 77
 - LOAPEX::CSingleChannelSincFilter, 79
- set_rx_start_time
 - LOAPEX::CSingleChannelSincFilter, 79
- SIGN
 - mapengine.cpp, 112
 - mapsolvers.cpp, 114
- sincfilter.cpp, 131
 - LOGNAME, 131
- sincfilter.hpp, 132
- SPACE
 - tstream.cpp, 139
- spline
 - spline.c, 133
 - tablefunction.cpp, 135
- spline.c, 133
 - dvector, 133
 - free_dvector, 133

- NRANSI, 133
- nrerror, 133
- spline, 133
- splint
 - splint.c, 134
 - tablefunction.cpp, 135
- splint.c, 134
 - nrerror, 134
 - splint, 134
- str_time_
 - LOAPEX::CTimeTag, 85
- submatrix
 - nrutil.c, 128
- TAB
 - tstream.cpp, 139
- tablefunction.cpp, 135
 - spline, 135
 - splint, 135
- tablefunction.hpp, 136
- timetag.cpp, 137
 - decYD_to_month, 137
 - isleap, 137
- timetag.hpp, 138
- tm_to_KM
 - LOAPEX::CTimeTag, 85
- TOKEN_BUFFER_SIZE
 - adffile.cpp, 95
- tstream, 92
 - ~tstream, 92
 - eos, 92
 - operator>>, 92
 - seekt, 92
 - tstream, 92
- tstream.cpp, 139
 - CR, 139
 - LINEFEED, 139
 - SPACE, 139
 - TAB, 139
- tstream.hpp, 140
- TXDEPTH
 - navigation.hpp, 126
- TXPERIOD
 - navigation.hpp, 126
- TXRADIUS
 - navigation.hpp, 126
- TXTIME_START
 - navigation.hpp, 126
- TXTIME_STOP
 - navigation.hpp, 126
- vector
 - nrutil.c, 128
- Write
 - LOAPEX::CMUXFile, 60
- write_header
 - LOAPEX::CMUXFile, 62
- x
 - dEuclideanVector_t, 89
- y
 - dEuclideanVector_t, 89
- year_
 - LOAPEX::CTimeTag, 85
- z
 - dEuclideanVector_t, 89