

CUDAProb3++

Generated by Doxygen 1.8.11

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	cuDprob3::CpuPropagator< FLOAT_T > Class Template Reference	5
3.1.1	Detailed Description	6
3.1.2	Constructor & Destructor Documentation	6
3.1.2.1	CpuPropagator(int n_cosines, int n_energies, int threads)	6
3.1.2.2	CpuPropagator(const CpuPropagator &other)	6
3.1.2.3	CpuPropagator(CpuPropagator &&other)	7
3.1.3	Member Function Documentation	7
3.1.3.1	calculateProbabilities(NeutrinoType type) override	7
3.1.3.2	getProbability(int index_cosine, int index_energy, ProbType t) override	7
3.1.3.3	operator=(const CpuPropagator &other)	7
3.1.3.4	operator=(CpuPropagator &&other)	8
3.2	cuDprob3::CudaPropagator< FLOAT_T > Class Template Reference	8
3.2.1	Detailed Description	10
3.2.2	Constructor & Destructor Documentation	10
3.2.2.1	CudaPropagator(int nc, int ne)	10
3.2.2.2	CudaPropagator(const std::vector< int > &ids, int nc, int ne, bool failOnInvalid↵ Id=true)	10
3.2.2.3	CudaPropagator(CudaPropagator &&other)	10

3.2.3	Member Function Documentation	11
3.2.3.1	calculateProbabilities(NeutrinoType type) override	11
3.2.3.2	getProbability(int index_cosine, int index_energy, ProbType t) override	11
3.2.3.3	operator=(CudaPropagator &&other)	11
3.2.3.4	setCosineList(const std::vector< FLOAT_T > &list) override	11
3.2.3.5	setDensity(const std::vector< FLOAT_T > &radii, const std::vector< FLOAT_T > &rhos) override	12
3.2.3.6	setDensityFromFile(const std::string &filename) override	12
3.2.3.7	setEnergyList(const std::vector< FLOAT_T > &list) override	12
3.2.3.8	setMNSMatrix(FLOAT_T theta12, FLOAT_T theta13, FLOAT_T theta23, FLOAT_T AT_T dCP) override	13
3.2.3.9	setNeutrinoMasses(FLOAT_T dm12sq, FLOAT_T dm23sq) override	13
3.2.3.10	setProductionHeight(FLOAT_T heightKM) override	13
3.3	cudaprob3::CudaPropagatorSingle< FLOAT_T > Class Template Reference	14
3.3.1	Detailed Description	15
3.3.2	Constructor & Destructor Documentation	15
3.3.2.1	CudaPropagatorSingle(int id, int n_cosines_, int n_energies_)	15
3.3.2.2	CudaPropagatorSingle(int n_cosines, int n_energies)	15
3.3.2.3	CudaPropagatorSingle(CudaPropagatorSingle &&other)	16
3.3.3	Member Function Documentation	16
3.3.3.1	calculateProbabilities(NeutrinoType type) override	16
3.3.3.2	getProbability(int index_cosine, int index_energy, ProbType t) override	16
3.3.3.3	operator=(CudaPropagatorSingle &&other)	16
3.3.3.4	setCosineList(const std::vector< FLOAT_T > &list) override	17
3.3.3.5	setDensity(const std::vector< FLOAT_T > &radii_, const std::vector< FLOAT_T > &rhos_) override	17
3.3.3.6	setEnergyList(const std::vector< FLOAT_T > &list) override	17
3.4	cudaprob3::Propagator< FLOAT_T > Class Template Reference	17
3.4.1	Detailed Description	19
3.4.2	Constructor & Destructor Documentation	19
3.4.2.1	Propagator(int n_cosines, int n_energies)	19
3.4.2.2	Propagator(const Propagator &other)	19

3.4.2.3	Propagator(Propagator &&other)	19
3.4.3	Member Function Documentation	19
3.4.3.1	calculateProbabilities(NeutrinoType type)=0	20
3.4.3.2	getProbability(int index_cosine, int index_energy, ProbType t)=0	20
3.4.3.3	operator=(const Propagator &other)	20
3.4.3.4	operator=(Propagator &&other)	20
3.4.3.5	setCosineList(const std::vector< FLOAT_T > &list)	21
3.4.3.6	setDensity(const std::vector< FLOAT_T > &radii_, const std::vector< FLOAT_T > &rhos_)	21
3.4.3.7	setDensityFromFile(const std::string &filename)	21
3.4.3.8	setEnergyList(const std::vector< FLOAT_T > &list)	21
3.4.3.9	setMNSMatrix(FLOAT_T theta12, FLOAT_T theta13, FLOAT_T theta23, FLOAT_T theta14, FLOAT_T theta24, FLOAT_T theta34, FLOAT_T dCP)	22
3.4.3.10	setNeutrinoMasses(FLOAT_T dm12sq, FLOAT_T dm23sq)	22
3.4.3.11	setProductionHeight(FLOAT_T heightKM)	22
	Index	25

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

<code>cuDaprob3::Propagator< FLOAT_T ></code>	17
<code>cuDaprob3::CpuPropagator< FLOAT_T ></code>	5
<code>cuDaprob3::CudaPropagator< FLOAT_T ></code>	8
<code>cuDaprob3::CudaPropagatorSingle< FLOAT_T ></code>	14

Chapter 2

Class Index

2.1 Class List

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

cudaprob3::CpuPropagator< FLOAT_T >	
Multi-threaded CPU neutrino propagation. Derived from Propagator	5
cudaprob3::CudaPropagator< FLOAT_T >	
Multi-GPU neutrino propagation. Derived from Propagator	8
cudaprob3::CudaPropagatorSingle< FLOAT_T >	
Single-GPU neutrino propagation. Derived from Propagator	14
cudaprob3::Propagator< FLOAT_T >	
Abstract base class of the library which sets up input parameter on the host. Concrete implementation of calculations is provided in derived classes	17

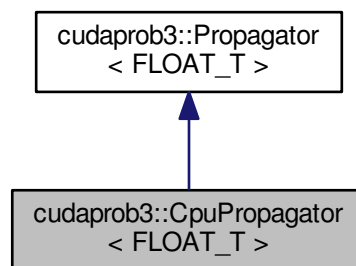
Chapter 3

Class Documentation

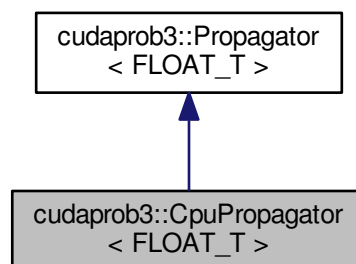
3.1 cudaprob3::CpuPropagator< FLOAT_T > Class Template Reference

Multi-threaded CPU neutrino propagation. Derived from [Propagator](#).

Inheritance diagram for cudaprob3::CpuPropagator< FLOAT_T >:



Collaboration diagram for cudaprob3::CpuPropagator< FLOAT_T >:



Public Member Functions

- [CpuPropagator](#) (int *n_cosines*, int *n_energies*, int *threads*)
Constructor.
- [CpuPropagator](#) (const [CpuPropagator](#) &*other*)
Copy constructor.
- [CpuPropagator](#) ([CpuPropagator](#) &&*other*)
Move constructor.
- [CpuPropagator](#) & [operator=](#) (const [CpuPropagator](#) &*other*)
Copy assignment operator.
- [CpuPropagator](#) & [operator=](#) ([CpuPropagator](#) &&*other*)
Move assignment operator.
- void [calculateProbabilities](#) (NeutrinoType *type*) override
Calculate the probability of each cell.
- [FLOAT_T](#) [getProbability](#) (int *index_cosine*, int *index_energy*, ProbType *t*) override
get oscillation weight for specific cosine and energy

3.1.1 Detailed Description

```
template<class FLOAT_T>
class cudaprob3::CpuPropagator< FLOAT_T >
```

Multi-threaded CPU neutrino propagation. Derived from [Propagator](#).

Parameters

<i>FLOAT_T</i>	The floating point type to use for calculations, i.e float, double
----------------	--

3.1.2 Constructor & Destructor Documentation

3.1.2.1 `template<class FLOAT_T > cudaprob3::CpuPropagator< FLOAT_T >::CpuPropagator (int n_cosines, int n_energies, int threads)` `[inline]`

Constructor.

Parameters

<i>n_cosines</i>	Number cosine bins
<i>n_energies</i>	Number of energy bins
<i>threads</i>	Number of threads

3.1.2.2 `template<class FLOAT_T > cudaprob3::CpuPropagator< FLOAT_T >::CpuPropagator (const CpuPropagator< FLOAT_T > &other)` `[inline]`

Copy constructor.

Parameters

<i>other</i>	
--------------	--

3.1.2.3 `template<class FLOAT_T > cudaprob3::CpuPropagator< FLOAT_T >::CpuPropagator (CpuPropagator< FLOAT_T > && other)` `[inline]`

Move constructor.

Parameters

<i>other</i>	
--------------	--

3.1.3 Member Function Documentation

3.1.3.1 `template<class FLOAT_T > void cudaprob3::CpuPropagator< FLOAT_T >::calculateProbabilities (NeutrinoType type)` `[inline]`, `[override]`, `[virtual]`

Calculate the probability of each cell.

Parameters

<i>type</i>	Neutrino or Antineutrino
-------------	--------------------------

Implements [cudaprob3::Propagator< FLOAT_T >](#).

3.1.3.2 `template<class FLOAT_T > FLOAT_T cudaprob3::CpuPropagator< FLOAT_T >::getProbability (int index_cosine, int index_energy, ProbType t)` `[inline]`, `[override]`, `[virtual]`

get oscillation weight for specific cosine and energy

Parameters

<i>index_cosine</i>	Cosine bin index (zero based)
<i>index_energy</i>	Energy bin index (zero based)
<i>t</i>	Specify which probability P(i->j)

Implements [cudaprob3::Propagator< FLOAT_T >](#).

3.1.3.3 `template<class FLOAT_T > CpuPropagator& cudaprob3::CpuPropagator< FLOAT_T >::operator= (const CpuPropagator< FLOAT_T > & other)` `[inline]`

Copy assignment operator.

Parameters

<i>other</i>	
--------------	--

3.1.3.4 `template<class FLOAT_T > CpuPropagator& cudaprob3::CpuPropagator< FLOAT_T >::operator= (CpuPropagator< FLOAT_T > && other) [inline]`

Move assignment operator.

Parameters

<i>other</i>	
--------------	--

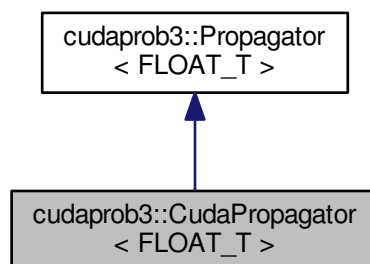
The documentation for this class was generated from the following file:

- `cpupropagator.hpp`

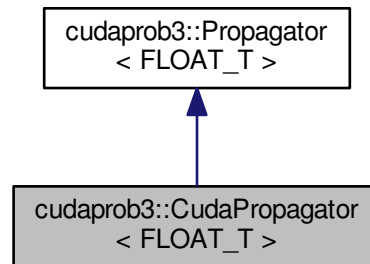
3.2 `cudaprob3::CudaPropagator< FLOAT_T >` Class Template Reference

Multi-GPU neutrino propagation. Derived from [Propagator](#).

Inheritance diagram for `cudaprob3::CudaPropagator< FLOAT_T >`:



Collaboration diagram for cudaprob3::CudaPropagator< FLOAT_T >:



Public Member Functions

- [CudaPropagator](#) (int nc, int ne)
Single GPU constructor for device id 0.
- [CudaPropagator](#) (const std::vector< int > &ids, int nc, int ne, bool failOnInvalidId=true)
Constructor.
- [CudaPropagator](#) ([CudaPropagator](#) &&other)
Move constructor.
- [CudaPropagator](#) & [operator=](#) ([CudaPropagator](#) &&other)
Move assignment operator.
- void [setDensityFromFile](#) (const std::string &filename) override
Set density information from file.
- void [setDensity](#) (const std::vector< FLOAT_T > &radii, const std::vector< FLOAT_T > &rhos) override
Set density information from arrays.
- void [setNeutrinoMasses](#) (FLOAT_T dm12sq, FLOAT_T dm23sq) override
Set neutrino mass differences $(m_{i,j})^2$ in $(\text{eV})^2$. no assumptions about mass hierarchy are made.
- void [setMNSMatrix](#) (FLOAT_T theta12, FLOAT_T theta13, FLOAT_T theta23, FLOAT_T dCP) override
Set mixing angles and cp phase in radians.
- void [setEnergyList](#) (const std::vector< FLOAT_T > &list) override
Set the energy bins. Energies are given in GeV.
- void [setCosineList](#) (const std::vector< FLOAT_T > &list) override
Set cosine bins. Cosines are given in radians.
- void [setProductionHeight](#) (FLOAT_T heightKM) override
Set production height in km of neutrinos.
- void [calculateProbabilities](#) (NeutrinoType type) override
Calculate the probability of each cell.
- FLOAT_T [getProbability](#) (int index_cosine, int index_energy, ProbType t) override
get oscillation weight for specific cosine and energy

3.2.1 Detailed Description

```
template<class FLOAT_T>
class cudaprob3::CudaPropagator< FLOAT_T >
```

Multi-GPU neutrino propagation. Derived from [Propagator](#).

This is essentially a wrapper around multiple [CudaPropagatorSingle](#) instances, one per used GPU. Most of the setters and calculation functions simply call the appropriate function for each GPU.

Parameters

<i>Float_T</i>	The floating point type to use for calculations, i.e float, double
----------------	--

3.2.2 Constructor & Destructor Documentation

```
3.2.2.1 template<class FLOAT_T > cudaprob3::CudaPropagator< FLOAT_T >::CudaPropagator ( int nc, int ne )
[inline]
```

Single GPU constructor for device id 0.

Parameters

<i>nc</i>	Number cosine bins
<i>ne</i>	Number of energy bins

```
3.2.2.2 template<class FLOAT_T > cudaprob3::CudaPropagator< FLOAT_T >::CudaPropagator ( const
std::vector< int > & ids, int nc, int ne, bool failOnInvalidId = true ) [inline]
```

Constructor.

Parameters

<i>ids</i>	List of device ids of the GPUs to use
<i>nc</i>	Number cosine bins
<i>ne</i>	Number of energy bins
<i>failOnInvalidId</i>	If true, throw exception if ids contains an invalid device id

```
3.2.2.3 template<class FLOAT_T > cudaprob3::CudaPropagator< FLOAT_T >::CudaPropagator (
CudaPropagator< FLOAT_T > && other ) [inline]
```

Move constructor.

Parameters

<i>other</i>	
--------------	--

3.2.3 Member Function Documentation

3.2.3.1 `template<class FLOAT_T > void cudaprob3::CudaPropagator< FLOAT_T >::calculateProbabilities (NeutrinoType type)` `[inline]`, `[override]`, `[virtual]`

Calculate the probability of each cell.

Parameters

<i>type</i>	Neutrino or Antineutrino
-------------	--------------------------

Implements [cudaprob3::Propagator< FLOAT_T >](#).

3.2.3.2 `template<class FLOAT_T > FLOAT_T cudaprob3::CudaPropagator< FLOAT_T >::getProbability (int index_cosine, int index_energy, ProbType t)` `[inline]`, `[override]`, `[virtual]`

get oscillation weight for specific cosine and energy

Parameters

<i>index_cosine</i>	Cosine bin index (zero based)
<i>index_energy</i>	Energy bin index (zero based)
<i>t</i>	Specify which probability P(i->j)

Implements [cudaprob3::Propagator< FLOAT_T >](#).

3.2.3.3 `template<class FLOAT_T > CudaPropagator& cudaprob3::CudaPropagator< FLOAT_T >::operator= (CudaPropagator< FLOAT_T > && other)` `[inline]`

Move assignment operator.

Parameters

<i>other</i>	
--------------	--

3.2.3.4 `template<class FLOAT_T > void cudaprob3::CudaPropagator< FLOAT_T >::setCosineList (const std::vector< FLOAT_T > & list)` `[inline]`, `[override]`, `[virtual]`

Set cosine bins. Cosines are given in radians.

Parameters

<i>list</i>	Cosine list
-------------	-------------

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

3.2.3.5 `template<class FLOAT_T > void cudaprob3::CudaPropagator< FLOAT_T >::setDensity (const std::vector< FLOAT_T > & radii_, const std::vector< FLOAT_T > & rhos_) [inline],[override],[virtual]`

Set density information from arrays.

`radii_` and `rhos_` must be same size. both `radii_` and `rhos_` must be sorted, in the same order. The density (g/cm^3) at a distance (km) from the center of the sphere between `radii_[i]`, exclusive, and `radii_[j]`, inclusive, $i < j$ is assumed to be `rhos_[j]`

Parameters

<i>radii_</i> ↔ —	List of radii
<i>rhos_</i> ↔ —	List of densities

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

3.2.3.6 `template<class FLOAT_T > void cudaprob3::CudaPropagator< FLOAT_T >::setDensityFromFile (const std::string & filename) [inline],[override],[virtual]`

Set density information from file.

File must contain two columns where the first column contains the radius (km) and the second column contains the density (g/cm^3). The first row must have the radius 0. The last row must have to radius of the sphere

Parameters

<i>filename</i>	File with density information
-----------------	-------------------------------

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

3.2.3.7 `template<class FLOAT_T > void cudaprob3::CudaPropagator< FLOAT_T >::setEnergyList (const std::vector< FLOAT_T > & list) [inline],[override],[virtual]`

Set the energy bins. Energies are given in GeV.

Parameters

<i>list</i>	Energy list
-------------	-------------

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

3.2.3.8 `template<class FLOAT_T> void cudaprob3::CudaPropagator< FLOAT_T >::setMNSMatrix (FLOAT_T theta12,
FLOAT_T theta13, FLOAT_T theta23, FLOAT_T dCP)` `[inline]`, `[override]`, `[virtual]`

Set mixing angles and cp phase in radians.

Parameters

<i>theta12</i>	
<i>theta13</i>	
<i>theta23</i>	
<i>dCP</i>	

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

3.2.3.9 `template<class FLOAT_T> void cudaprob3::CudaPropagator< FLOAT_T >::setNeutrinoMasses (FLOAT_T
dm12sq, FLOAT_T dm23sq)` `[inline]`, `[override]`, `[virtual]`

Set neutrino mass differences $(m_{i_j})^2$ in $(\text{eV})^2$. no assumptions about mass hierarchy are made.

Parameters

<i>dm12sq</i>	
<i>dm23sq</i>	

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

3.2.3.10 `template<class FLOAT_T> void cudaprob3::CudaPropagator< FLOAT_T >::setProductionHeight (FLOAT_T
heightKM)` `[inline]`, `[override]`, `[virtual]`

Set production height in km of neutrinos.

Adds a layer of length heightKM with zero density to the density model

Parameters

<i>heightKM</i>	Set neutrino production height
-----------------	--------------------------------

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

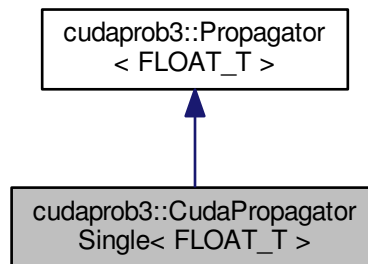
The documentation for this class was generated from the following file:

- `cudapropagator.cuh`

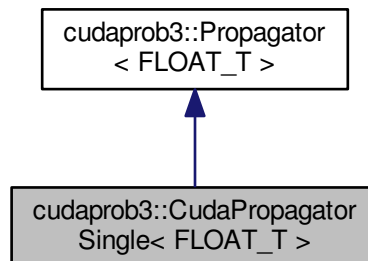
3.3 cudaprob3::CudaPropagatorSingle< FLOAT_T > Class Template Reference

Single-GPU neutrino propagation. Derived from [Propagator](#).

Inheritance diagram for cudaprob3::CudaPropagatorSingle< FLOAT_T >:



Collaboration diagram for cudaprob3::CudaPropagatorSingle< FLOAT_T >:



Public Member Functions

- [CudaPropagatorSingle](#) (int id, int n_cosines_, int n_energies_)
Constructor.
- [CudaPropagatorSingle](#) (int n_cosines, int n_energies)
Constructor which uses device id 0.
- [~CudaPropagatorSingle](#) ()
Destructor.
- [CudaPropagatorSingle](#) (CudaPropagatorSingle &&other)
Move constructor.
- [CudaPropagatorSingle](#) & operator= (CudaPropagatorSingle &&other)
Move assignment operator.

- void [setDensity](#) (const std::vector< FLOAT_T > &radii_, const std::vector< FLOAT_T > &rhos_) override
Set density information from arrays.
- void [setEnergyList](#) (const std::vector< FLOAT_T > &list) override
Set the energy bins. Energies are given in GeV.
- void [setCosineList](#) (const std::vector< FLOAT_T > &list) override
Set cosine bins. Cosines are given in radians.
- void [calculateProbabilities](#) (NeutrinoType type) override
Calculate the probability of each cell.
- FLOAT_T [getProbability](#) (int index_cosine, int index_energy, ProbType t) override
get oscillation weight for specific cosine and energy

3.3.1 Detailed Description

```
template<class FLOAT_T>
class cudaprob3::CudaPropagatorSingle< FLOAT_T >
```

Single-GPU neutrino propagation. Derived from [Propagator](#).

Parameters

<i>FLOAT_T</i>	The floating point type to use for calculations, i.e float, double
----------------	--

3.3.2 Constructor & Destructor Documentation

3.3.2.1 `template<class FLOAT_T > cudaprob3::CudaPropagatorSingle< FLOAT_T >::CudaPropagatorSingle (int id, int n_cosines_, int n_energies_) [inline]`

Constructor.

Parameters

<i>id</i>	device id of the GPU to use
<i>n_cosines_</i>	Number cosine bins
<i>n_energies_</i>	Number of energy bins

3.3.2.2 `template<class FLOAT_T > cudaprob3::CudaPropagatorSingle< FLOAT_T >::CudaPropagatorSingle (int n_cosines, int n_energies) [inline]`

Constructor which uses device id 0.

Parameters

<i>n_cosines</i>	Number cosine bins
<i>n_energies</i>	Number of energy bins

3.3.2.3 `template<class FLOAT_T > cudaprob3::CudaPropagatorSingle< FLOAT_T >::CudaPropagatorSingle (CudaPropagatorSingle< FLOAT_T > && other) [inline]`

Move constructor.

Parameters

<i>other</i>	
--------------	--

3.3.3 Member Function Documentation

3.3.3.1 `template<class FLOAT_T > void cudaprob3::CudaPropagatorSingle< FLOAT_T >::calculateProbabilities (NeutrinoType type) [inline],[override],[virtual]`

Calculate the probability of each cell.

Parameters

<i>type</i>	Neutrino or Antineutrino
-------------	--------------------------

Implements [cudaprob3::Propagator< FLOAT_T >](#).

3.3.3.2 `template<class FLOAT_T > FLOAT_T cudaprob3::CudaPropagatorSingle< FLOAT_T >::getProbability (int index_cosine, int index_energy, ProbType t) [inline],[override],[virtual]`

get oscillation weight for specific cosine and energy

Parameters

<i>index_cosine</i>	Cosine bin index (zero based)
<i>index_energy</i>	Energy bin index (zero based)
<i>t</i>	Specify which probability P(i->j)

Implements [cudaprob3::Propagator< FLOAT_T >](#).

3.3.3.3 `template<class FLOAT_T > CudaPropagatorSingle& cudaprob3::CudaPropagatorSingle< FLOAT_T >::operator= (CudaPropagatorSingle< FLOAT_T > && other) [inline]`

Move assignment operator.

Parameters

<i>other</i>	
--------------	--

3.3.3.4 `template<class FLOAT_T> void cudaprob3::CudaPropagatorSingle< FLOAT_T >::setCosineList (const std::vector< FLOAT_T > & list) [inline],[override],[virtual]`

Set cosine bins. Cosines are given in radians.

Parameters

<i>list</i>	Cosine list
-------------	-------------

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

3.3.3.5 `template<class FLOAT_T> void cudaprob3::CudaPropagatorSingle< FLOAT_T >::setDensity (const std::vector< FLOAT_T > & radii_, const std::vector< FLOAT_T > & rhos_) [inline],[override],[virtual]`

Set density information from arrays.

`radii_` and `rhos_` must be same size. both `radii_` and `rhos_` must be sorted, in the same order. The density (g/cm³) at a distance (km) from the center of the sphere between `radii_[i]`, exclusive, and `radii_[j]`, inclusive, $i < j$ is assumed to be `rhos_[j]`

Parameters

<i>radii_</i> ↔ —	List of radii
<i>rhos_</i> ↔ —	List of densities

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

3.3.3.6 `template<class FLOAT_T> void cudaprob3::CudaPropagatorSingle< FLOAT_T >::setEnergyList (const std::vector< FLOAT_T > & list) [inline],[override],[virtual]`

Set the energy bins. Energies are given in GeV.

Parameters

<i>list</i>	Energy list
-------------	-------------

Reimplemented from [cudaprob3::Propagator< FLOAT_T >](#).

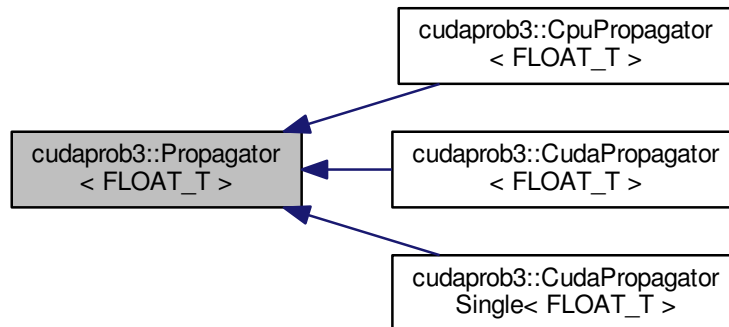
The documentation for this class was generated from the following file:

- `cudapropagator.cuh`

3.4 cudaprob3::Propagator< FLOAT_T > Class Template Reference

Abstract base class of the library which sets up input parameter on the host. Concrete implementation of calculations is provided in derived classes.

Inheritance diagram for `cuaprob3::Propagator< FLOAT_T >`:



Public Member Functions

- [Propagator](#) (int n_cosines, int n_energies)
Constructor.
- [Propagator](#) (const [Propagator](#) &other)
Copy constructor.
- [Propagator](#) ([Propagator](#) &&other)
Move constructor.
- [Propagator](#) & [operator=](#) (const [Propagator](#) &other)
Copy assignment operator.
- [Propagator](#) & [operator=](#) ([Propagator](#) &&other)
Move assignment operator.
- virtual void [setDensity](#) (const std::vector< FLOAT_T > &radii_, const std::vector< FLOAT_T > &rhos_)
Set density information from arrays.
- virtual void [setDensityFromFile](#) (const std::string &filename)
Set density information from file.
- virtual void [setMNSMatrix](#) (FLOAT_T theta12, FLOAT_T theta13, FLOAT_T theta23, FLOAT_T dCP)
Set mixing angles and cp phase in radians.
- virtual void [setNeutrinoMasses](#) (FLOAT_T dm12sq, FLOAT_T dm23sq)
Set neutrino mass differences $(m_{i_j})^2$ in $(\text{eV})^2$. no assumptions about mass hierarchy are made.
- virtual void [setEnergyList](#) (const std::vector< FLOAT_T > &list)
Set the energy bins. Energies are given in GeV.
- virtual void [setCosineList](#) (const std::vector< FLOAT_T > &list)
Set cosine bins. Cosines are given in radians.
- virtual void [setProductionHeight](#) (FLOAT_T heightKM)
Set production height in km of neutrinos.
- virtual void [calculateProbabilities](#) (NeutrinoType type)=0
Calculate the probability of each cell.
- virtual FLOAT_T [getProbability](#) (int index_cosine, int index_energy, ProbType t)=0
get oscillation weight for specific cosine and energy

3.4.1 Detailed Description

```
template<class FLOAT_T>
class cudaprob3::Propagator< FLOAT_T >
```

Abstract base class of the library which sets up input parameter on the host. Concrete implementation of calculations is provided in derived classes.

Parameters

<i>Float_T</i>	The floating point type to use for calculations, i.e float, double
----------------	--

3.4.2 Constructor & Destructor Documentation

```
3.4.2.1 template<class FLOAT_T > cudaprob3::Propagator< FLOAT_T >::Propagator ( int n_cosines, int n_energies ) [inline]
```

Constructor.

Parameters

<i>n_cosines</i>	Number cosine bins
<i>n_energies</i>	Number of energy bins

```
3.4.2.2 template<class FLOAT_T > cudaprob3::Propagator< FLOAT_T >::Propagator ( const Propagator<
FLOAT_T > & other ) [inline]
```

Copy constructor.

Parameters

<i>other</i>	
--------------	--

```
3.4.2.3 template<class FLOAT_T > cudaprob3::Propagator< FLOAT_T >::Propagator ( Propagator< FLOAT_T >
&& other ) [inline]
```

Move constructor.

Parameters

<i>other</i>	
--------------	--

3.4.3 Member Function Documentation

3.4.3.1 `template<class FLOAT_T > virtual void cudaprob3::Propagator< FLOAT_T >::calculateProbabilities (NeutrinoType type) [pure virtual]`

Calculate the probability of each cell.

Parameters

<i>type</i>	Neutrino or Antineutrino
-------------	--------------------------

Implemented in [cudaprob3::CudaPropagator< FLOAT_T >](#), [cudaprob3::CudaPropagatorSingle< FLOAT_T >](#), and [cudaprob3::CpuPropagator< FLOAT_T >](#).

3.4.3.2 `template<class FLOAT_T > virtual FLOAT_T cudaprob3::Propagator< FLOAT_T >::getProbability (int index_cosine, int index_energy, ProbType t) [pure virtual]`

get oscillation weight for specific cosine and energy

Parameters

<i>index_cosine</i>	Cosine bin index (zero based)
<i>index_energy</i>	Energy bin index (zero based)
<i>t</i>	Specify which probability P(i->j)

Implemented in [cudaprob3::CudaPropagator< FLOAT_T >](#), [cudaprob3::CudaPropagatorSingle< FLOAT_T >](#), and [cudaprob3::CpuPropagator< FLOAT_T >](#).

3.4.3.3 `template<class FLOAT_T > Propagator& cudaprob3::Propagator< FLOAT_T >::operator= (const Propagator< FLOAT_T > &other) [inline]`

Copy assignment operator.

Parameters

<i>other</i>	
--------------	--

3.4.3.4 `template<class FLOAT_T > Propagator& cudaprob3::Propagator< FLOAT_T >::operator= (Propagator< FLOAT_T > &&other) [inline]`

Move assignment operator.

Parameters

<i>other</i>	
--------------	--

3.4.3.5 `template<class FLOAT_T> virtual void cudaprob3::Propagator< FLOAT_T >::setCosineList (const std::vector< FLOAT_T > & list) [inline],[virtual]`

Set cosine bins. Cosines are given in radians.

Parameters

<i>list</i>	Cosine list
-------------	-------------

Reimplemented in [cudaprob3::CudaPropagator< FLOAT_T >](#), and [cudaprob3::CudaPropagatorSingle< FLOAT_T >](#).

3.4.3.6 `template<class FLOAT_T> virtual void cudaprob3::Propagator< FLOAT_T >::setDensity (const std::vector< FLOAT_T > & radii_, const std::vector< FLOAT_T > & rhos_) [inline],[virtual]`

Set density information from arrays.

`radii_` and `rhos_` must be same size. both `radii_` and `rhos_` must be sorted, in the same order. The density (g/cm³) at a distance (km) from the center of the sphere between `radii_[i]`, exclusive, and `radii_[j]`, inclusive, $i < j$ is assumed to be `rhos_[j]`

Parameters

<i>radii_</i>	List of radii
—	
<i>rhos_</i>	List of densities
—	

Reimplemented in [cudaprob3::CudaPropagator< FLOAT_T >](#), and [cudaprob3::CudaPropagatorSingle< FLOAT_T >](#).

3.4.3.7 `template<class FLOAT_T> virtual void cudaprob3::Propagator< FLOAT_T >::setDensityFromFile (const std::string & filename) [inline],[virtual]`

Set density information from file.

File must contain two columns where the first column contains the radius (km) and the second column contains the density (g/cm³). The first row must have the radius 0. The last row must have to radius of the sphere

Parameters

<i>filename</i>	File with density information
-----------------	-------------------------------

Reimplemented in [cudaprob3::CudaPropagator< FLOAT_T >](#).

3.4.3.8 `template<class FLOAT_T> virtual void cudaprob3::Propagator< FLOAT_T >::setEnergyList (const std::vector< FLOAT_T > & list) [inline],[virtual]`

Set the energy bins. Energies are given in GeV.

Parameters

<i>list</i>	Energy list
-------------	-------------

Reimplemented in [cudaprob3::CudaPropagator< FLOAT_T >](#), and [cudaprob3::CudaPropagatorSingle< FLOAT_T >](#).

3.4.3.9 `template<class FLOAT_T > virtual void cudaprob3::Propagator< FLOAT_T >::setMNSMatrix (FLOAT_T theta12, FLOAT_T theta13, FLOAT_T theta23, FLOAT_T dCP)` `[inline]`, `[virtual]`

Set mixing angles and cp phase in radians.

Parameters

<i>theta12</i>	
<i>theta13</i>	
<i>theta23</i>	
<i>dCP</i>	

Reimplemented in [cudaprob3::CudaPropagator< FLOAT_T >](#).

3.4.3.10 `template<class FLOAT_T > virtual void cudaprob3::Propagator< FLOAT_T >::setNeutrinoMasses (FLOAT_T dm12sq, FLOAT_T dm23sq)` `[inline]`, `[virtual]`

Set neutrino mass differences $(m_{ij})^2$ in $(\text{eV})^2$. no assumptions about mass hierarchy are made.

Parameters

<i>dm12sq</i>	
<i>dm23sq</i>	

Reimplemented in [cudaprob3::CudaPropagator< FLOAT_T >](#).

3.4.3.11 `template<class FLOAT_T > virtual void cudaprob3::Propagator< FLOAT_T >::setProductionHeight (FLOAT_T heightKM)` `[inline]`, `[virtual]`

Set production height in km of neutrinos.

Adds a layer of length *heightKM* with zero density to the density model

Parameters

<i>heightKM</i>	Set neutrino production height
-----------------	--------------------------------

Reimplemented in [cudaprob3::CudaPropagator< FLOAT_T >](#).

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

- [propagator.hpp](#)

Index

- calculateProbabilities
 - `cudaProb3::CpuPropagator`, 7
 - `cudaProb3::CudaPropagator`, 11
 - `cudaProb3::CudaPropagatorSingle`, 16
 - `cudaProb3::Propagator`, 19
- CpuPropagator
 - `cudaProb3::CpuPropagator`, 6, 7
- CudaPropagator
 - `cudaProb3::CudaPropagator`, 10
- CudaPropagatorSingle
 - `cudaProb3::CudaPropagatorSingle`, 15, 16
- `cudaProb3::CpuPropagator`
 - `calculateProbabilities`, 7
 - `CpuPropagator`, 6, 7
 - `getProbability`, 7
 - `operator=`, 7, 8
- `cudaProb3::CpuPropagator< FLOAT_T >`, 5
- `cudaProb3::CudaPropagator`
 - `calculateProbabilities`, 11
 - `CudaPropagator`, 10
 - `getProbability`, 11
 - `operator=`, 11
 - `setCosineList`, 11
 - `setDensity`, 12
 - `setDensityFromFile`, 12
 - `setEnergyList`, 12
 - `setMNSMatrix`, 13
 - `setNeutrinoMasses`, 13
 - `setProductionHeight`, 13
- `cudaProb3::CudaPropagator< FLOAT_T >`, 8
- `cudaProb3::CudaPropagatorSingle`
 - `calculateProbabilities`, 16
 - `CudaPropagatorSingle`, 15, 16
 - `getProbability`, 16
 - `operator=`, 16
 - `setCosineList`, 16
 - `setDensity`, 17
 - `setEnergyList`, 17
- `cudaProb3::CudaPropagatorSingle< FLOAT_T >`, 14
- `cudaProb3::Propagator`
 - `calculateProbabilities`, 19
 - `getProbability`, 20
 - `operator=`, 20
 - `Propagator`, 19
 - `setCosineList`, 20
 - `setDensity`, 21
 - `setDensityFromFile`, 21
 - `setEnergyList`, 21
 - `setMNSMatrix`, 22
 - `setNeutrinoMasses`, 22
 - `setProductionHeight`, 22
- `setNeutrinoMasses`, 22
- `setProductionHeight`, 22
- `cudaProb3::Propagator< FLOAT_T >`, 17
- getProbability
 - `cudaProb3::CpuPropagator`, 7
 - `cudaProb3::CudaPropagator`, 11
 - `cudaProb3::CudaPropagatorSingle`, 16
 - `cudaProb3::Propagator`, 20
- operator=
 - `cudaProb3::CpuPropagator`, 7, 8
 - `cudaProb3::CudaPropagator`, 11
 - `cudaProb3::CudaPropagatorSingle`, 16
 - `cudaProb3::Propagator`, 20
- Propagator
 - `cudaProb3::Propagator`, 19
- setCosineList
 - `cudaProb3::CudaPropagator`, 11
 - `cudaProb3::CudaPropagatorSingle`, 16
 - `cudaProb3::Propagator`, 20
- setDensity
 - `cudaProb3::CudaPropagator`, 12
 - `cudaProb3::CudaPropagatorSingle`, 17
 - `cudaProb3::Propagator`, 21
- setDensityFromFile
 - `cudaProb3::CudaPropagator`, 12
 - `cudaProb3::Propagator`, 21
- setEnergyList
 - `cudaProb3::CudaPropagator`, 12
 - `cudaProb3::CudaPropagatorSingle`, 17
 - `cudaProb3::Propagator`, 21
- setMNSMatrix
 - `cudaProb3::CudaPropagator`, 13
 - `cudaProb3::Propagator`, 22
- setNeutrinoMasses
 - `cudaProb3::CudaPropagator`, 13
 - `cudaProb3::Propagator`, 22
- setProductionHeight
 - `cudaProb3::CudaPropagator`, 13
 - `cudaProb3::Propagator`, 22