NATriuM

Generated by Doxygen 1.7.6.1

Tue Sep 10 2013 20:01:34

Contents

Chapter 1

Class Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

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

Chapter 4

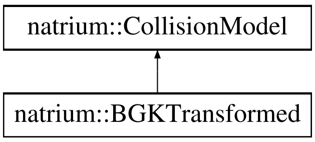
Class Documentation

4.1 natrium::BGKTransformed Class Reference

Description of the BGK model for the transformed particle distributions, as described in Global data which is used by Min and Lee (2011): A spectral-element discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259.

```
#include <BGKTransformed.h>
```

Inheritance diagram for natrium::BGKTransformed:



Public Member Functions

- BGKTransformed (float_t relaxationParameter, boost::shared_ptr< BoltzmannModel > boltzmannModel)

    constructor
- virtual ∼BGKTransformed ()

    destructor
- virtual void collide (vector< float_t > &distributions) const

    function for collision

4.1.1 Detailed Description

Description of the BGK model for the transformed particle distributions, as described in Global data which is used by Min and Lee (2011): A spectral-element discontinuous

Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 natrium::BGKTransformed::BGKTransformed (float_t *relaxationParameter,* boost::shared_ptr< BoltzmannModel > *boltzmannModel*)

constructor

Parameters

in	*relaxation-Parameter*	relaxation parameter tau

4.1.3 Member Function Documentation

4.1.3.1 void natrium::BGKTransformed::collide (vector< float_t > & *distributions*) const [virtual]

function for collision

Parameters

in/out]	distributions the particle distribution functions

Implements natrium::CollisionModel.

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

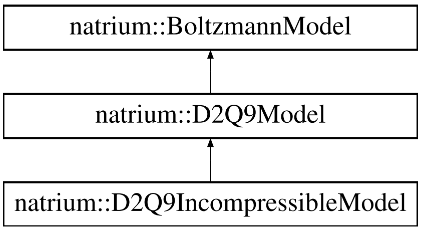
- /home/kraemer/eclipse_workspace/NATriuM/src/collisionmodels/BGKTransformed.-h
- /home/kraemer/eclipse_workspace/NATriuM/src/collisionmodels/BGKTransformed.-cpp

4.2 natrium::BoltzmannModel Class Reference

Abstract class for the description of a boltzmann model, e.g. D2Q9.

`#include <BoltzmannModel.h>`

Inheritance diagram for natrium::BoltzmannModel:



Public Member Functions

- BoltzmannModel (size_t d, size_t q, const vector< numeric_vector > &directions, const vector< float_t > &weights, StencilType stencilType)

  constructor
- virtual ∼BoltzmannModel ()

  destructor
- const vector< numeric_vector > & getDirections () const

  get a reference to the vector of directions
- const numeric_vector & getDirection (size_t i) const

  get the i-th direction
- size_t getQ () const

  get q, the number of directions in the DdQq-stencil
- size_t getD () const

  get d, the dimension of the DdQq-stencil
- const vector< float_t > & getWeights () const

  get the weights of the equilibrium distributions
- float_t getWeight (size_t i) const

  get the weight belonging to a certain direction
- const StencilType getStencilType () const

  get stencil type
- float_t calculateDensity (const vector< float_t > &distributions) const

  calculate macroscopic density
- numeric_vector calculateVelocity (const vector< float_t > &distributions) const

  calculate macroscopic velocity
- void calculateVelocity (const vector< float_t > &distributions, const float_t rho, numeric_vector &u) const

  calculate macroscopic velocity; saves the double calculation of the density
- virtual float_t getEquilibriumDistribution (size_t i, const numeric_vector &u, const float_t rho=1) const =0

  virtual function for the calculation of the equilibrium distribution
- virtual void getEquilibriumDistributions (vector< float_t > &feq, const numeric_vector &u, const float_t rho=1) const

  function for the calculation of all equilibrium distributions

4.2.1 Detailed Description

Abstract class for the description of a boltzmann model, e.g. D2Q9.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 natrium::BoltzmannModel::BoltzmannModel (size_t *d,* size_t *q,* const vector< numeric_vector > & *directions,* const vector< float_t > & *weights,* StencilType *stencilType*)

constructor

Parameters

d	dimension
q	number of directions
directions	the directions of the stencil
weights	the weights of the equilibrium distribution
stencilType	type of the stencil (e.g. D2Q9)

4.2.3 Member Function Documentation

4.2.3.1 float_t natrium::BoltzmannModel::calculateDensity (const vector< float_t > & *distributions*) const [inline]

calculate macroscopic density

Parameters

in	*distributions*	particle distribution functions at a given point

Returns

macroscopic density (sum of all distributions)

4.2.3.2 numeric_vector natrium::BoltzmannModel::calculateVelocity (const vector< float_t > & *distributions*) const [inline]

calculate macroscopic velocity

Parameters

in	*distributions*	particle distribution functions at a given point

Returns

macroscopic velocity

4.2.3.3 void **natrium::BoltzmannModel::calculateVelocity** (const vector< float_t > & *distributions,* const float_t *rho,* numeric_vector & *u*) const [inline]

calculate macroscopic velocity; saves the double calculation of the density

Note

more efficient

Parameters

in	*distributions*	particle distribution functions at a given point
in	*rho*	macroscopic density
out	*u*	macroscopic velocity

4.2.3.4 size_t **natrium::BoltzmannModel::getD** ( ) const [inline]

get d, the dimension of the DdQq-stencil

Returns

d

4.2.3.5 const numeric_vector& **natrium::BoltzmannModel::getDirection** (size_t *i*) const [inline]

get the i-th direction

Parameters

i	index i

Returns

a reference to the i-th direction of the DdQq-stencil

4.2.3.6 const vector<numeric_vector>& **natrium::BoltzmannModel::getDirections** ( ) const [inline]

get a reference to the vector of directions

Returns

a ublas_vector, which contains the directions of the DdQq-stencil as ublas_vectors

4.2.3.7 virtual float_t **natrium::BoltzmannModel::getEquilibriumDistribution** (size_t *i*, const numeric_vector & *u*, const float_t *rho* = `1`) const `[pure virtual]`

virtual function for the calculation of the equilibrium distribution

Parameters

i	index of the direction
u	macroscopic velocity
rho	macroscopic density

Returns

value of the equilibrium distribution

Note

The calculation can surely be done more efficiently by passing different arguments, e.g. u∗u or u/(c^2)

Implemented in natrium::D2Q9IncompressibleModel.

4.2.3.8 void **natrium::BoltzmannModel::getEquilibriumDistributions** (vector< float_t > & *feq*, const numeric_vector & *u*, const float_t *rho* = `1`) const `[virtual]`

function for the calculation of all equilibrium distributions

Parameters

`out`	*feq*	vector of all equality distributions, must have size Q
`in`	*u*	macroscopic velocity
`in`	*rho*	macroscopic density

Note

The calculation can surely be done more efficiently by passing different arguments, e.g. u∗u or u/(c^2)

4.2.3.9 size_t **natrium::BoltzmannModel::getQ** ( ) const `[inline]`

get q, the number of directions in the DdQq-stencil

Returns

q

4.2.3.10 const StencilType natrium::BoltzmannModel::getStencilType ( ) const [inline]

get stencil type

Returns

stencil type, e.g. D2Q9

4.2.3.11 float_t natrium::BoltzmannModel::getWeight (size_t *i*) const [inline]

get the weight belonging to a certain direction

Parameters

i	index i of the direction (1 <= i <= q)

Returns

the i-th weight

4.2.3.12 const vector<float_t>& natrium::BoltzmannModel::getWeights ( ) const [inline]

get the weights of the equilibrium distributions

Returns

a reference to the vector of weights

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

- /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/BoltzmannModel.h
- /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/BoltzmannModel.cpp

4.3 natrium::CFDSolver< dim > Class Template Reference

The central class for the CFD simulation based on the DBE.

```
#include <CFDSolver.h>
```

Public Member Functions

- CFDSolver ()
- virtual ∼CFDSolver ()

    destructor

4.3.1 Detailed Description

template<int dim>class natrium::CFDSolver< dim >

The central class for the CFD simulation based on the DBE.

Note

> The CFDSolver itself is quite static but it contains interchangeable modules, e.g. for the Boltzmann model or the time integrator. By these means, a variety of different simulation methods can be covered.

Template Parameters

dim	The dimension of the flow (2 or 3).

4.3.2 Constructor & Destructor Documentation

4.3.2.1 template<int dim> natrium::CFDSolver< dim >::CFDSolver ( ) `[inline]`

constructor

Note

> : has to be inlined, if the template parameter is not made explicit

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

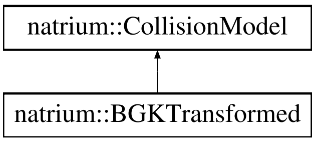
- /home/kraemer/eclipse_workspace/NATriuM/src/solver/CFDSolver.h

4.4 natrium::CollisionModel Class Reference

Abstract class for the description of collision schemes.

`#include <CollisionModel.h>`

Inheritance diagram for natrium::CollisionModel:



Public Member Functions

- CollisionModel (float_t relaxationParameter, boost::shared_ptr< BoltzmannModel > boltzmannModel)

  constructor
- virtual ∼CollisionModel ()

  destructor
- virtual void collide (vector< float_t > &distributions) const =0

  virtual function for collision
- float_t getRelaxationParameter () const

  get relaxation parameter

Protected Attributes

- float_t m_relaxationParameter

  relaxation parameter
- const boost::shared_ptr < BoltzmannModel > m_boltzmannModel

  Boltzmann model (e.g. D2Q9Incompressible)
- size_t m_d

  D (dimension)
- float_t m_q

  Q (number of directions)

4.4.1 Detailed Description

Abstract class for the description of collision schemes.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 natrium::CollisionModel::CollisionModel (float_t *relaxationParameter,* boost::shared_ptr< BoltzmannModel > *boltzmannModel*)

constructor

Parameters

in	*relaxationParameter*	relaxation parameter tau

4.4.3 Member Function Documentation

4.4.3.1 virtual void **natrium::CollisionModel::collide** (vector< float_t > & *distributions*) const [pure virtual]

virtual function for collision

Parameters

in/out]	distributions the particle distribution functions

Implemented in natrium::BGKTransformed.

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

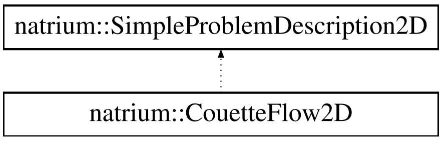
- /home/kraemer/eclipse_workspace/NATriuM/src/collisionmodels/CollisionModel.h
- /home/kraemer/eclipse_workspace/NATriuM/src/collisionmodels/CollisionModel.cpp

4.5 natrium::CouetteFlow2D Class Reference

Description of a simple Couette Flow (regular channel flow in rectangular domain).

`#include <CouetteFlow2D.h>`

Inheritance diagram for natrium::CouetteFlow2D:



Public Member Functions

- CouetteFlow2D ()

  constructor
- virtual ∼CouetteFlow2D ()

  destructor

4.5.1 Detailed Description

Description of a simple Couette Flow (regular channel flow in rectangular domain).

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

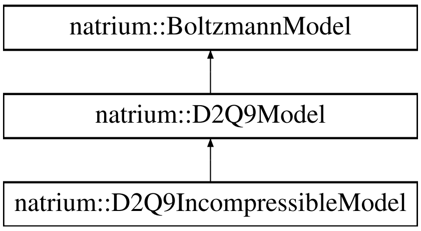
- /home/kraemer/eclipse_workspace/NATriuM/src/benchmarks/CouetteFlow2D.h
- /home/kraemer/eclipse_workspace/NATriuM/src/benchmarks/CouetteFlow2D.cpp

4.6 natrium::D2Q9IncompressibleModel Class Reference

D2Q9 model description for incompressible flow.

```
#include <D2Q9IncompressibleModel.h>
```

Inheritance diagram for natrium::D2Q9IncompressibleModel:



Public Member Functions

- D2Q9IncompressibleModel ()

  constructor
- virtual ∼D2Q9IncompressibleModel ()

  destructor
- virtual float_t getEquilibriumDistribution (size_t i, const numeric_vector &u, const float_t rho=1) const

  function for the calculation of the equilibrium distribution in the incompressible D2Q9 model

4.6.1 Detailed Description

D2Q9 model description for incompressible flow.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 natrium::D2Q9IncompressibleModel::∼D2Q9IncompressibleModel ( ) [virtual]

destructor

constructor

4.6.3 Member Function Documentation

4.6.3.1 float_t natrium::D2Q9IncompressibleModel::getEquilibriumDistribution (size_t *i*, const numeric_vector & *u*, const float_t *rho* = `1`) const `[virtual]`

function for the calculation of the equilibrium distribution in the incompressible D2Q9 model

destructor

Parameters

i	index of the direction
u	macroscopic velocity
rho	macroscopic density

Returns

value of the equilibrium distribution

Note

The calculation can surely be done more efficiently by passing different arguments, e.g. u∗u or u/(c^2)

getEquilibriumDistribution

Implements natrium::BoltzmannModel.

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

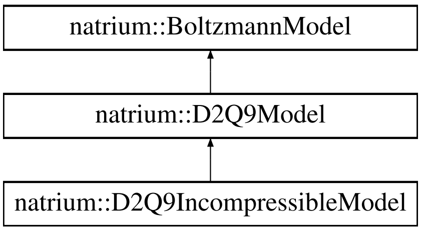
- /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/D2Q9IncompressibleModel.h
- /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/D2Q9IncompressibleModel.cpp

4.7 natrium::D2Q9Model Class Reference

D2Q9 Model.

`#include <D2Q9Model.h>`

Inheritance diagram for natrium::D2Q9Model:



Public Member Functions

- D2Q9Model ()

  constructor
- virtual ∼D2Q9Model ()

  destructor

Static Public Attributes

- static const size_t D = 2

  D.
- static const size_t Q = 9

  Q.
- static const float_t speedOfSound = pow(3, -0.5)

  speed of sound
- static const float_t speedOfSoundSquare = 1. / 3.

  (speed of sound)^2

4.7.1 Detailed Description

D2Q9 Model.

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

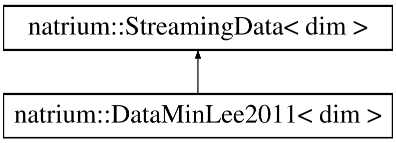
- /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/D2Q9Model.h
- /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/D2Q9Model.cpp

4.8 natrium::DataMinLee2011< dim > Class Template Reference

Global data which is used, e.g., by Min and Lee (2011): A spectral-element discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259. including particle distributions f, system matrix L, diagonal mass matrix M, gradient matrices Dx, Dy, (Dz) and boundary matrix R.

```
#include <DataMinLee2011.h>
```

Inheritance diagram for natrium::DataMinLee2011< dim >:



Public Member Functions

- DataMinLee2011 ()

  constructor
- virtual ∼DataMinLee2011 ()

  destructor

4.8.1 Detailed Description

template<int dim>class natrium::DataMinLee2011< dim >

Global data which is used, e.g., by Min and Lee (2011): A spectral-element discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259. including particle distributions f, system matrix L, diagonal mass matrix M, gradient matrices Dx, Dy, (Dz) and boundary matrix R.

Template Parameters

dim	The dimension of the flow (2 or 3).

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

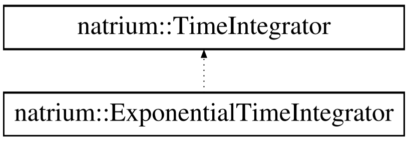
- /home/kraemer/eclipse_workspace/NATriuM/src/streamingdata/DataMinLee2011.h

4.9 natrium::ExponentialTimeIntegrator Class Reference

Exponential time integration scheme for the solution of f' = L∗f, as used in Uga etal. (2012) Spectral-element discontinuous Galerkin lattice Boltzmann simulation of flow past two cylinders in tandem with an exponential time integrator, CMWA 65 pp. 239-251.

```
#include <ExponentialTimeIntegrator.h>
```

Inheritance diagram for natrium::ExponentialTimeIntegrator:



Public Member Functions

- ExponentialTimeIntegrator ()

  constructor
- virtual ∼ExponentialTimeIntegrator ()

destructor

4.9.1 Detailed Description

Exponential time integration scheme for the solution of f' = L∗f, as used in Uga etal. (2012) Spectral-element discontinuous Galerkin lattice Boltzmann simulation of flow past two cylinders in tandem with an exponential time integrator, CMWA 65 pp. 239-251.

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

- /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/ExponentialTimeIntegrator.h
- /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/ExponentialTimeIntegrator.cpp

4.10 natrium::Math Class Reference

class which contains basic math functions

```
#include <BasicNames.h>
```

Static Public Member Functions

- static float_t scalar_product (const numeric_vector &x, const numeric_vector &y)

  scalar product
- static void scale_vector (float_t a, numeric_vector &x)

  scale existing vector
- static numeric_vector scalar_vector (float_t a, const numeric_vector &x)

  scalar times vector
- static void **add_vector** (numeric_vector &x, const numeric_vector &y)
- static float_t **euclidean_norm** (numeric_vector &x)
- static float_t **by_two** (float_t a)

4.10.1 Detailed Description

class which contains basic math functions

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

- /home/kraemer/eclipse_workspace/NATriuM/src/utilities/BasicNames.h

4.11 natrium::ProblemDescription< dim > Class Template - Reference

Abstract class for the description of a CFD problem. The description includes the computational mesh, boundary description, viscosity and initial values.

```
#include <ProblemDescription.h>
```

Public Member Functions

- ProblemDescription ()

  constructor
- virtual ∼ProblemDescription ()

  destructor

4.11.1 Detailed Description

template<int dim>class natrium::ProblemDescription< dim >

Abstract class for the description of a CFD problem. The description includes the computational mesh, boundary description, viscosity and initial values.

Template Parameters

dim	The dimension of the flow (2 or 3).

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

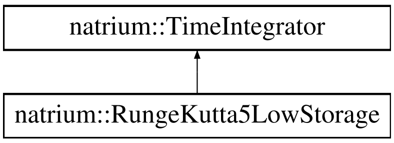
- /home/kraemer/eclipse_workspace/NATriuM/src/problemdescription/ProblemDescription.h

4.12 natrium::RungeKutta5LowStorage Class Reference

Implementation of the fifth-order Runge-Kutta time integration scheme with low storage consumption.

```
#include <RungeKutta5LowStorage.h>
```

Inheritance diagram for natrium::RungeKutta5LowStorage:



Public Member Functions

- RungeKutta5LowStorage ()

  constructor
- virtual ∼RungeKutta5LowStorage ()

  destructor

4.12.1 Detailed Description

Implementation of the fifth-order Runge-Kutta time integration scheme with low storage consumption.

Note

> The scheme is described in Min and Lee (2011): A spectral-element discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259.

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

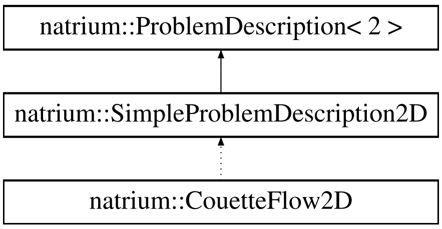
- /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/RungeKutta5LowStorage.h
- /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/RungeKutta5LowStorage.cpp

4.13 natrium::SimpleProblemDescription2D Class Reference

Description of simple 2D test problems, using boundary IDs and easy-to-use boundary functions.

```
#include <SimpleProblemDescription2D.h>
```

Inheritance diagram for natrium::SimpleProblemDescription2D:


natrium::ProblemDescription< 2 >
natrium::SimpleProblemDescription2D
natrium::CouetteFlow2D

Public Member Functions

- SimpleProblemDescription2D ()

constructor

- virtual ∼SimpleProblemDescription2D ()

  destructor

4.13.1 Detailed Description

Description of simple 2D test problems, using boundary IDs and easy-to-use boundary functions.

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

- /home/kraemer/eclipse_workspace/NATriuM/src/problemdescription/SimpleProblemDescription2D.h
- /home/kraemer/eclipse_workspace/NATriuM/src/problemdescription/SimpleProblemDescription2D.cpp

4.14 natrium::SolverConfiguration Class Reference

Class that stores the configuration for a CFD simulation based on the Discrete - Boltzmann Equation (DBE).

```
#include <SolverConfiguration.h>
```

Public Member Functions

- SolverConfiguration ()

  constructor

- virtual ∼SolverConfiguration ()

  destructor

4.14.1 Detailed Description

Class that stores the configuration for a CFD simulation based on the Discrete - Boltzmann Equation (DBE).

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

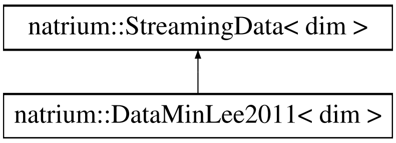
- /home/kraemer/eclipse_workspace/NATriuM/src/solver/SolverConfiguration.h
- /home/kraemer/eclipse_workspace/NATriuM/src/solver/SolverConfiguration.cpp

4.15 natrium::StreamingData< dim > Class Template Reference

Abstract class to store global streaming data, like e.g. the particle distributions.

`#include <StreamingData.h>`

Inheritance diagram for natrium::StreamingData< dim >:



Public Member Functions

- StreamingData ()

  constructor
- virtual ∼StreamingData ()

  destructor

4.15.1 Detailed Description

template<int dim>class natrium::StreamingData< dim >

Abstract class to store global streaming data, like e.g. the particle distributions.

Note

> The data to store differs for different approaches of solving the DBE. The only common data which will appear in every method is the particle distribution functions and some global system matrix.

Template Parameters

dim	The dimension of the flow (2 or 3).

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

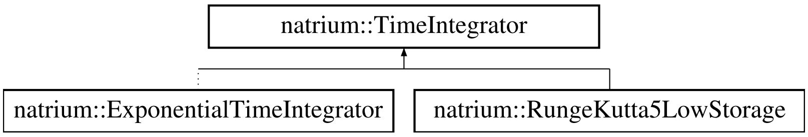
- /home/kraemer/eclipse_workspace/NATriuM/src/streamingdata/StreamingData.h

4.16 natrium::TimeIntegrator Class Reference

Abstract class for time integration (solution of ordinary differential equations (ODE)).

`#include <TimeIntegrator.h>`

Inheritance diagram for natrium::TimeIntegrator:


natrium::TimeIntegrator
natrium::ExponentialTimeIntegrator
natrium::RungeKutta5LowStorage

Public Member Functions

- TimeIntegrator ()

  constructor
- virtual ∼TimeIntegrator ()

  destructor

4.16.1 Detailed Description

Abstract class for time integration (solution of ordinary differential equations (ODE)).

Note

> The ODEs arise from the space discretization of the DBE, which is basically a partial differential equation (PDE). By application of a space discretization scheme (like discontinuous Galerkin or standard FEM methods) the space derivatives in the PDE are replaced with arithmetic expressions. The only remaining derivative is then the time derivative, which makes the equation an ODE. The latter can be solved using classical time integration methods like Runge-Kutta or Adams-Moulton.

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

- /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/TimeIntegrator.h
- /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/TimeIntegrator.cpp

Chapter 5

File Documentation

5.1 /home/kraemer/eclipse_workspace/NATriuM/src/benchmarks/CouetteFlow2D.cpp File Reference

```
#include "CouetteFlow2D.h"
```

5.1.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.2 /home/kraemer/eclipse_workspace/NATriuM/src/benchmarks/CouetteFlow2D.h File Reference

Description of a simple Couette Flow (regular channel flow in rectangular domain).

```
#include "../problemdescription/SimpleProblemDescription2D.h"
```

Classes

- class natrium::CouetteFlow2D

  Description of a simple Couette Flow (regular channel flow in rectangular domain).

5.2.1 Detailed Description

Description of a simple Couette Flow (regular channel flow in rectangular domain).

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.3 /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/-BoltzmannModel.cpp File Reference

Abstract class for the description of a boltzmann model.

```
#include "BoltzmannModel.h"
```

5.3.1 Detailed Description

Abstract class for the description of a boltzmann model.

Date

04.06.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.4 /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/-BoltzmannModel.h File Reference

Abstract class for the description of a boltzmann model.

```
#include "assert.h" #include "../utilities/BasicNames.h"
```

Classes

- class natrium::BoltzmannModel

  Abstract class for the description of a boltzmann model, e.g. D2Q9.

Enumerations

- enum **StencilType** { **natrium::D2Q9** }

  Enum type for the difference stencil.

5.4.1 Detailed Description

Abstract class for the description of a boltzmann model.

Date

04.06.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.5 /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/-D2Q9Model.cpp File Reference

```
#include "D2Q9Model.h" #include <math.h>
```

5.5.1 Detailed Description

Date

30.08.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.6 /home/kraemer/eclipse_workspace/NATriuM/src/boltzmannmodels/-D2Q9Model.h File Reference

D2Q9 Boltzmann Model.

```
#include "BoltzmannModel.h" #include "../utilities/Basic-
Names.h" #include "boost/assign/std/vector.hpp"
```

Classes

- class natrium::D2Q9Model

  D2Q9 Model.

5.6.1 Detailed Description

D2Q9 Boltzmann Model.

Date

02.06.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.7 /home/kraemer/eclipse_workspace/NATriuM/src/collisionmodels/-BGKTransformed.cpp File Reference

```
#include "BGKTransformed.h"
```

5.7.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.8 /home/kraemer/eclipse_workspace/NATriuM/src/collisionmodels/-BGKTransformed.h File Reference

Description of the BGK model for the transformed particle distributions, as described in Global data which is used by Min and Lee (2011): A spectral-elemennt discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259.

```
#include "CollisionModel.h" #include "../utilities/Basic-
Names.h"
```

Classes

- class natrium::BGKTransformed

  Description of the BGK model for the transformed particle distributions, as described in Global data which is used by Min and Lee (2011): A spectral-element discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259.

5.8.1 Detailed Description

Description of the BGK model for the transformed particle distributions, as described in Global data which is used by Min and Lee (2011): A spectral-elemennt discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259.

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.9 /home/kraemer/eclipse_workspace/NATriuM/src/collisionmodels/CollisionModel.cpp File Reference

```
#include "CollisionModel.h"
```

5.9.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.10 /home/kraemer/eclipse_workspace/NATriuM/src/collisionmodels/CollisionModel.h File Reference

Abstract class for the description of collision schemes.

```
#include "boost/shared_ptr.hpp" #include "../boltzmannmodels/BoltzmannModel.h" #include "../utilities/BasicNames.h"
```

Classes

- class natrium::CollisionModel

  Abstract class for the description of collision schemes.

5.10.1 Detailed Description

Abstract class for the description of collision schemes.

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.11 /home/kraemer/eclipse_workspace/NATriuM/src/problemdescription/-ProblemDescription.cpp File Reference

```
#include "ProblemDescription.h"
```

5.11.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.12 /home/kraemer/eclipse_workspace/NATriuM/src/problemdescription/-ProblemDescription.h File Reference

Abstract class for the description of a CFD problem. The description includes the computational mesh, boundary description, viscosity and initial values.

Classes

- class natrium::ProblemDescription< dim >

  Abstract class for the description of a CFD problem. The description includes the computational mesh, boundary description, viscosity and initial values.

5.12.1 Detailed Description

Abstract class for the description of a CFD problem. The description includes the computational mesh, boundary description, viscosity and initial values.

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.13 /home/kraemer/eclipse_workspace/NATriuM/src/problemdescription/SimpleProblemDescription2D.cpp File Reference

```
#include "SimpleProblemDescription2D.h"
```

5.13.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.14 /home/kraemer/eclipse_workspace/NATriuM/src/problemdescription/SimpleProblemDescription2D.h File Reference

Description of simple 2D test problems, using boundary IDs and easy-to-use boundary functions.

```
#include "ProblemDescription.h"
```

Classes

- class natrium::SimpleProblemDescription2D

  Description of simple 2D test problems, using boundary IDs and easy-to-use boundary functions.

5.14.1 Detailed Description

Description of simple 2D test problems, using boundary IDs and easy-to-use boundary functions.

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.15 /home/kraemer/eclipse_workspace/NATriuM/src/solver/CFDSolver.cpp File Reference

```
#include "CFDSolver.h"
```

5.15.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.16 /home/kraemer/eclipse_workspace/NATriuM/src/solver/CFDSolver.h File Reference

Central class of the CFD Simulation based on the Discrete Boltzmann Equation (DBE)

```
#include "boost/shared_ptr.hpp" #include "../streamingdata/StreamingData.h" #include "../problemdescription/ProblemDescription.h" #include "../boltzmannmodels/BoltzmannModel.h" #include "../collisionmodels/CollisionModel.h" × #include "../timeintegration/TimeIntegrator.h"
```

Classes

- class natrium::CFDSolver< dim >

  The central class for the CFD simulation based on the DBE.

5.16.1 Detailed Description

Central class of the CFD Simulation based on the Discrete Boltzmann Equation (DBE)

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.17 /home/kraemer/eclipse_workspace/NATriuM/src/solver/SolverConfiguration.cpp File Reference

```
#include "SolverConfiguration.h"
```

5.17.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.18 /home/kraemer/eclipse_workspace/NATriuM/src/solver/SolverConfiguration.h File Reference

Class that stores the configuration for a CFD simulation based on the Discrete - Boltzmann Equation (DBE).

Classes

- class natrium::SolverConfiguration

  Class that stores the configuration for a CFD simulation based on the Discrete - Boltzmann Equation (DBE).

5.18.1 Detailed Description

Class that stores the configuration for a CFD simulation based on the Discrete - Boltzmann Equation (DBE).

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.19 /home/kraemer/eclipse_workspace/NATriuM/src/streamingdata/-DataMinLee2011.cpp File Reference

```
#include "DataMinLee2011.h"
```

5.19.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.20 /home/kraemer/eclipse_workspace/NATriuM/src/streamingdata/-DataMinLee2011.h File Reference

Global data which is used by Min and Lee (2011): A spectral-elemennt discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259.

```
#include "StreamingData.h"
```

Classes

- class natrium::DataMinLee2011< dim >

  Global data which is used, e.g., by Min and Lee (2011): A spectral-element discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259. including particle distributions f, system matrix L, diagonal mass matrix M, gradient matrices Dx, Dy, (Dz) and boundary matrix R.

5.20.1 Detailed Description

Global data which is used by Min and Lee (2011): A spectral-elemennt discontinuous Galerkin lattice Boltzmann method for nearly incompressible flows, JCP 230 pp. 245-259.

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.21 /home/kraemer/eclipse_workspace/NATriuM/src/streamingdata/StreamingData.cpp File Reference

```
#include "StreamingData.h"
```

5.21.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.22 /home/kraemer/eclipse_workspace/NATriuM/src/streamingdata/StreamingData.h File Reference

Abstract class to store global streaming data like the particle distributions and assemble the matrices.

Classes

- class natrium::StreamingData$<$ dim $>$

  Abstract class to store global streaming data, like e.g. the particle distributions.

5.22.1 Detailed Description

Abstract class to store global streaming data like the particle distributions and assemble the matrices.

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.23 /home/kraemer/eclipse_workspace/NATriuM/src/test/benchmarks/CouetteFlow2D_test.cpp File Reference

```
#include "../../benchmarks/CouetteFlow2D.h"
```

5.23.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.24 /home/kraemer/eclipse_workspace/NATriuM/src/test/collisionmodels/BGKTransformed_test.cpp File Reference

```
#include "../../collisionmodels/BGKTransformed.h" #include
"boost/test/unit_test.hpp"    #include "boost/shared_ptr.-
hpp" #include "boost/make_shared.hpp" #include "../../boltzmannmodels/-
D2Q9IncompressibleModel.h" #include "../../boltzmannmodels/-
BoltzmannModel.h"   #include "../../utilities/BasicNames.-
h"
```

Functions

- **natrium::BOOST_AUTO_TEST_CASE** (BGKTransformedConstruction_test)
- **natrium::BOOST_AUTO_TEST_CASE** (BGKTransformedGetter_test)
- **natrium::BOOST_AUTO_TEST_CASE** (BGKTransformedCollisionInvariants_test)

5.24.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.25 /home/kraemer/eclipse_workspace/NATriuM/src/test/collisionmodels/-CollisionModel_test.cpp File Reference

```
#include "../../collisionmodels/CollisionModel.h"
```

5.25.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.26 /home/kraemer/eclipse_workspace/NATriuM/src/test/MainTest.cpp File Reference

```
#include "boost/test/unit_test.hpp"
```

Defines

- #define **BOOST_TEST_DYN_LINK**
- #define **BOOST_TEST_MODULE** Main

Functions

- **BOOST_AUTO_TEST_CASE** (Boost_test)

5.26.1 Detailed Description

Date

04.09.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.27 /home/kraemer/eclipse_workspace/NATriuM/src/test/problemdescription/-ProblemDescription_test.cpp File Reference

```
#include "../../problemdescription/ProblemDescription.-
h"
```

5.27.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.28 /home/kraemer/eclipse_workspace/NATriuM/src/test/problemdescription/-SimpleProblemDescription2D_test.cpp File Reference

```
#include "../../problemdescription/SimpleProblemDescription2-
D.h"
```

5.28.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.29 /home/kraemer/eclipse_workspace/NATriuM/src/test/solver/C-FDSolver_test.cpp File Reference

```
#include "../../solver/CFDSolver.h"
```

5.29.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.30 /home/kraemer/eclipse_workspace/NATriuM/src/test/solver/-SolverConfiguration_test.cpp File Reference

```
#include "../../solver/SolverConfiguration.h"
```

5.30.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.31 /home/kraemer/eclipse_workspace/NATriuM/src/test/streamingdata/DataMinLee2011_test.cpp File Reference

```
#include "../../streamingdata/DataMinLee2011.h"
```

5.31.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.32 /home/kraemer/eclipse_workspace/NATriuM/src/test/streamingdata/StreamingData_test.cpp File Reference

```
#include "../../streamingdata/StreamingData.h"
```

5.32.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.33 /home/kraemer/eclipse_workspace/NATriuM/src/test/timeintegration/ExponentialTimeIntegrator_test.cpp File Reference

```
#include "../../timeintegration/ExponentialTimeIntegrator.-
h"
```

5.33.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.34 /home/kraemer/eclipse_workspace/NATriuM/src/test/timeintegration/-RungeKutta5LowStorage_test.cpp File Reference

```
#include "../../timeintegration/RungeKutta5LowStorage.-
h"
```

5.34.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.35 /home/kraemer/eclipse_workspace/NATriuM/src/test/timeintegration/-TimeIntegrator_test.cpp File Reference

```
#include "../../timeintegration/TimeIntegrator.h"
```

5.35.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.36 /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/-ExponentialTimeIntegrator.cpp File Reference

```
#include "ExponentialTimeIntegrator.h"
```

5.36.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.37 /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/-ExponentialTimeIntegrator.h File Reference

Exponential time integration scheme for the solution of f' = L∗f.

```
#include "TimeIntegrator.h"
```

Classes

- class natrium::ExponentialTimeIntegrator

  Exponential time integration scheme for the solution of f' = L∗f, as used in Uga etal. (2012) Spectral-element discontinuous Galerkin lattice Boltzmann simulation of flow past two cylinders in tandem with an exponential time integrator, CMWA 65 pp. 239-251.

5.37.1 Detailed Description

Exponential time integration scheme for the solution of f' = L∗f.

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.38 /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/-RungeKutta5LowStorage.cpp File Reference

```
#include "RungeKutta5LowStorage.h"
```

5.38.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.39 /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/-RungeKutta5LowStorage.h File Reference

Fifth-order Runge-Kutta time integration scheme with low storage consumption.

```
#include "TimeIntegrator.h"
```

Classes

- class natrium::RungeKutta5LowStorage

  Implementation of the fifth-order Runge-Kutta time integration scheme with low storage consumption.

5.39.1 Detailed Description

Fifth-order Runge-Kutta time integration scheme with low storage consumption.

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.40 /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/-TimeIntegrator.cpp File Reference

```
#include "TimeIntegrator.h"
```

5.40.1 Detailed Description

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.41 /home/kraemer/eclipse_workspace/NATriuM/src/timeintegration/TimeIntegrator.h File Reference

Abstract class for time integrationof ordinary differential equations (ODEs).

Classes

- class natrium::TimeIntegrator

  Abstract class for time integration (solution of ordinary differential equations (ODE)).

5.41.1 Detailed Description

Abstract class for time integrationof ordinary differential equations (ODEs).

Date

29.05.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin

5.42 /home/kraemer/eclipse_workspace/NATriuM/src/utilities/BasicNames.h File Reference

`#include "deal.II/numerics/vector_tools.h"` `#include "deal.II/lac/petsc_vector.h"` `#include "deal.II/lac/petsc_parallel_vector.h"`

Classes

- class natrium::Math

  class which contains basic math functions

Typedefs

- typedef unsigned int **natrium::size_t**

  size type
- typedef double **natrium::float_t**

  floating point number
- typedef dealii::Vector< float_t > **natrium::numeric_vector**

  vector for numeric operations
- typedef numeric_vector **natrium::distributed_vector**

  vector which can be distributed over different cores
- typedef dealii::FullMatrix < float_t > **natrium::numeric_matrix**

  matrix for numeric operations

5.42.1 Detailed Description

Date

30.08.2013

Author

Andreas Kraemer, Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin