# PSpectRE

Generated by Doxygen 1.6.2

Mon Sep 13 09:37:28 2010

# Contents

# Chapter 1

# SpectRE - A Spectral Code for Reheating

SpectRE is a pseudo-spectral code for simulating a pair of interacting scalar fields in a self-consistently expanding background. These fields are named phi and chi.

The time-dependent variable-rescaling scheme from LatticeEasy is used to eliminate the first order term from the equations of motion. The fields can be initialized using either the scheme from LatticeEasy or the scheme from Defrost.



## 1.1 References


- Gary Felder and Igor Tkachev. LATTICEEASY: A Program for Lattice Simulations of Scalar Fields in an Expanding Universe. arXiv:hep-ph/0011159v1. http://www.science.smith.edu/departments/Physics/fstaff/gfelder/latticeeasy/
- Andrei V. Frolov. DEFROST: A New Code for Simulating Preheating after Inflation. arXiv:0809.4904v2 [hep-ph]. http://www.sfu.ca/physics/cosmology/defrost/

# Chapter 2

# energy.tsv

energy.tsv is a tab serarated file with the following fields:

- Program time
- Physical time
- Average physical energy (w.r.t. the rescaled length)
- Average energy normalized by the Friedmann equation
- Average normalized $\phi'^2$ energy contribution
- Average normalized $\chi'^2$ energy contribution
- Average normalized $\phi\phi'$ energy contribution
- Average normalized $\chi\chi'$ energy contribution
- Average normalized $\phi^2$ energy contribution
- Average normalized $\chi^2$ energy contribution
- Average normalized $\nabla\phi$ energy contribution
- Average normalized $\nabla\chi$ energy contribution
- Average normalized potential-energy contribution
- Average physical $\phi'^2$ energy contribution
- Average physical $\chi'^2$ energy contribution
- Average physical $\phi\phi'$ energy contribution
- Average physical $\chi\chi'$ energy contribution
- Average physical $\phi^2$ energy contribution
- Average physical $\chi^2$ energy contribution
- Average physical $\nabla\phi$ energy contribution
- Average physical $\nabla\chi$ energy contribution
- Average physical potential-energy contribution

# Chapter 3

# Running

## 3.1 Parameters

SpectRE Usage:

```
./pspectre [-h]
./pspectre [-r] [-l [-B <real>]] [-V] [-H <name>[,<name>]*] [-O] [-N <int>] [-P
     <int>] [-L <real>] [-R <int>] [-o <dir name>] [-t <real>[:<real>]] [-T <real>] [-
     A <real>] [-p <name>=<value>[,<name>=<value>]*] [-s <name>[,<name>]*] [-S <name>[
     =<value>][,<name>[=<value>]]*] [-I <name>=<value>[,<name>=<value>]*] [--long] [@<
     file name>]
```

- -h: Display usage information and exit
- -r: Use the RK4 integrator (default is the Verlet integrator)
- -l: Use LatticeEasy-style initial conditions (default is DEFROST-style initial conditions)
- -B: The base length scale (default is 1.0 to match LatticeEasy)
- -V: Allow the field variance to change with L
- -H <name>[,<name>]*: Use homogeneous (zero variance) initial conditions. Field names are:

  ```
  phi
  chi
  ```

- -O: Use out-of-place transforms
- -N <int>: The number of grid points per side of the box
- -P <int>: The padding factor used for position-space integration
- -L <real>: The physical size of the box
- -R <int>: The random seed
- -o <dir name>: Set the output directory name
- -t <real>[:<real>]: Set dt with an optional start time in program units
- -T <real>: The final time in program units
- -A <real>: The final scale factor
- -p <name>=<value>[,<name>=<value>]*: Set a parameter value. Valid parameters are:

```
gamma_phi
gamma_chi
lambda_phi
lambda_chi
g
m_phi
m_chi
phi0
chi0
phidot0
chidot0
ics_scale
```

- -s <name>[,<name>]∗: Enable slice output of a variable. Valid variables are:

```
phi
chi
V
T_phi
T_chi
G_phi
G_chi
rho
p
gpot
```

- -S <name>[=<value>][,<name>[=<value>]]∗: Set a slice output option value. Valid options are:

```
dim
length
skip
avg
(avg does not take a value)
```

- -I <name>=<value>[,<name>=<value>]∗: Set an output interval. Valid intervals are:

```
scale
energy
spectra
twoptcorr
screen
slice
stats
all
(intervals are specified as a number of iterations)
```

- --long: Run using long-double (extended) precision (this must be the ∗last∗ command-line option argument)

- @<file name>: The name of a parameters file. The parameters file has the same syntax as the command line except that it may be divided among multiple lines and may contain comment lines which begin with a # character.

The default parameters model a situation generally similar to the default model provided with DEFROST version 1.1.

## 3.2 Examples

The following runs the model with the default parameters except that it sets a 128^3 grid with dt = 0.0005. Also, -r selects the RK4 integrator (Verlet is default). -l selects LE-style initial conditions. -I all=1 sets all output intervals to 1 time step (the default is 25).

```
./pspectre -N 128 -t 0.0005 -r -l -I all=1
```

The following runs the model with the default parameters and has binary slice outputs for the energy density, pressure and gravitational potential. The slices to have a length of 32 points per side and were constructed by averaging (not skipping) over every eight-point cube (since the dimension is 3). -P 2 causes the integration over the potential energy to use a (2N)^3 grid.

```
./pspectre -P 2 -s rho,p,gpot -S dim=3,length=32,skip=1,avg
```

# Chapter 4

# Output Files

All output files generated by SpectRE are placed into a directory named output-YYYYMMDDHHMMSS where YYYY is the current year, etc.

- info.txt
- sf.tsv
- energy.tsv
- stats.tsv
- spectra.tsv
- twoptcorr.tsv
- Binary Slices

# Chapter 5

# info.txt

The info.txt contains a human-readable summary of the run parameters (both physical and numerical).

# Chapter 6

# sf.tsv

sf.tsv is a tab serarated file with the following fields:

- Program time
- Physical time
- a
- H

# Chapter 7

# Building

## 7.1 Make

Building SpectRE requires GNU make. On systems where GNU make is not the system's default make command, GNU make is often called gmake.

## 7.2 Requirements

SpectRE should build and run on any POSIX-style operating system, and uses OpenMP for shared-memory parallelism. It requires:

- FFTW 3 or Intel's MKL version 10+.
- G++ (the GNU C++ compiler version 4+) or ICC (the Intel C++ compiler).

## 7.3 Targets

The following (phony) targets are defined:

- rel - Build the release (optimized) spectre executable. This is the default target.
- profile - Build the optimized profiling executable spectre-pg.
- debug - Build the debug spectre-dbg executable.
- debug-mudflap - Build the mudflap-enabled debug executable spectre-dbg-mf.
- doc - Build the documentation (doxygen and dot required).
- clean - Remove all generated files (including executables) except for the documentation.
- clean-doc - Remove the documentation files.
- clean-all - A combination of clean and clean-doc.
- dist - A combination of clean-all and doc followed by the creation of a source archive.

## 7.4 Variables

The make file recognizes the following variables which can be specified on the command line prior to or after the target name(s):

- USE_ICC=yes - Use the Intel C++ compiler instead of the GNU C++ compiler.

- USE_MKL=yes - Use the Intel Math Kernel Libraries intead of FFTW. The MKL FFTW wrapper library is used, which is provided in source form with the MKL installation, and so the MKLROOT environmental variable must be set appropriately.
- USE_LD=yes - Enable long-double support (not supported when using MKL). If the fftwl-wisdom utility exists in a directory in the current search path, then long double support is active by default.

## 7.5 Examples

To build spectre using g++ and FFTW:

```
make
```

To build spectre using icc and the MKL:

```
make USE_ICC=yes USE_MKL=yes
```

To build spectre-dbg using icc and FFTW:

```
make USE_ICC=yes debug
```

## 7.6 Compiler Selection

The name of the compiler used can be overridden by setting the GXX variable. By default, this variable has the value g++ or icc. If an executable called g++-4 is found in the current search path, then it is used in preference to g++.

# Chapter 8

# Binary Slices

Binary slices are optionally generated for many different variables.

Single-precision floating point format is used regardless of the precision used for computation. The "length" parameter indicates the length of the side of the grid from which the slice is taken, **not** the size of the output slice if "skip" is $> 0$. "skip" is the number of grid points inbetween output points. If averaging is active, the skipped points are averaged over instead of actually being skipped.

# Chapter 9

# spectra.tsv

spectra.tsv is a tab serarated file with the following fields:

- Program time
- Physical time
- Bin number
- Grid points per bin
- Physical bin momentum
- Total program-unit phi momentum
- Total program-unit chi momentum
- Total physical phi momentum
- Total physical chi momentum

# Chapter 10

# stats.tsv

stats.tsv is a tab serarated file with the following fields:

- Program time
- Physical time
- Mean of phi in program units
- Variance of phi in program units
- Mean of chi in program units
- Variance of chi in program units
- Mean of phi
- Variance of phi
- Mean of chi
- Variance of chi

# Chapter 11

# twoptcorr.tsv

twoptcorr.tsv is a tab serarated file with the following fields:

- Program time
- Physical time
- Length-bin number
- Length grid points
- Program-unit length
- Physical length
- Program-unit phi two-point correlation
- Program-unit chi two-point correlation
- Physical phi two-point correlation
- Physical chi two-point correlation

# Chapter 12

# Class Index

## 12.1 Class Hierarchy

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

# Chapter 13

# Class Index

## 13.1 Class List

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

# Chapter 14

# File Index

## 14.1 File List

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

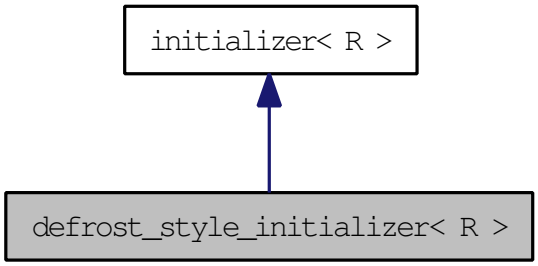
# Chapter 15

# Class Documentation

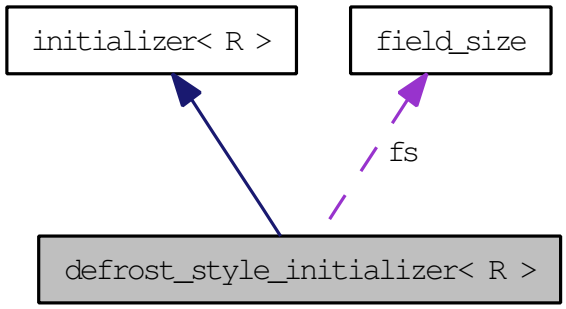
## 15.1 defrost_style_initializer< R > Class Template Reference

DEFROST-style initial conditions.

`#include <defrost_style_initializer.hpp>`Inheritance diagram for defrost_style_initializer< R >:



Collaboration diagram for defrost_style_initializer< R >:




## Public Member Functions

- **defrost_style_initializer** (field_size &fs_, model_params< R > &mp_, field< R > &phi_, field< R > &phidot_, field< R > &chi_, field< R > &chidot_, R adot_)
- virtual void initialize ()

  *Initialize the phi, phidot, chi and chidot fields.*

## Protected Member Functions

- void sample_grf (field< R > &fld, R gamma, R m2eff)

  *Sample a Gaussian random field.*

## Protected Attributes

- field_size & **fs**
- model_params< R > & **mp**
- field< R > & **phi**
- field< R > & **phidot**
- field< R > & **chi**
- field< R > & **chidot**
- R **adot**

### 15.1.1 Detailed Description

**template<typename R> class defrost_style_initializer< R >**

DEFROST-style initial conditions.

### 15.1.2 Member Function Documentation

#### 15.1.2.1 template<typename R > void defrost_style_initializer< R >::sample_grf (field< R > & *fld*, R *gamma*, R *m2eff*) [inline, protected]

Sample a Gaussian random field. Random Gaussian-mode amplitudes $b_k$ are chosen such that $< b_k b^*_{k'} >= \delta(k - k')$ using the Box-Muller transformation. The kernel function is defined as:

$$\zeta(r) = \frac{1}{\sqrt{\pi}} \int dk\, k^2 (k^2 + m_{\text{eff}})^\gamma \frac{\sin(kr)}{kr} e^{-k^2/q^2}$$

$q$ is chosen to be some scale below the Nyquist frequency.

**Parameters:**

***fld*** The field into which to store the random field sample.

***gamma*** The $(k^2 + m^2)$ exponent.

***m2eff*** The effective mass.

References field< R >::data, field< R >::ldl, and field< R >::mdata.

Referenced by defrost_style_initializer< R >::initialize().

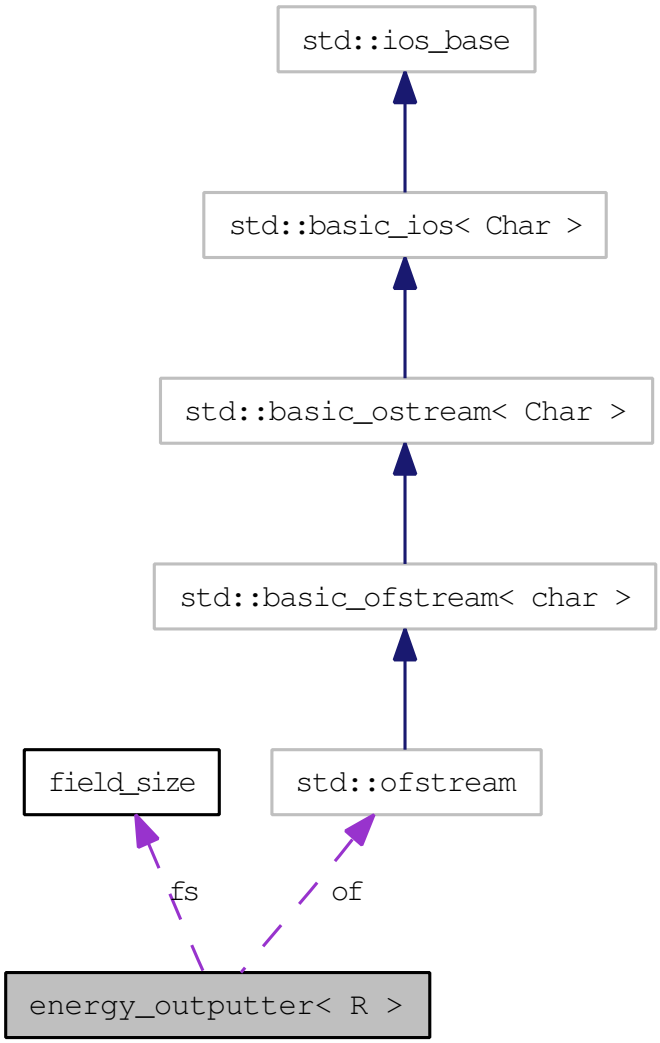
The documentation for this class was generated from the following files:

- defrost_style_initializer.hpp
- defrost_style_initializer.cpp

## 15.2 energy_outputter< R > Class Template Reference

Outputter for the energy TSV file.

`#include <energy_outputter.hpp>`Collaboration diagram for energy_outputter< R >:



## Public Member Functions

- **energy_outputter** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_, field< R > &phi_, field< R > &chi_, field< R > &phidot_, field< R > &chidot_)
- void output (bool no_output=false)

  *Compute the integrated energy density and write it to the file.*

## Public Attributes

- R avg_rho_phys

  *The average energy density in physical units.*

- R avg_rho

  *The average energy density normalized by the Friedmann equation.*

## Protected Attributes

- field_size & **fs**
- model_params< R > & **mp**
- time_state< R > & **ts**
- field< R > & **phi**
- field< R > & **chi**
- field< R > & **phidot**
- field< R > & **chidot**
- v_integrator< R > **vi**
- std::ofstream **of**

### 15.2.1 Detailed Description

**template<typename R> class energy_outputter< R >**

Outputter for the energy TSV file.

### 15.2.2 Member Function Documentation

#### 15.2.2.1 template<typename R > void energy_outputter< R >::output (bool *no_output* = `false`) [inline]

Compute the integrated energy density and write it to the file.

**Parameters:**

***no_output*** If true the result is not output to the file.

References energy_outputter< R >::avg_rho, and energy_outputter< R >::avg_rho_-phys.

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

- energy_outputter.hpp
- energy_outputter.cpp

## 15.3 fft_dft_c2r_3d_plan< R > Class Template Reference

**template<typename R> class fft_dft_c2r_3d_plan< R >**

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

- fft.hpp

## 15.4 fft_dft_c2r_3d_plan< double > Class Template Reference

### Public Types

- typedef fftw_complex **complex_t**

### Public Member Functions

- **fft_dft_c2r_3d_plan** (int n0, int n1, int n2, complex_t *in, double *out, bool estimate=true)
- void **construct** (int n0, int n1, int n2, complex_t *in, double *out, bool estimate=true)
- void **execute** ()
- bool **constructed** ()

### Protected Attributes

- fftw_plan **plan**

**template<> class fft_dft_c2r_3d_plan< double >**

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

- fft.hpp

## 15.5 fft_dft_r2c_3d_plan< R > Class Template Reference

**template<typename R> class fft_dft_r2c_3d_plan< R >**

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

- fft.hpp

## 15.6 fft_dft_r2c_3d_plan< double > Class Template Reference

### Public Types

- typedef fftw_complex **complex_t**

### Public Member Functions

- **fft_dft_r2c_3d_plan** (int n0, int n1, int n2, double *in, complex_t *out, bool estimate=true)
- void **execute** ()
- void **construct** (int n0, int n1, int n2, double *in, complex_t *out, bool estimate=true)
- bool **constructed** ()

### Protected Attributes

- fftw_plan **plan**

**template<> class fft_dft_r2c_3d_plan< double >**

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

- fft.hpp

## 15.7 fft_r2r_1d_plan< R > Class Template Reference

**template<typename R> class fft_r2r_1d_plan< R >**

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

- fft.hpp

## 15.8 fft_r2r_1d_plan< double > Class Template Reference

### Public Member Functions

- **fft_r2r_1d_plan** (int n, double *in, double *out, fft_r2r_kind kind, bool estimate=true)
- void **construct** (int n, double *in, double *out, fft_r2r_kind kind, bool estimate=true)
- void **execute** ()
- bool **constructed** ()

### Protected Attributes

- fftw_plan **plan**

**template<> class fft_r2r_1d_plan< double >**

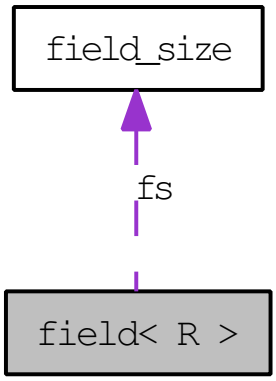
The documentation for this class was generated from the following file:

- fft.hpp

## 15.9 field< R > Class Template Reference

A three-dimensional scalar field in both position and momentum space.

`#include <field.hpp>`Collaboration diagram for field< R >:




### Public Types

- typedef fft_dft_r2c_3d_plan< R >::complex_t **complex_t**

### Public Member Functions

- **field** (field_size &fs_, bool oop=false, const char *name_=0)
- **field** (const char *name_=0)
- void **construct** (field_size &fs_, bool oop=false)
- void **divby** (R v)
- void **switch_state** (field_state state_, bool mmo=false)
- bool **is_in_place** ()

### Public Attributes

- field_size **fs**
- R * data

  *The position-space data.*

- int ldl

  *The length of the last dimension of the data array.*

- int pldl

  *The length of the last dimension of the padded data array.*

- fft_dft_c2r_3d_plan< R >::complex_t * mdata

  *The momentum-space data.*

## Protected Member Functions

- void **pad_momentum_grid** ()
- void **unpad_momentum_grid** ()

## Protected Attributes

- field_state **state**
- fft_dft_r2c_3d_plan< R > **p2m_plan**
- fft_dft_c2r_3d_plan< R > **m2p_plan**
- fft_dft_r2c_3d_plan< R > **padded_p2m_plan**
- fft_dft_c2r_3d_plan< R > **padded_m2p_plan**
- fft_dft_c2r_3d_plan< R >::complex_t * **mdata_saved**
- const char * **name**

### 15.9.1 Detailed Description

**template<typename R> class field< R >**

A three-dimensional scalar field in both position and momentum space.

### 15.9.2 Member Data Documentation

#### 15.9.2.1 template<typename R> R* field< R >::data

The position-space data.

**Note:**

The inner (z) dimension is padded to a size of 2*(fs.n/2+1).

Referenced by defrost_style_initializer< R >::sample_grf().

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

- field.hpp
- field.cpp

## 15.10 field_size Struct Reference

### Public Member Functions

- **field_size** (int n_=0, int n_pad_factor_=1)
- void **calculate_size_totals** ()

### Public Attributes

- int **n**
- int **n_pad_factor**
- int **total_gridpoints**
- int **total_padded_gridpoints**
- int **total_momentum_gridpoints**
- int **total_padded_momentum_gridpoints**
- int **power_length**

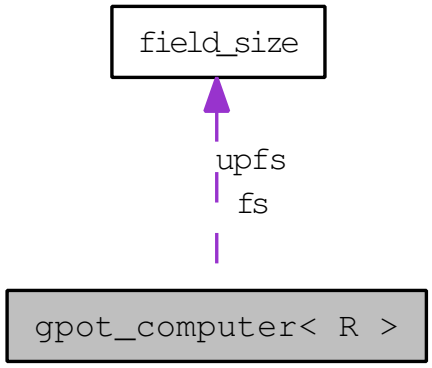
The documentation for this struct was generated from the following file:

- field_size.hpp

## 15.11 gpot_computer< R > Class Template Reference

Computer of the gravitational potential from the energy density of the phi and chi fields.

`#include <gpot_computer.hpp>`Collaboration diagram for gpot_computer< R >:




### Public Member Functions

- **gpot_computer** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_, field< R > &phi_, field< R > &chi_, field< R > &phidot_, field< R > &chidot_, grad_computer< R > &gc_)
- void compute (field_state final_state=position, bool grad_computed=false)

  *Compute gpot.*

### Public Attributes

- field< R > gpot

  *The gravitational potential field.*

### Protected Attributes

- field_size & **fs**
- field_size **upfs**
- model_params< R > & **mp**
- time_state< R > & **ts**
- field< R > & **phi**

- field< R > & **chi**
- field< R > & **phidot**
- field< R > & **chidot**
- grad_computer< R > & **gc**

## 15.11.1 Detailed Description

**template<typename R> class gpot_computer< R >**

Computer of the gravitational potential from the energy density of the phi and chi fields.

## 15.11.2 Member Function Documentation

### 15.11.2.1 template<typename R > void gpot_computer< R >::compute (field_state *final_state* = `position`, bool *grad_computed* = `false`) `[inline]`

Compute gpot.

**Parameters:**

***final_state*** The final state of gpot.

***grad_computed*** True if the gradient fields have already been computed (otherwise gc.compute() is called).

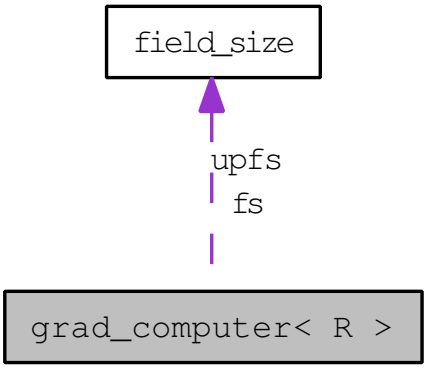
References gpot_computer< R >::gpot.

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

- gpot_computer.hpp
- gpot_computer.cpp

## 15.12 grad_computer< R > Class Template Reference

Collaboration diagram for grad_computer< R >:




### Public Member Functions

- **grad_computer** (field_size &fs_, model_params< R > &mp_, field< R > &phi_, field< R > &chi_)
- void **compute** (field_state final_state=position)

### Public Attributes

- field< R > **phigradx**
- field< R > **chigradx**
- field< R > **phigrady**
- field< R > **chigrady**
- field< R > **phigradz**
- field< R > **chigradz**

### Protected Attributes

- field_size & **fs**
- field_size **upfs**
- model_params< R > & **mp**
- field< R > & **phi**
- field< R > & **chi**

**template<typename R> class grad_computer< R >**

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

- grad_computer.hpp
- grad_computer.cpp

## 15.13 grid_funcs< R > Struct Template Reference

### Static Public Member Functions

- static R **compute_energy_scaling** (model_params< R > &mp, time_state< R > &ts)
- static R **compute_phi** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_chi** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_gpot** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_V_phys** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_V** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_T_phi_phys** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_T_phi** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_T_chi_phys** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_T_chi** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_G_phi_phys** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_G_phi** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_G_chi_phys** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_G_chi** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)

- static R **compute_rho_phys** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_rho** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_p_phys** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)
- static R **compute_p** (field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)

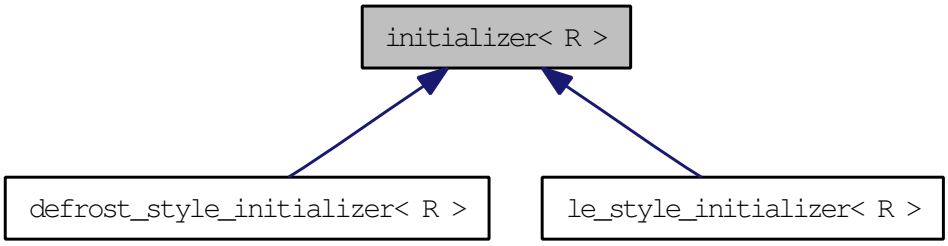
**template<typename R> struct grid_funcs< R >**

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

- grid_funcs.hpp
- grid_funcs.cpp

## 15.14 initializer< R > Class Template Reference

Inheritance diagram for initializer< R >:




### Public Member Functions

- virtual void **initialize** ()=0

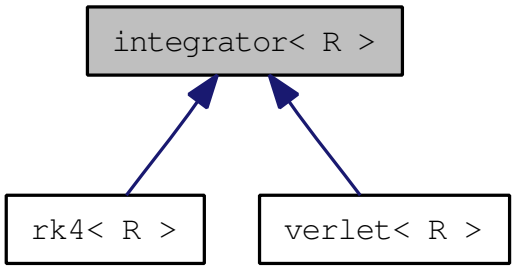
**template<typename R> class initializer< R >**

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

- initializer.hpp

## 15.15 integrator< R > Class Template Reference

Inheritance diagram for integrator< R >:




### Public Member Functions

- virtual void **step** ()=0
- virtual void **initialize** ()=0

### Static Public Member Functions

- static void **avg_gradients** (field_size &fs, model_params< R > &mp, field< R > &phi, field< R > &chi, R &avg_gradient_phi, R &avg_gradient_chi)

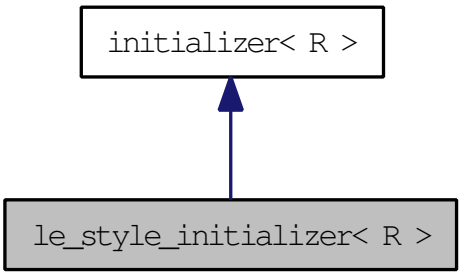
**template<typename R> class integrator< R >**

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

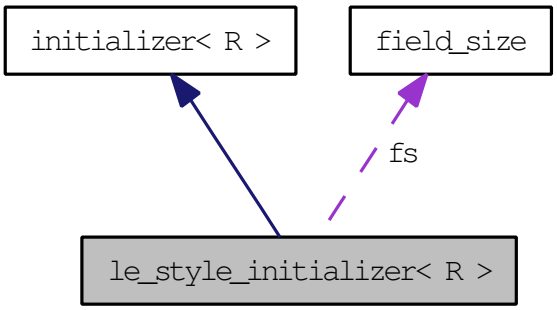
- integrator.hpp
- integrator.cpp

## 15.16 le_style_initializer< R > Class Template Reference

Inheritance diagram for le_style_initializer< R >:




Collaboration diagram for le_style_initializer< R >:




### Public Member Functions

- **le_style_initializer** (field_size &fs_, model_params< R > &mp_, field< R > &phi_, field< R > &phidot_, field< R > &chi_, field< R > &chidot_, R adot_, R len0)
- virtual void **initialize** ()

### Protected Member Functions

- void **set_mode** (int px, int py, int pz, int idx, bool real=false)

## Protected Attributes

- field_size & **fs**
- model_params< R > & **mp**
- field< R > & **phi**
- field< R > & **phidot**
- field< R > & **chi**
- field< R > & **chidot**
- R **adot**
- R **fluctuation_amplitude**

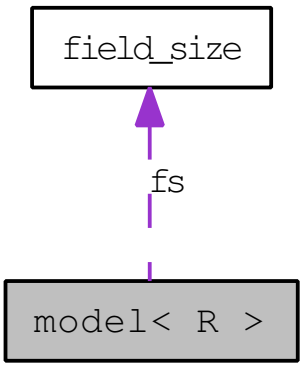
**template<typename R> class le_style_initializer< R >**

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

- le_style_initializer.hpp
- le_style_initializer.cpp

## 15.17 model< R > Class Template Reference

Collaboration diagram for model< R >:




### Public Member Functions

- **model** (int argc, char *argv[ ])
- void **run** ()

### Protected Member Functions

- void **set_output_directory** (const char *uodn)
- void **write_info_file** ()
- void **set_initial_conditions** ()
- void **evolve** (integrator< R > *ig)
- void **private_allocate** ()
- void **private_set_sf_info** ()
- void **private_evolve** (int counter)
- void **private_info_file_output** (std::ofstream &info_file)

### Protected Attributes

- field_size **fs**
- model_params< R > **mp**
- time_state< R > **ts**
- bool **use_verlet**
- bool **le_init**
- bool **homo_ic_phi**

- bool **homo_ic_chi**
- int **seed**
- R **tf**
- int **scale_interval**
- int **energy_interval**
- int **spectra_interval**
- int **screen_interval**
- int **slice_interval**
- int **stats_interval**
- int **twoptcorr_interval**
- field< R > **phi**
- field< R > **phidot**
- field< R > **chi**
- field< R > **chidot**
- grad_computer< R > * **gc**
- gpot_computer< R > * **gpotc**
- slice_output_manager< R > * **som**
- R **ics_scale**
- R **len0**
- bool **vvwl**
- R **af**

**template<typename R> class model< R >**

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

- model.hpp
- model.cpp

## 15.18 model_params< R > Struct Template Reference

Static model parameters.

`#include <model_params.hpp>`

### Public Member Functions

- void **calculate_derived_params** (bool report=false)
- R V (R phi, R chi, R a_t)

  *Returns the value of the field potential at a point given the values of the fields at that point.*

- void derivs (R phi, R chi, R phidot, R chidot, R chi2phi, R phi2chi, R phi3, R chi3, R phi5, R chi5, R a_t, R adot_t, R addot_t, R mom2, R &dphidt, R &dchidt, R &dphidotdt, R &dchidotdt)

  *This is where the equations of motion for the fields are actually evaluated.*

- R adoubledot (R a_t, R adot_t, R avg_gradient_phi, R avg_gradient_chi, R avg_V)

  *Returns the second time derivative of the scale factor in program units.*

- R adoubledot_staggered (R dt, R a_t, R adot_t, R avg_gradient_phi, R avg_gradient_chi, R avg_V)

  *Returns the second time derivative of the scale factor in program units at a half-time-step.*

### Public Attributes

- R **gamma_phi**
- R **gamma_chi**
- R **lambda_phi**
- R **lambda_chi**
- R **g**
- R **m_phi**
- R **m_chi**
- R **len**
- R **phi0**
- R **chi0**
- R **phidot0**
- R **chidot0**
- R **rescale_A**

- R **rescale_B**
- R **rescale_s**
- R **rescale_r**
- R **dp**

## 15.18.1 Detailed Description

**template<typename R> struct model_params< R >**

Static model parameters.

## 15.18.2 Member Function Documentation

### 15.18.2.1 template<typename R> R model_params< R >::adoubledot (R *a_t*, R *adot_t*, R *avg_gradient_phi*, R *avg_gradient_chi*, R *avg_V*) [inline]

Returns the second time derivative of the scale factor in program units. See equation 6.26 of the LatticeEasy manual.

### 15.18.2.2 template<typename R> R model_params< R >::adoubledot_staggered (R *dt*, R *a_t*, R *adot_t*, R *avg_gradient_phi*, R *avg_gradient_chi*, R *avg_V*) [inline]

Returns the second time derivative of the scale factor in program units at a half-time-step. See equation 6.35/6.36 of the LatticeEasy manual.

### 15.18.2.3 template<typename R> void model_params< R >::derivs (R *phi*, R *chi*, R *phidot*, R *chidot*, R *chi2phi*, R *phi2chi*, R *phi3*, R *chi3*, R *phi5*, R *chi5*, R *a_t*, R *adot_t*, R *addot_t*, R *mom2*, R & *dphidt*, R & *dchidt*, R & *dphidotdt*, R & *dchidotdt*) [inline]

This is where the equations of motion for the fields are actually evaluated. The first and second time derivatives of the fields are computed in accordance with the Klein-Gordon equation, which is written in program units and transformed to momentum-space. Note that the choice of program units has eliminated the first-time-derivative term from the second-time-derivative equation.

#### 15.18.2.4 template<typename R> R model_params< R >::V (R *phi*, R *chi*, R *a_t*) [inline]

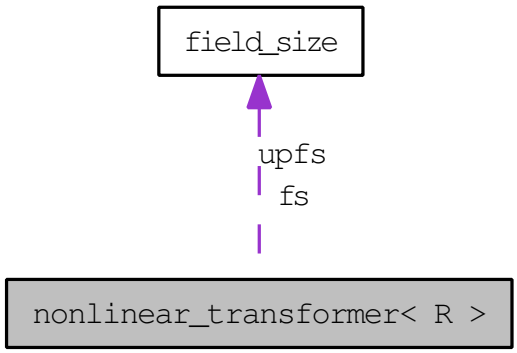
Returns the value of the field potential at a point given the values of the fields at that point. The field values are sent in program units, and the potential is returned in program units. This is equation 6.5 from the LatticeEasy manual.

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

- model_params.hpp

# 15.19 nonlinear_transformer< R > Class Template Reference

Collaboration diagram for nonlinear_transformer< R >:




## Public Member Functions

- **nonlinear_transformer** (field_size &fs_, model_params< R > &mp_)
- void **transform** (field< R > &phi, field< R > &chi, field_state final_state=momentum)

## Public Attributes

- field< R > **phi2chi**
- field< R > **chi2phi**
- field< R > **phi3**
- field< R > **chi3**
- field< R > **phi5**
- field< R > **chi5**

## Protected Attributes

- field_size & **fs**
- field_size **upfs**
- model_params< R > & **mp**

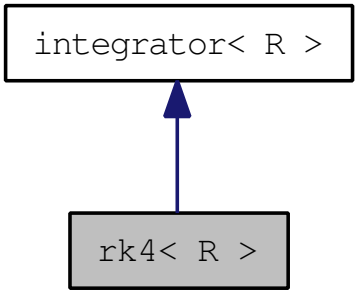
**template<typename R> class nonlinear_transformer< R >**

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

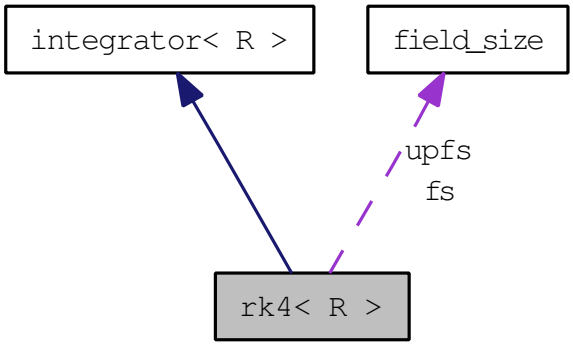
- nonlinear_transformer.hpp
- nonlinear_transformer.cpp

## 15.20 rk4< R > Class Template Reference

Inheritance diagram for rk4< R >:




Collaboration diagram for rk4< R >:




### Public Member Functions

- **rk4** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_, field< R > &phi_, field< R > &phidot_, field< R > &chi_, field< R > &chidot_)
- virtual void **step** ()
- virtual void **initialize** ()

### Protected Member Functions

- void **substep_scale** (R fac, field< R > &phip, field< R > &chip, R ap, R adotp, R ptp, R &an, R &adotn, R &ptn, R &dan, R &dadotn, R &dptn, R &avg_gradient_phi, R &avg_gradient_chi)

- void **substep** (R fac, field< R > &phip, field< R > &chip, field< R > &phidotp, field< R > &chidotp, field< R > &phin, field< R > &chin, field< R > &phidotn, field< R > &chidotn, R ap, R adotp, R ptp, R &an, R &adotn, R &ptn, R &avg_gradient_phi, R &avg_gradient_chi)

## Protected Attributes

- field_size & **fs**
- field_size **upfs**
- model_params< R > & **mp**
- time_state< R > & **ts**
- field< R > & **phi**
- field< R > & **phidot**
- field< R > & **chi**
- field< R > & **chidot**
- field< R > **phi1**
- field< R > **phidot1**
- field< R > **chi1**
- field< R > **chidot1**
- field< R > **phi2**
- field< R > **phidot2**
- field< R > **chi2**
- field< R > **chidot2**
- field< R > **phi3**
- field< R > **phidot3**
- field< R > **chi3**
- field< R > **chidot3**
- nonlinear_transformer< R > **nlt**
- v_integrator< R > **vi**
- R **a1**
- R **a2**
- R **a3**
- R **a4**
- R **adot1**
- R **adot2**
- R **adot3**
- R **adot4**
- R **pt1**
- R **pt2**
- R **pt3**
- R **pt4**

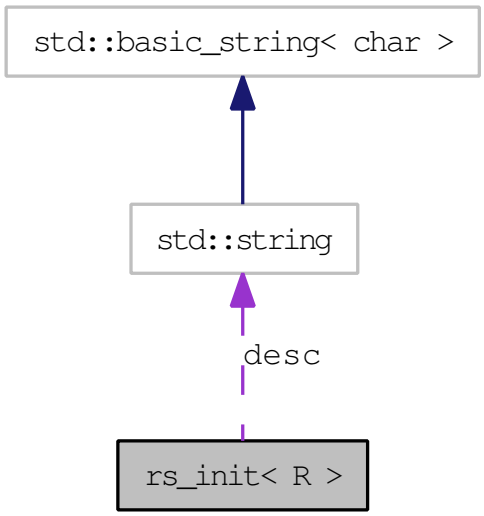
**template<typename R> class rk4< R >**

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

- rk4.hpp
- rk4.cpp

## 15.21 rs_init< R > Struct Template Reference

Collaboration diagram for rs_init< R >:




### Public Member Functions

- **rs_init** (R m, R B, R s, R r, R A, const std::string &d)
- bool **operator**< (const rs_init &rs) const

### Public Attributes

- R **mag**
- R **rescale_B**
- R **rescale_s**
- R **rescale_r**
- R **rescale_A**
- std::string **desc**

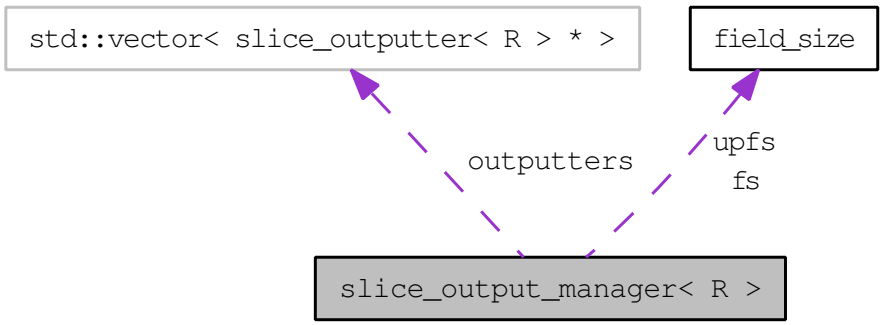
**template<typename R> struct rs_init< R >**

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

- model_params.hpp

# 15.22 slice_output_manager< R > Class Template Reference

Collaboration diagram for slice_output_manager< R >:




## Public Types

- typedef slice_outputter< R >::var_func **var_func**

## Public Member Functions

- **slice_output_manager** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_, field< R > &phi_, field< R > &chi_, field< R > &phidot_, field< R > &chidot_, grad_computer< R > &gc_, gpot_computer< R > &gpotc_, int slicedim_=3, int slicelength_=0, int sliceskip_=1, bool sliceaverage_=true)
- void **add_outputter** (std::string varname, var_func vf)
- void **output** ()

## Protected Attributes

- field_size & **fs**
- field_size **upfs**
- model_params< R > & **mp**
- time_state< R > & **ts**
- field< R > & **phi**
- field< R > & **chi**
- field< R > & **phidot**
- field< R > & **chidot**

- grad_computer< R > & **gc**
- gpot_computer< R > & **gpotc**
- int **slicedim**
- int **slicelength**
- int **sliceskip**
- bool **sliceaverage**
- int **bin_idx**
- std::vector< slice_outputter< R > * > **outputters**

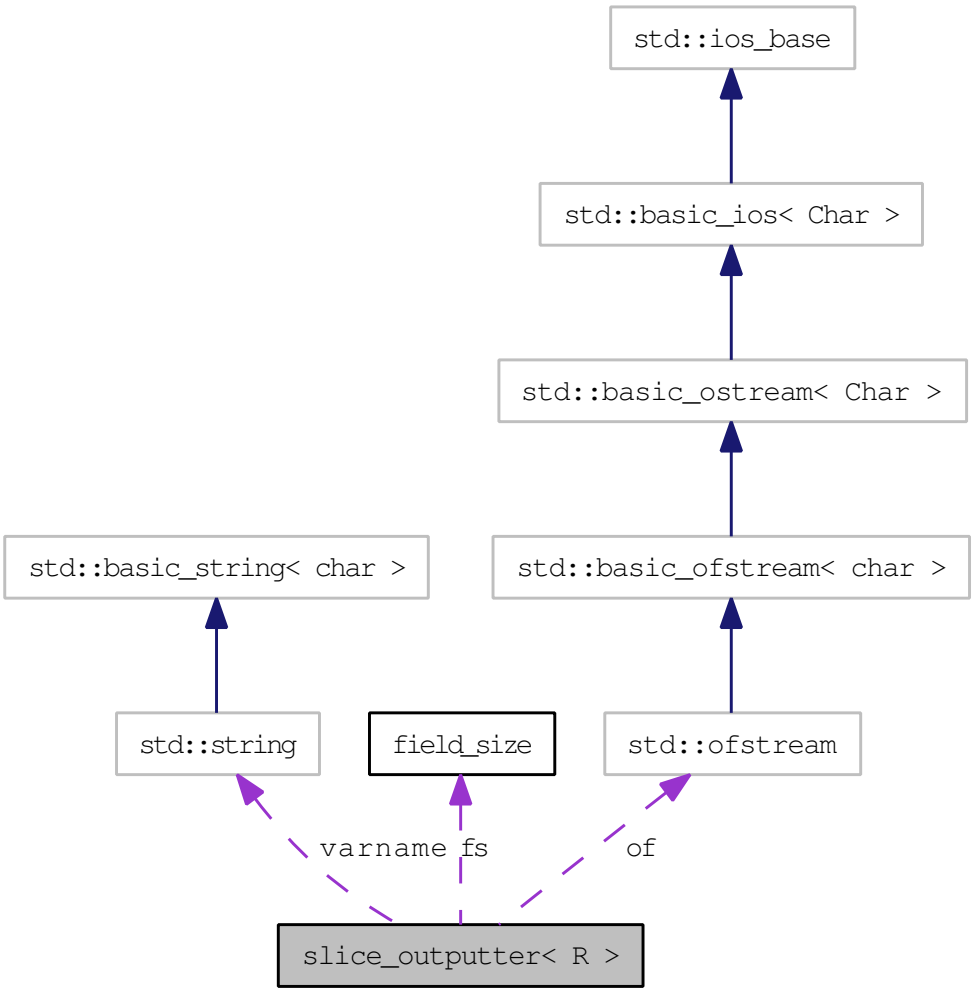
**template<typename R> class slice_output_manager< R >**

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

- slice_output_manager.hpp
- slice_output_manager.cpp

## 15.23 slice_outputter< R > Class Template Reference

Collaboration diagram for slice_outputter< R >:




### Public Types

- typedef R(∗ **var_func** )(field_size &fs, model_params< R > &mp, time_state< R > &ts, R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)

## Public Member Functions

- **slice_outputter** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_, int slicelength_, std::string varname_, var_func vf_)
- void **begin** (int bin_idx)
- void **flush** ()
- void **advance** ()
- void **accumulate** (R phi, R chi, R phidot, R chidot, R phigradx, R chigradx, R phigrady, R chigrady, R phigradz, R chigradz, R gpot)

## Protected Attributes

- field_size & **fs**
- model_params< R > & **mp**
- time_state< R > & **ts**
- int **slicelength**
- std::string **varname**
- var_func **vf**
- R * **buffer**
- float * **bufferf**
- std::ofstream **of**
- int **cp**
- int **cn**

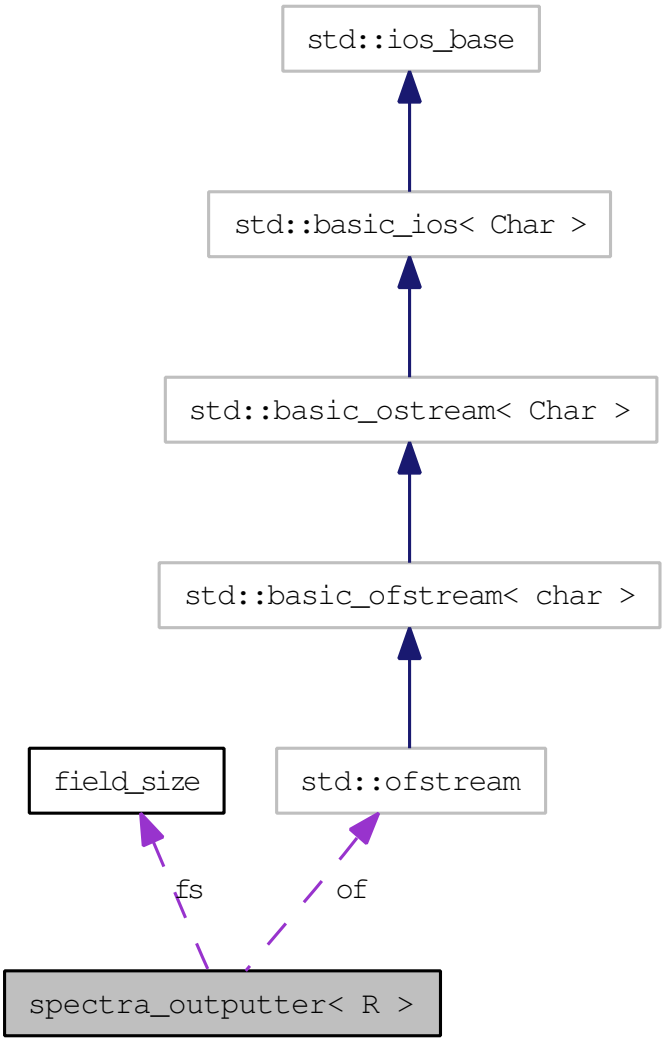
**template<typename R> class slice_outputter< R >**

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

- slice_outputter.hpp
- slice_outputter.cpp

## 15.24 spectra_outputter< R > Class Template Reference

Collaboration diagram for spectra_outputter< R >:




### Public Member Functions

- **spectra_outputter** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_, field< R > &phi_, field< R > &chi_)

- void **output** ()

## Protected Attributes

- field_size & **fs**
- model_params< R > & **mp**
- time_state< R > & **ts**
- field< R > & **phi**
- field< R > & **chi**
- std::ofstream **of**
- R * **phi_total**
- R * **chi_total**
- int * **counts**

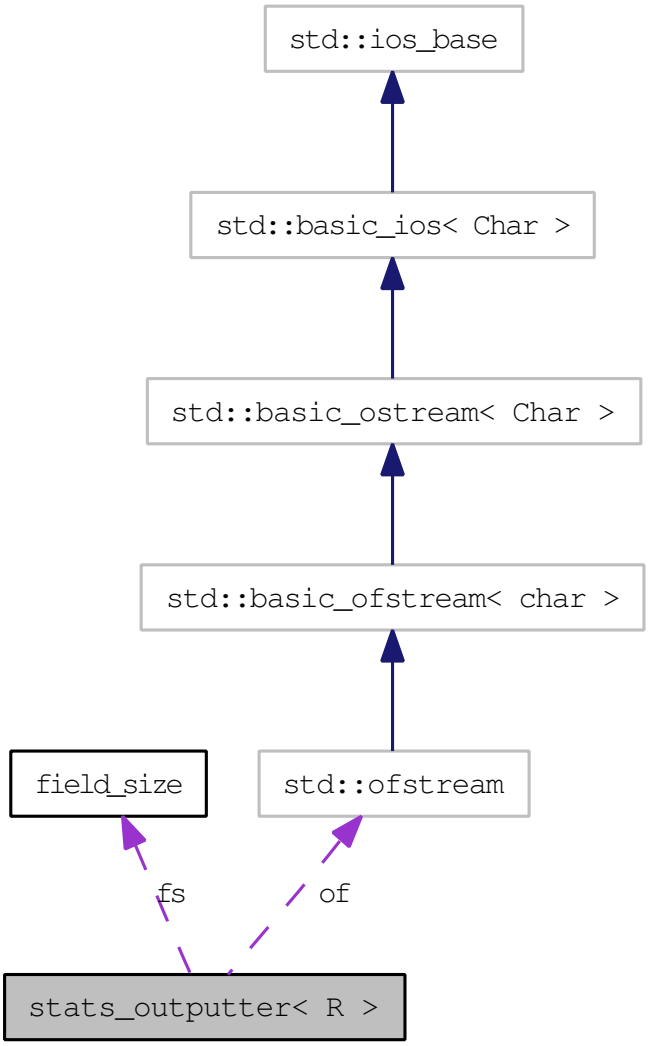
**template<typename R> class spectra_outputter< R >**

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

- spectra_outputter.hpp
- spectra_outputter.cpp

## 15.25 stats_outputter< R > Class Template Reference

Collaboration diagram for stats_outputter< R >:




### Public Member Functions

- **stats_outputter** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_, field< R > &phi_, field< R > &chi_)
- void **output** ()

## Protected Attributes

- field_size & **fs**
- model_params< R > & **mp**
- time_state< R > & **ts**
- field< R > & **phi**
- field< R > & **chi**
- std::ofstream **of**

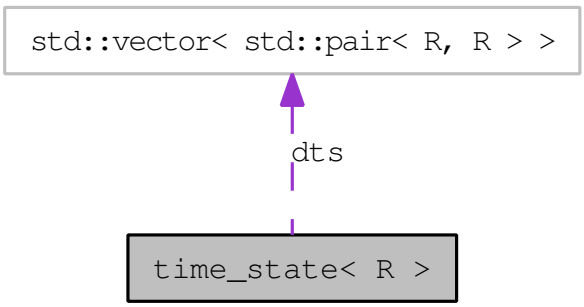
**template<typename R> class stats_outputter< R >**

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

- stats_outputter.hpp
- stats_outputter.cpp

# 15.26 time_state< R > Struct Template Reference

Collaboration diagram for time_state< R >:




## Public Member Functions

- void **advance** ()
- void **add_dt** (R start_time, R dt_)
- void **finalize_dts** ()
- void **dt_summary** (std::ostream &os)

## Static Public Member Functions

- static R **default_dt** ()

## Public Attributes

- R **t**
- R **physical_time**
- R **a**
- R **adot**
- R **addot**
- R **dt**

## Protected Attributes

- std::vector< std::pair< R, R > > **dts**

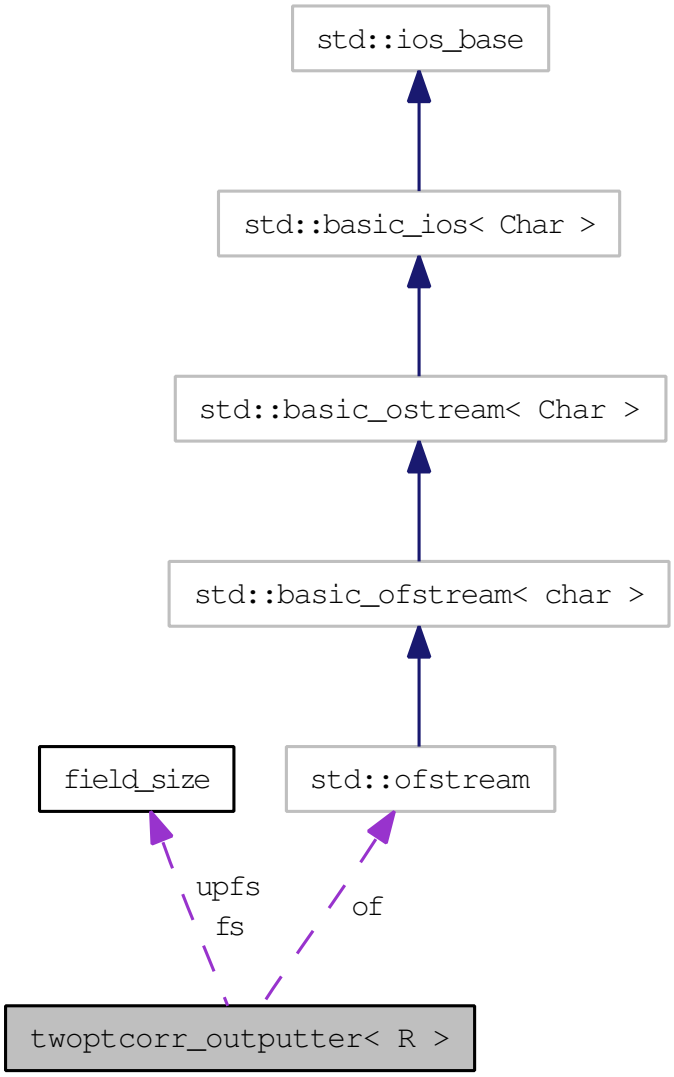
**template<typename R> struct time_state< R >**

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

- time_state.hpp

## 15.27 twoptcorr_outputter< R > Class Template Reference

Collaboration diagram for twoptcorr_outputter< R >:



## Public Member Functions

- **twoptcorr_outputter** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_, field< R > &phi_, field< R > &chi_)
- void **output** ()

## Protected Attributes

- field_size & **fs**
- field_size **upfs**
- model_params< R > & **mp**
- time_state< R > & **ts**
- field< R > & **phi**
- field< R > & **chi**
- std::ofstream **of**
- R * **phi_total**
- R * **chi_total**
- int * **counts**
- int **dmax**
- field< R > **corr**

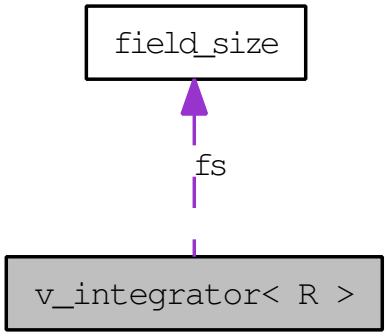
**template<typename R> class twoptcorr_outputter< R >**

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

- twoptcorr_outputter.hpp
- twoptcorr_outputter.cpp

## 15.28 v_integrator< R > Class Template Reference

Collaboration diagram for v_integrator< R >:




### Public Member Functions

- **v_integrator** (field_size &fs_, model_params< R > &mp_)
- R **integrate** (field< R > &phi, field< R > &chi, R a_t)

### Protected Attributes

- field_size & **fs**
- model_params< R > & **mp**
- R * **y_integral**
- R * **z_integral**

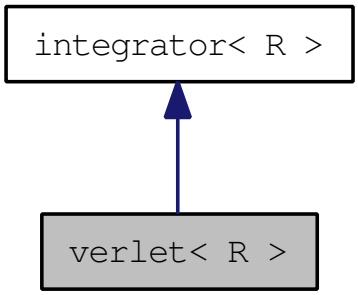
**template<typename R> class v_integrator< R >**

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

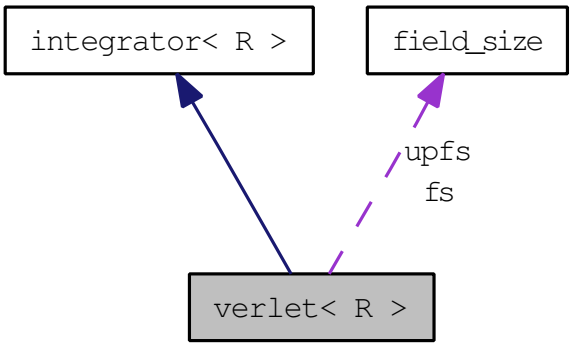
- v_integrator.hpp
- v_integrator.cpp

## 15.29 verlet< R > Class Template Reference

Inheritance diagram for verlet< R >:




Collaboration diagram for verlet< R >:




### Public Member Functions

- **verlet** (field_size &fs_, model_params< R > &mp_, time_state< R > &ts_-, field< R > &phi_, field< R > &phidot_, field< R > &chi_, field< R > &chidot_)
- virtual void **step** ()
- virtual void **initialize** ()

### Protected Attributes

- field_size & **fs**
- field_size **upfs**

- model_params< R > & **mp**
- time_state< R > & **ts**
- field< R > & **phi**
- field< R > & **phidot**
- field< R > & **chi**
- field< R > & **chidot**
- field< R > **phiddot**
- field< R > **phidot_staggered**
- field< R > **chiddot**
- field< R > **chidot_staggered**
- nonlinear_transformer< R > **nlt**
- v_integrator< R > **vi**
- R **addot**
- R **adot_staggered**
- R **dptdt**
- R **ddptdt**
- R **dptdt_staggered**

**template<typename R> class verlet< R >**

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

- verlet.hpp
- verlet.cpp

# Chapter 16

# File Documentation

## 16.1 defrost_style_initializer.hpp File Reference

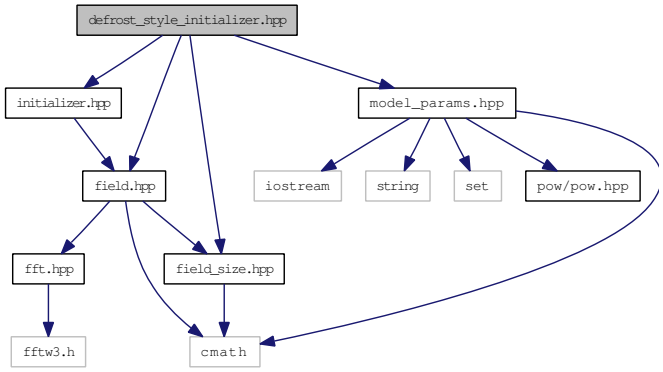
DEFROST-style initial conditions. `#include "field_size.hpp"`

`#include "model_params.hpp"`

`#include "field.hpp"`

`#include "initializer.hpp"`

Include dependency graph for defrost_style_initializer.hpp:




### Classes

- class defrost_style_initializer< R >

  *DEFROST-style initial conditions.*

### 16.1.1 Detailed Description

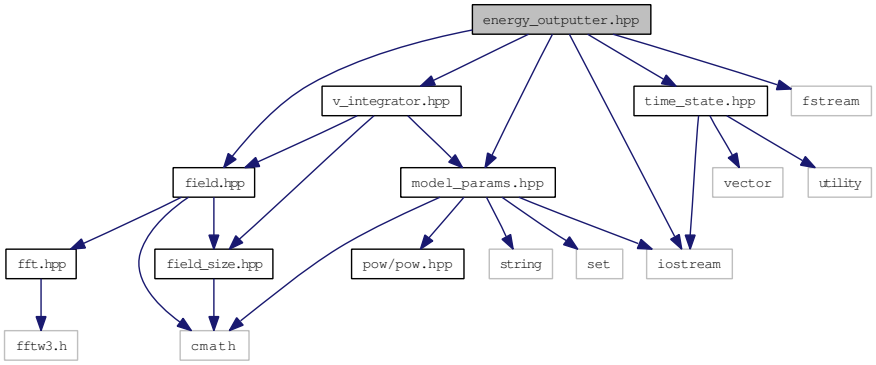
DEFROST-style initial conditions.

## 16.2 energy_outputter.hpp File Reference

Outputter for the energy TSV file. `#include "field.hpp"`

```
#include "model_params.hpp"
#include "time_state.hpp"
#include "v_integrator.hpp"
#include <iostream>
#include <fstream>
```

Include dependency graph for energy_outputter.hpp:




### Classes

- class energy_outputter< R >

  *Outputter for the energy TSV file.*

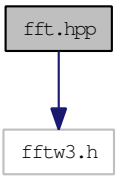
### 16.2.1 Detailed Description

Outputter for the energy TSV file.

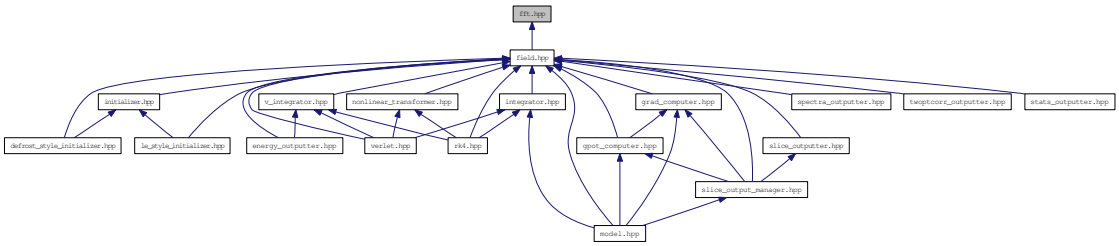
## 16.3 fft.hpp File Reference

FFT wrappers. `#include <fftw3.h>`

Include dependency graph for fft.hpp:




This graph shows which files directly or indirectly include this file:




### Classes

- class fft_r2r_1d_plan< R >
- class fft_r2r_1d_plan< double >
- class fft_dft_c2r_3d_plan< R >
- class fft_dft_c2r_3d_plan< double >
- class fft_dft_r2c_3d_plan< R >
- class fft_dft_r2c_3d_plan< double >

### Enumerations

- enum **fft_r2r_kind** {

  **r2hc** = FFTW_R2HC, **hc2r** = FFTW_HC2R, **dht** = FFTW_DHT, **redft00** = FFTW_REDFT00,

  **redft10** = FFTW_REDFT10, **redft01** = FFTW_REDFT01, **redft11** = FFTW_-REDFT11, **rodft00** = FFTW_RODFT00,

  **rodft10** = FFTW_RODFT10, **rodft01** = FFTW_RODFT01, **rodft11** = FFTW_-RODFT11 }

## Functions

- template<typename R >
  R * **fft_malloc** (size_t sz)
- template<>
  double * **fft_malloc**< **double** > (size_t sz)
- template<typename R >
  void **fft_free** (R *ptr)
- template<>
  void **fft_free**< **double** > (double *ptr)

### 16.3.1 Detailed Description

FFT wrappers.

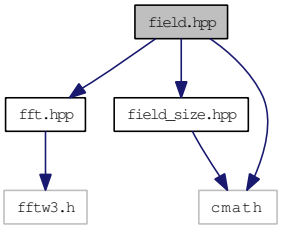
## 16.4 field.hpp File Reference

Three-dimensional scalar fields. `#include "fft.hpp"`

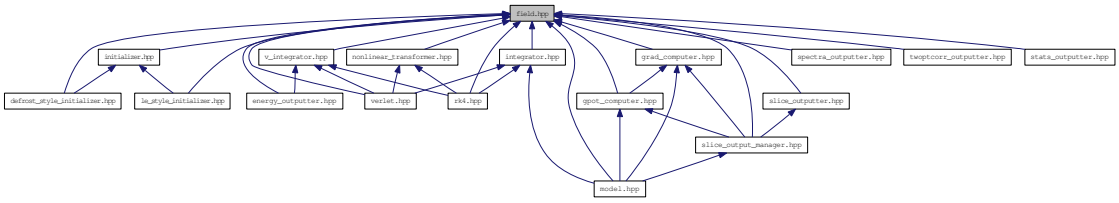
`#include "field_size.hpp"`

`#include <cmath>`

Include dependency graph for field.hpp:




This graph shows which files directly or indirectly include this file:




### Classes

- class field< R >

  *A three-dimensional scalar field in both position and momentum space.*

### Enumerations

- enum **field_state** {

  **uninitialized**, **position**, **momentum**, **padded_position**,

  **padded_momentum** }

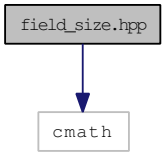
### 16.4.1 Detailed Description

Three-dimensional scalar fields.

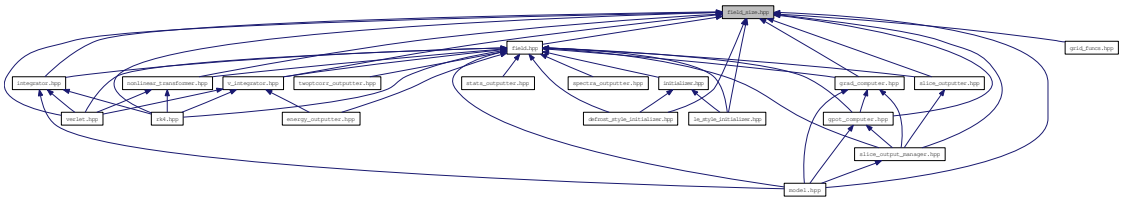
## 16.5 field_size.hpp File Reference

Field grid size and derived size-related quantities. `#include <cmath>`

Include dependency graph for field_size.hpp:




This graph shows which files directly or indirectly include this file:



### Classes

- struct field_size

### 16.5.1 Detailed Description

Field grid size and derived size-related quantities.

## 16.6 gpot_computer.hpp File Reference

Gravitational-potential computations. `#include "field_size.hpp"`

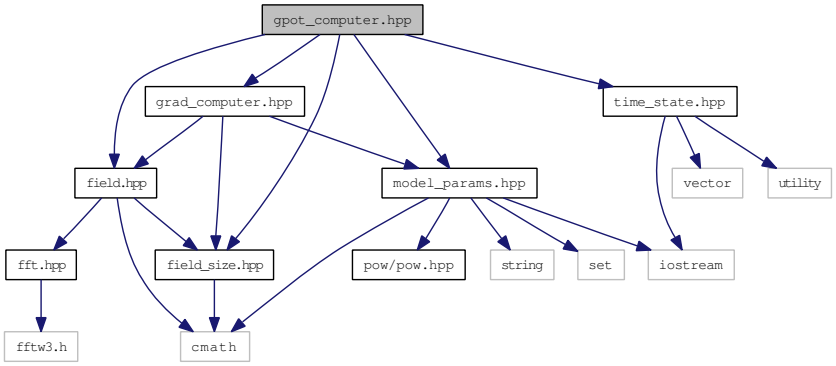
`#include "model_params.hpp"`

`#include "time_state.hpp"`

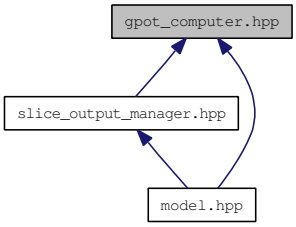
`#include "field.hpp"`

`#include "grad_computer.hpp"`

Include dependency graph for gpot_computer.hpp:




This graph shows which files directly or indirectly include this file:




## Classes

- class gpot_computer< R >

  *Computer of the gravitational potential from the energy density of the phi and chi fields.*

### 16.6.1 Detailed Description

Gravitational-potential computations.

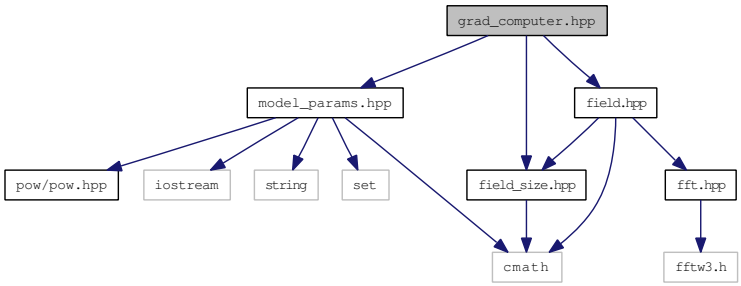
# 16.7 grad_computer.hpp File Reference

Computation of the gradient in Fourier space. `#include "field_size.hpp"`

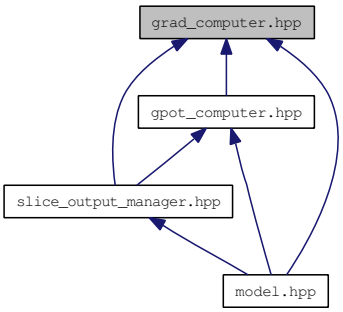
`#include "model_params.hpp"`

`#include "field.hpp"`

Include dependency graph for grad_computer.hpp:




This graph shows which files directly or indirectly include this file:




## Classes

- class grad_computer< R >

### 16.7.1 Detailed Description

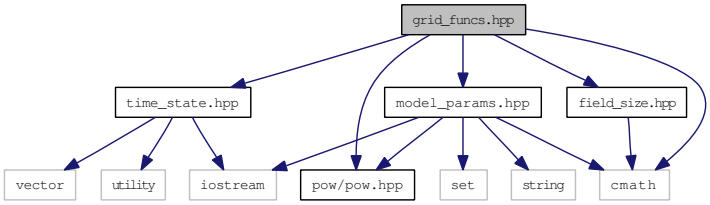
Computation of the gradient in Fourier space.

## 16.8 grid_funcs.hpp File Reference

Grid point functions used for slice output, etc. `#include "pow/pow.hpp"`

```
#include "model_params.hpp"
#include "field_size.hpp"
#include "time_state.hpp"
#include <cmath>
```

Include dependency graph for grid_funcs.hpp:




### Classes

- struct grid_funcs< R >

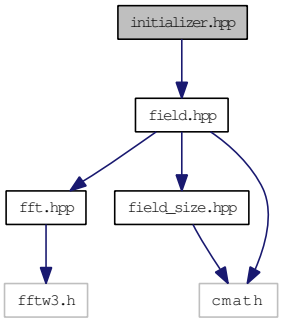
### 16.8.1 Detailed Description

Grid point functions used for slice output, etc.

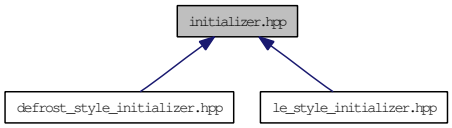
# 16.9 initializer.hpp File Reference

Generic field-initialization. `#include "field.hpp"`

Include dependency graph for initializer.hpp:




This graph shows which files directly or indirectly include this file:




## Classes

- class initializer< R >

### 16.9.1 Detailed Description

Generic field-initialization.

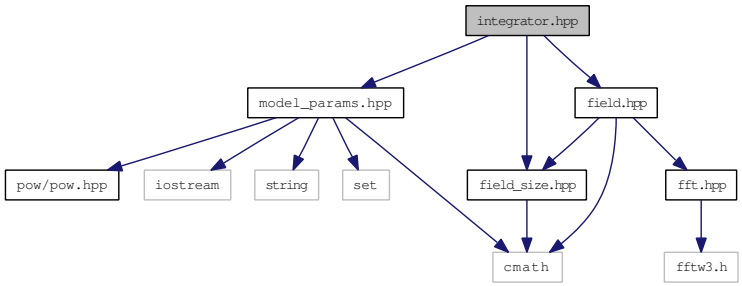
## 16.10 integrator.hpp File Reference

Generic time-step evolution. `#include "field_size.hpp"`

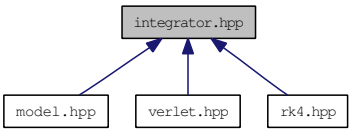
`#include "model_params.hpp"`

`#include "field.hpp"`

Include dependency graph for integrator.hpp:




This graph shows which files directly or indirectly include this file:




### Classes

- class integrator< R >

### 16.10.1 Detailed Description

Generic time-step evolution.

# 16.11 le_style_initializer.hpp File Reference

LatticeEasy-style initialization. `#include "field_size.hpp"`

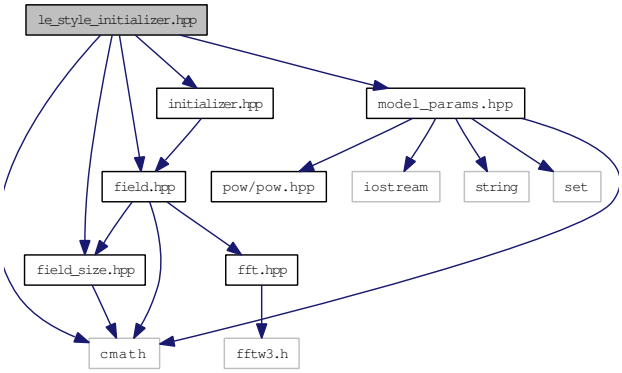
`#include "model_params.hpp"`

`#include "field.hpp"`

`#include "initializer.hpp"`

`#include <cmath>`

Include dependency graph for le_style_initializer.hpp:




## Classes

- class le_style_initializer< R >

### 16.11.1 Detailed Description

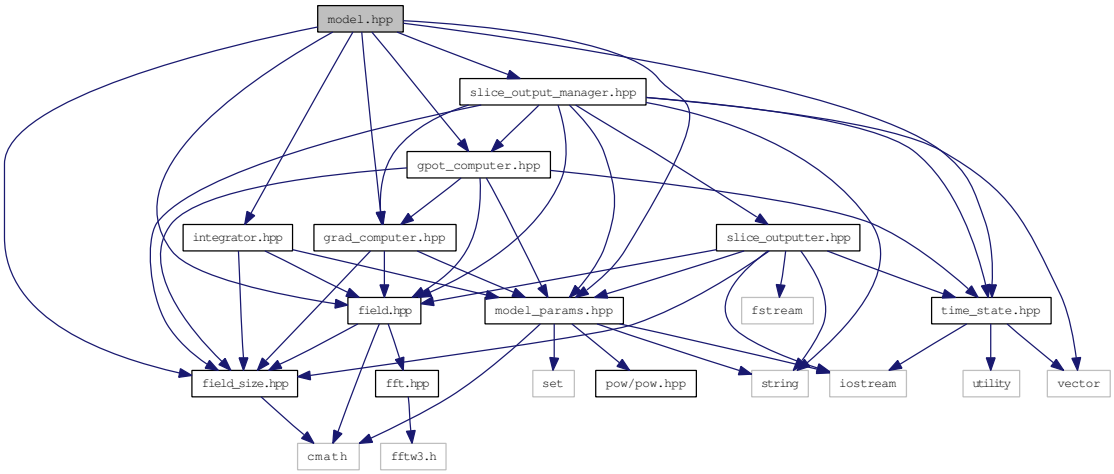
LatticeEasy-style initialization.

## 16.12 model.hpp File Reference

A particular simulated situation. `#include "field_size.hpp"`

```
#include "model_params.hpp"
#include "time_state.hpp"
#include "field.hpp"
#include "integrator.hpp"
#include "slice_output_manager.hpp"
#include "grad_computer.hpp"
#include "gpot_computer.hpp"
```

Include dependency graph for model.hpp:




### Classes

- class model< R >

### 16.12.1 Detailed Description

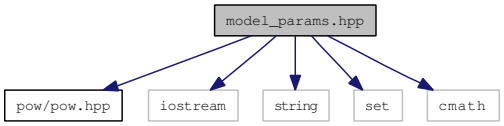
A particular simulated situation.

# 16.13 model_params.hpp File Reference

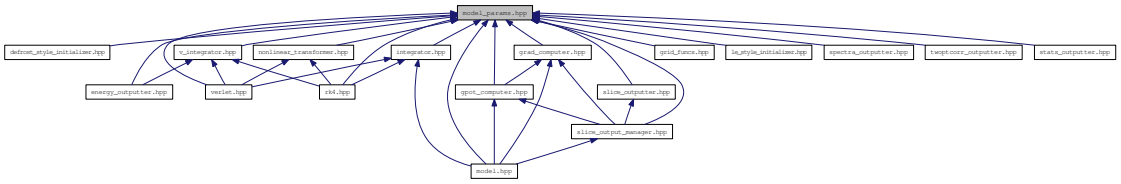
The physical model parameters. `#include "pow/pow.hpp"`

```
#include <iostream>
#include <string>
#include <set>
#include <cmath>
```

Include dependency graph for model_params.hpp:




This graph shows which files directly or indirectly include this file:



## Classes

- struct rs_init< R >
- struct model_params< R >

  *Static model parameters.*

## Functions

- template<typename R >
  bool **operator**< (const rs_init< R > &rs1, const rs_init< R > &rs2)

### 16.13.1 Detailed Description

The physical model parameters.

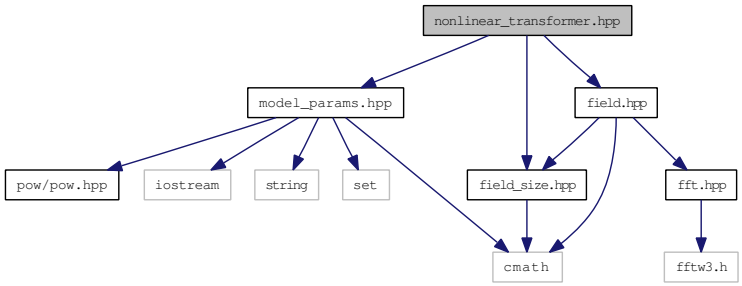
## 16.14 nonlinear_transformer.hpp File Reference

Momentum-space representations of nonlinear field terms. `#include "field_size.hpp"`

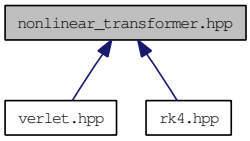
`#include "model_params.hpp"`

`#include "field.hpp"`

Include dependency graph for nonlinear_transformer.hpp:




This graph shows which files directly or indirectly include this file:




### Classes

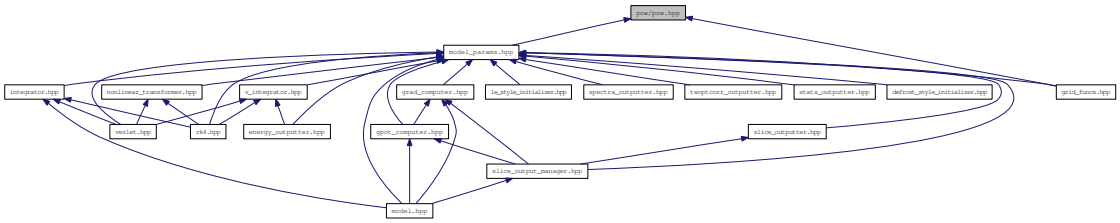
- class nonlinear_transformer< R >

### 16.14.1 Detailed Description

Momentum-space representations of nonlinear field terms.

# 16.15 pow/pow.hpp File Reference

Template function to compute the integer power of its argument. This graph shows which files directly or indirectly include this file:




## 16.15.1 Detailed Description

Template function to compute the integer power of its argument.

## 16.16 rk4.hpp File Reference

Fourth-order Runge–Kutta (RK4) integrator. `#include "field_size.hpp"`

`#include "model_params.hpp"`

`#include "time_state.hpp"`

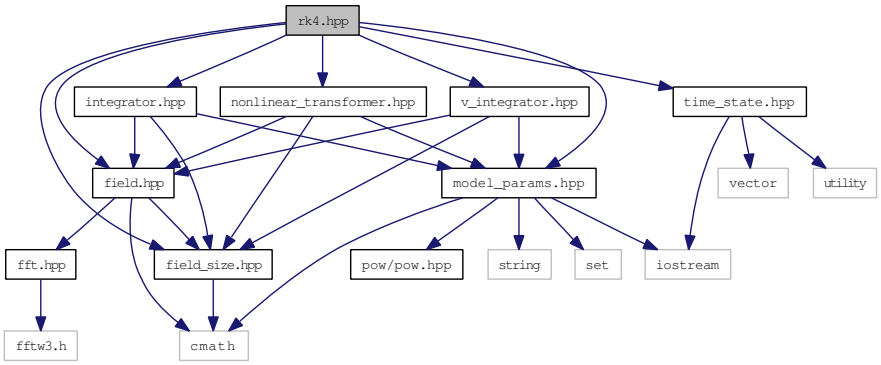
`#include "field.hpp"`

`#include "integrator.hpp"`

`#include "nonlinear_transformer.hpp"`

`#include "v_integrator.hpp"`

Include dependency graph for rk4.hpp:




### Classes

- class rk4< R >

### 16.16.1 Detailed Description

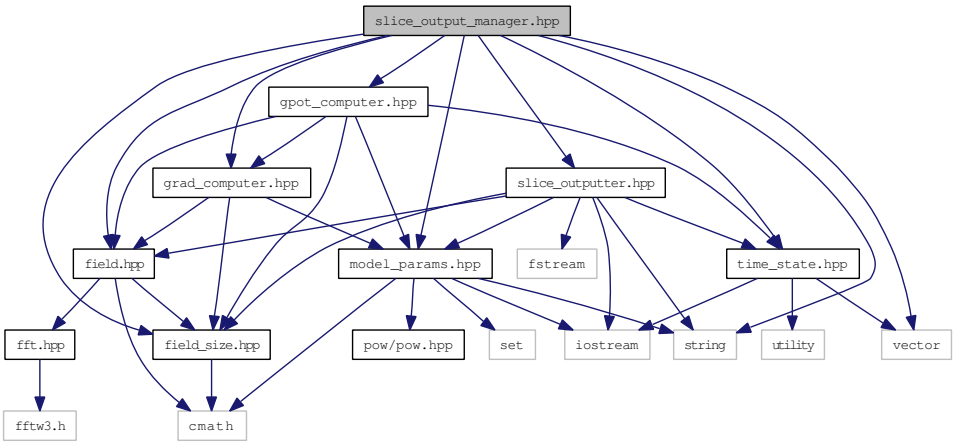
Fourth-order Runge–Kutta (RK4) integrator.

# 16.17 slice_output_manager.hpp File Reference

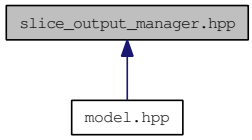
Field slice output manager. `#include "field_size.hpp"`

```
#include "model_params.hpp"
#include "time_state.hpp"
#include "field.hpp"
#include "slice_outputter.hpp"
#include "grad_computer.hpp"
#include "gpot_computer.hpp"
#include <string>
#include <vector>
```

Include dependency graph for slice_output_manager.hpp:




This graph shows which files directly or indirectly include this file:




## Classes

- class slice_output_manager< R >

### 16.17.1 Detailed Description

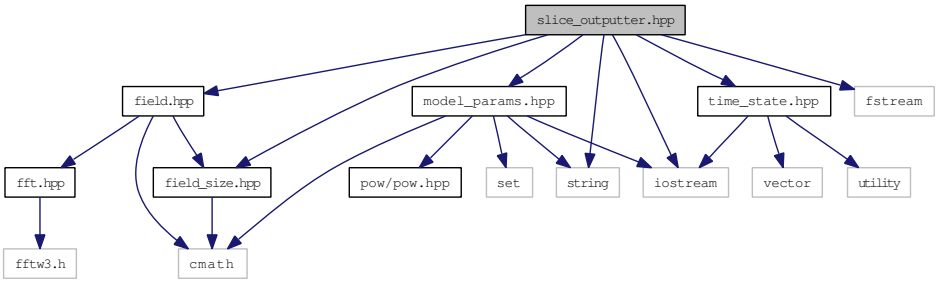
Field slice output manager.

## 16.18 slice_outputter.hpp File Reference

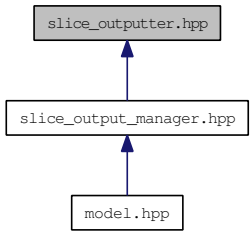
Outputter for the file slices. `#include "field_size.hpp"`

```
#include "model_params.hpp"
#include "time_state.hpp"
#include "field.hpp"
#include <string>
#include <iostream>
#include <fstream>
```

Include dependency graph for slice_outputter.hpp:




This graph shows which files directly or indirectly include this file:




### Classes

- class slice_outputter< R >

### 16.18.1 Detailed Description

Outputter for the file slices.

## 16.19 spectra_outputter.hpp File Reference

Outputter for the spectra TSV file. `#include "field.hpp"`

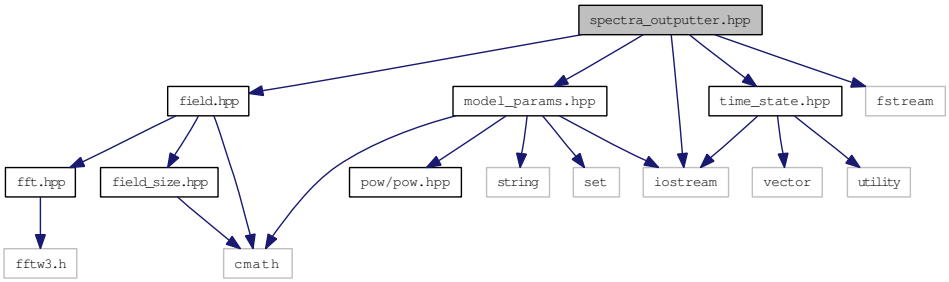
`#include "model_params.hpp"`

`#include "time_state.hpp"`

`#include <iostream>`

`#include <fstream>`

Include dependency graph for spectra_outputter.hpp:




### Classes

- class spectra_outputter< R >

### 16.19.1 Detailed Description

Outputter for the spectra TSV file.

# 16.20 stats_outputter.hpp File Reference

Outputter for the stats TSV file. `#include "field.hpp"`

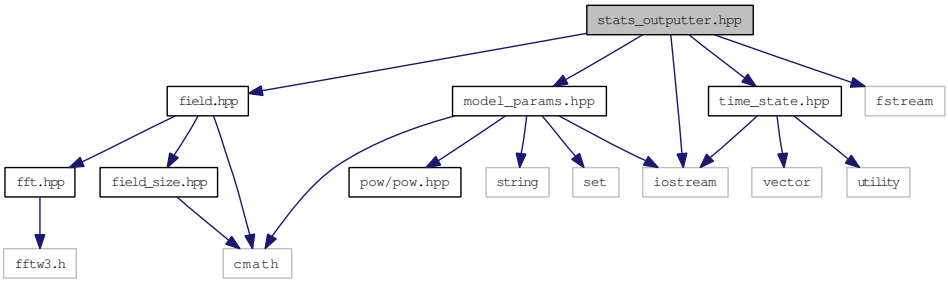
`#include "model_params.hpp"`

`#include "time_state.hpp"`

`#include <iostream>`

`#include <fstream>`

Include dependency graph for stats_outputter.hpp:




## Classes

- class stats_outputter< R >

### 16.20.1 Detailed Description

Outputter for the stats TSV file.

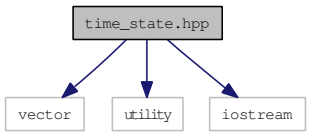
## 16.21 time_state.hpp File Reference

Time-varying model parameters. `#include <vector>`

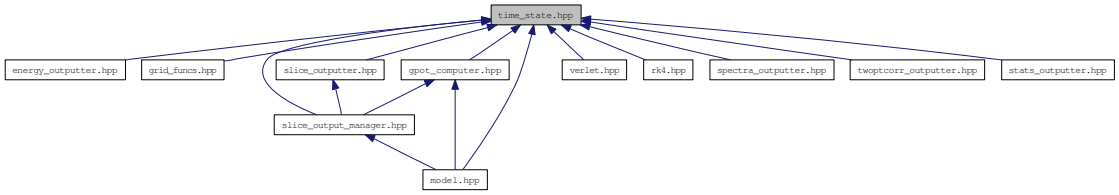
`#include <utility>`

`#include <iostream>`

Include dependency graph for time_state.hpp:




This graph shows which files directly or indirectly include this file:




### Classes

- struct time_state< R >

### 16.21.1 Detailed Description

Time-varying model parameters.

# 16.22 twoptcorr_outputter.hpp File Reference

Outputter for the twoptcorr TSV file. `#include "field.hpp"`

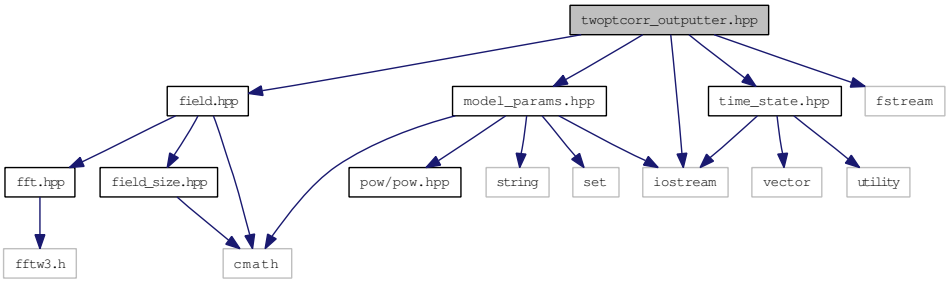
`#include "model_params.hpp"`

`#include "time_state.hpp"`

`#include <iostream>`

`#include <fstream>`

Include dependency graph for twoptcorr_outputter.hpp:




## Classes

- class twoptcorr_outputter< R >

### 16.22.1 Detailed Description

Outputter for the twoptcorr TSV file.

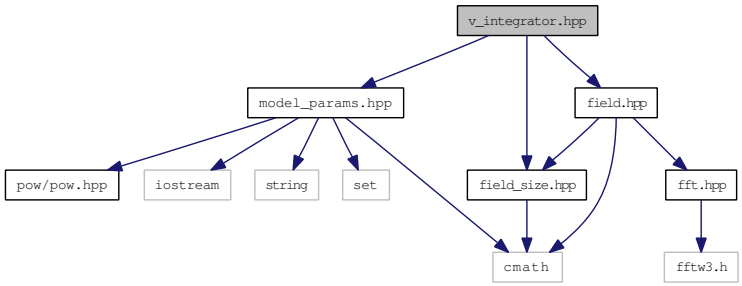
## 16.23 v_integrator.hpp File Reference

Integrate the potential energy over the field. `#include "field_size.hpp"`

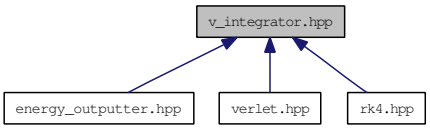
`#include "model_params.hpp"`

`#include "field.hpp"`

Include dependency graph for v_integrator.hpp:




This graph shows which files directly or indirectly include this file:




### Classes

- class v_integrator< R >

### 16.23.1 Detailed Description

Integrate the potential energy over the field.

# 16.24 verlet.hpp File Reference

Second-order Verlet integrator. `#include "field_size.hpp"`

`#include "model_params.hpp"`

`#include "time_state.hpp"`

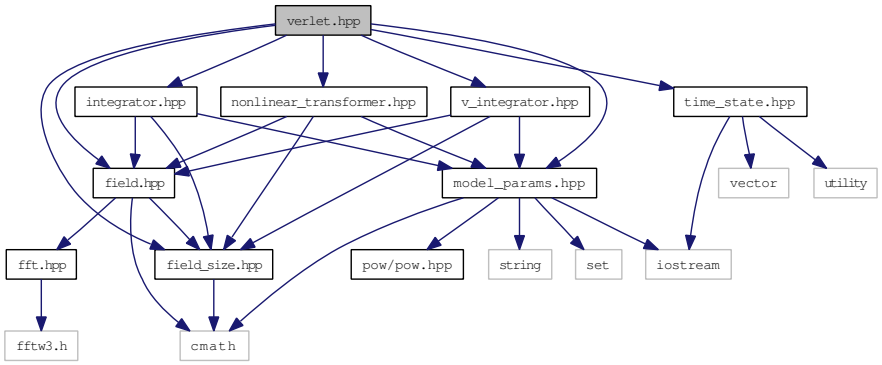
`#include "field.hpp"`

`#include "integrator.hpp"`

`#include "nonlinear_transformer.hpp"`

`#include "v_integrator.hpp"`

Include dependency graph for verlet.hpp:




## Classes

- class verlet< R >

### 16.24.1 Detailed Description

Second-order Verlet integrator.

# Index