fermiqcd

Generated by Doxygen 1.6.1

Wed Dec 23 14:03:13 2009

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 cleanup_cpp Namespace Reference

Functions

- def cleanup_cpp

5.1.1 Function Documentation

5.1.1.1 def cleanup_cpp::cleanup_cpp (*data*)

5.2 searchandreplace Namespace Reference

Variables

- tuple sin = raw_input('pattern to replace: ')
- tuple sout = raw_input('replace with: ')
- string choice = 'y'
- tuple file = open(filename,'r')
- string s = ''
- tuple line = line.replace(sin,sout)

5.2.1 Variable Documentation

5.2.1.1 string searchandreplace::choice = 'y'

5.2.1.2 tuple searchandreplace::file = open(filename,'r')

5.2.1.3 tuple searchandreplace::line = line.replace(sin,sout)

5.2.1.4 searchandreplace::s = ''

5.2.1.5 tuple searchandreplace::sin = raw_input('pattern to replace: ')

5.2.1.6 tuple searchandreplace::sout = raw_input('replace with: ')

5.3 searchandreplace2 Namespace Reference

Variables

- string choice = 'y'
- tuple file = open(filename,'r')
- string s = ''
- list c = line[4:5]
- list line2 = line[:4]
- line = line2

5.3.1 Variable Documentation

5.3.1.1 tuple searchandreplace2::c = line[4:5]

5.3.1.2 string searchandreplace2::choice = 'y'

5.3.1.3 tuple searchandreplace2::file = open(filename,'r')

5.3.1.4 searchandreplace2::line = line2

5.3.1.5 list searchandreplace2::line2 = line[:4]

5.3.1.6 searchandreplace2::s = ''

Chapter 6

Class Documentation

6.1 _sse_double Struct Reference

`#include <fermiqcd_sse.h>`

Public Attributes

- double c1
- double c2

6.1.1 Member Data Documentation

6.1.1.1 double _sse_double::c1

6.1.1.2 double _sse_double::c2

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse.h

6.2 _sse_float Struct Reference

`#include <fermiqcd_sse.h>`

Public Attributes

- float c1
- float c2
- float c3
- float c4

6.2.1 Member Data Documentation

6.2.1.1 float _sse_float::c1

6.2.1.2 float _sse_float::c2

6.2.1.3 float _sse_float::c3

6.2.1.4 float _sse_float::c4

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse.h

6.3 _sse_int Struct Reference

`#include <fermiqcd_sse.h>`

Public Attributes

- int c1
- int c2
- int c3
- int c4

6.3.1 Member Data Documentation

6.3.1.1 int _sse_int::c1

6.3.1.2 int _sse_int::c2

6.3.1.3 int _sse_int::c3

6.3.1.4 int _sse_int::c4

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse.h

6.4 _sse_spinor Struct Reference

`#include <fermiqcd_sse.h>`

Public Attributes

- _sse_su3_vector c1
- _sse_su3_vector c2
- _sse_su3_vector c3
- _sse_su3_vector c4

6.4.1 Member Data Documentation

6.4.1.1 _sse_su3_vector _sse_spinor::c1

6.4.1.2 _sse_su3_vector _sse_spinor::c2

6.4.1.3 _sse_su3_vector _sse_spinor::c3

6.4.1.4 _sse_su3_vector _sse_spinor::c4

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse.h

6.5 _sse_su3 Struct Reference

`#include <fermiqcd_sse.h>`

Public Attributes

- mdp_complex c11
- mdp_complex c12
- mdp_complex c13
- mdp_complex c21
- mdp_complex c22
- mdp_complex c23
- mdp_complex c31
- mdp_complex c32
- mdp_complex c33

6.5.1 Member Data Documentation

6.5.1.1 mdp_complex _sse_su3::c11

6.5.1.2 mdp_complex _sse_su3::c12

6.5.1.3 mdp_complex _sse_su3::c13

6.5.1.4 mdp_complex _sse_su3::c21

6.5.1.5 mdp_complex _sse_su3::c22

6.5.1.6 mdp_complex _sse_su3::c23

6.5.1.7 mdp_complex _sse_su3::c31

6.5.1.8 mdp_complex _sse_su3::c32

6.5.1.9 mdp_complex _sse_su3::c33

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse.h

6.6 _sse_su3_vector Struct Reference

`#include <fermiqcd_sse.h>`

Public Attributes

- mdp_complex c1
- mdp_complex c2
- mdp_complex c3

6.6.1 Member Data Documentation

6.6.1.1 mdp_complex _sse_su3_vector::c1

6.6.1.2 mdp_complex _sse_su3_vector::c2

6.6.1.3 mdp_complex _sse_su3_vector::c3

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse.h

6.7 _sse_vector Struct Reference

`#include <fermiqcd_sse.h>`

Public Attributes

- _sse_float c1
- _sse_float c2
- _sse_float c3

6.7.1 Member Data Documentation

6.7.1.1 _sse_float _sse_vector::c1

6.7.1.2 _sse_float _sse_vector::c2

6.7.1.3 _sse_float _sse_vector::c3

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse.h

6.8 ApeSmearing Class Reference

`#include <fermiqcd_topological_charge.h>`

Static Public Member Functions

- static void smear (gauge_field &U, mdp_real alpha=0.7, int iterations=20, int cooling_steps=10)

6.8.1 Member Function Documentation

6.8.1.1 static void ApeSmearing::smear (gauge_field & *U*, mdp_real *alpha* = `0.7`, int *iterations* = `20`, int *cooling_steps* = `10`) `[inline, static]`

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_topological_charge.h

6.9 BiCGStab Class Reference

the stabilized biconjugate inverter

`#include <fermiqcd_bicgstab_inverter.h>`

Static Public Member Functions

- template<class fieldT , class fieldG >
  static inversion_stats inverter (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_precision=0, int max_steps=2000)

6.9.1 Detailed Description

the stabilized biconjugate inverter It inverts mul_Q(psi_out,psi_in,U,coeff) iteratively

Parameters:

psi_out the output field passed by reference

psi_in the input field passed by reference

U the gauge field to be passed to mul_Q

coeff the gauge parameters to be passed to mul_Q

absolute_precision the target absolute precision

relative_precision the target relative precision

max_steps the maximum number of steps

Example:

```
/// gauge_field U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// fermi_field chi(lattice,nc);
/// coefficinets coeff;
/// coeff["kappa"]=1.12;
/// U.load("myfield");
/// psi.load("myfield_psi");
/// default_fermi_inverter=BiCGStab::inverter<fermi_field,gauge_field>;
/// default_fermi_action=FermiCloverActionSlow::mul_Q;
/// mul_invQ(chi,psi,U,coeff);
/// chi.save("myfield_chi");
///
```

Note that mul_invQ(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)^{-1}\psi$

6.9.2 Member Function Documentation

6.9.2.1 template<class fieldT , class fieldG > static inversion_stats BiCGStab::inverter (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_bicgstab_inverter.h

6.10 BiCGStabVtk Class Reference

the stabilized biconjugate inverter

`#include <fermiqcd_bicgstab_inverter_vtk.h>`

Static Public Member Functions

- template<class fieldT , class fieldG >
  static inversion_stats inverter (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_precision=0, int max_steps=2000)

6.10.1 Detailed Description

the stabilized biconjugate inverter It inverts mul_Q(psi_out,psi_in,U,coeff) iteratively

Parameters:

psi_out the output field passed by reference

psi_in the input field passed by reference

U the gauge field to be passed to mul_Q

coeff the gauge parameters to be passed to mul_Q

absolute_precision the target absolute precision

relative_precision the target relative precision

max_steps the maximum number of steps

Example:

```
/// gauge_field U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// fermi_field chi(lattice,nc);
/// coefficinets coeff;
/// coeff["kappa"]=1.12;
/// U.load("myfield");
/// psi.load("myfield_psi");
/// default_fermi_inverter=BiCGStab::inverter<fermi_field,gauge_field>;
/// default_fermi_action=FermiCloverActionSlow::mul_Q;
/// mul_invQ(chi,psi,U,coeff);
/// chi.save("myfield_chi");
///
```

Note that mul_invQ(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)^{-1}\psi$

6.10.2 Member Function Documentation

6.10.2.1 template<class fieldT , class fieldG > static inversion_stats BiCGStabVtk::inverter (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_bicgstab_inverter_vtk.h

6.11 CG2 Class Reference

the conjugate gradient inverter

`#include <fermiqcd_cg_inverter.h>`

Static Public Member Functions

- template<class fieldT , class fieldG >
  static inversion_stats inverter (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_precision=0, int max_steps=2000, bool qdaggerq=false)

6.11.1 Detailed Description

the conjugate gradient inverter It inverts mul_Q(psi_out,psi_in,U,coeff) ///not really

Parameters:

psi_out the output field passed by reference

psi_in the input field passed by reference

U the gauge field to be passed to mul_Q

coeff the gauge parameters to be passed to mul_Q

absolute_precision the target absolute precision

relative_precision the target relative precision

max_steps the maximum number of steps

Example:

```
/// gauge_field U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// fermi_field chi(lattice,nc);
/// coefficinets coeff;
/// coeff["kappa"]=1.12;
/// U.load("myfield");
/// psi.load("myfield_psi");
/// CG2::inverter(chi,psi,U,coeff);
/// chi.save("myfield_chi");
///
```

Note that mul_invQ(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)^{-1}\psi$

6.11.2 Member Function Documentation

6.11.2.1 template<class fieldT , class fieldG > static inversion_stats CG2::inverter (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000, bool *qdaggerq* = false) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_cg_inverter.h

6.12 coefficients Class Reference

container for action parameters

`#include <fermiqcd_coefficients.h>`

Public Member Functions

- bool has_key (const string s) const

6.12.1 Detailed Description

container for action parameters All FermiQCD actions are classe and share the same prototype. Parameters are passed to the action via coefficients objects which are nothing more than hash tables.

Example:

```
///    gauge_field U(lattice,nc);
///    coefficients gauge;
///    gauge["beta"]=6.0;
///    WilsonGaugeAction::heatbath(U,gauge);
///
```

Please check the spalling of the variables you store into the coefficients object (each action has its own coefficients).

Why? This allows the creating of new actions while reusing inverters and simplify passing parameters to the action.

6.12.2 Member Function Documentation

6.12.2.1 bool coefficients::has_key (const string *s*) const [inline]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_coefficients.h

6.13 dwfermi_field Class Reference

domain wall fermionic field

`#include <fermiqcd_dwfermi_field.h>`Inheritance diagram for dwfermi_field::

Public Member Functions

- dwfermi_field ()
- dwfermi_field (mdp_lattice &a, int L5_, int nc_, int nspin_=4)
- dwfermi_field (const dwfermi_field &psi)
- void allocate_dwfermi_field (mdp_lattice &a, int L5_, int nc_, int nspin_=4)
- mdp_matrix operator() (site x, int L5_)
- mdp_matrix operator() (site x, int L5_, int a)
- mdp_complex & operator() (site x, int L5_, int a, int i)
- const mdp_complex & operator() (site x, int L5_, int a, int i) const
- void operator= (mdp_complex a)

Public Attributes

- int nspin
- int nc
- int L5

6.13.1 Detailed Description

domain wall fermionic field Example:

```
/// int L5=10; // size in 5th dimension
/// fermi_field psi(lattice,L5,nc);
/// mdp_site x(lattice);
/// forallsites(x)
///    for(int k=0; k<L5; k++)
///      for(int spin=0; spin<4; spin++)
///        for(int i=0; i<nc; i++)
///          psi(x,k,spin,i)=0.0+0.0*I;
///
```

6.13.2 Constructor & Destructor Documentation

6.13.2.1 dwfermi_field::dwfermi_field () [inline]

6.13.2.2 dwfermi_field::dwfermi_field (mdp_lattice & *a*, int *L5_*, int *nc_*, int *nspin_* = 4) [inline]

6.13.2.3 dwfermi_field::dwfermi_field (const dwfermi_field & *psi*) [inline]

6.13.3 Member Function Documentation

6.13.3.1 void dwfermi_field::allocate_dwfermi_field (mdp_lattice & *a*, int *L5_*, int *nc_*, int *nspin_* = 4) [inline]

6.13.3.2 const mdp_complex& dwfermi_field::operator() (site *x*, int *L5_*, int *a*, int *i*) const [inline]

6.13.3.3 mdp_complex& dwfermi_field::operator() (site *x*, int *L5_*, int *a*, int *i*) [inline]

6.13.3.4 mdp_matrix dwfermi_field::operator() (site *x*, int *L5_*, int *a*) [inline]

6.13.3.5 mdp_matrix dwfermi_field::operator() (site *x*, int *L5_*) [inline]

6.13.3.6 void dwfermi_field::operator= (mdp_complex *a*) [inline]

Reimplemented from mdp_field< mdp_complex >.

6.13.4 Member Data Documentation

6.13.4.1 int dwfermi_field::L5

6.13.4.2 int dwfermi_field::nc

6.13.4.3 int dwfermi_field::nspin

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_dwfermi_field.h

6.14 DWFermiActionFast Class Reference

domain wall action fast

`#include <fermiqcd_dwfermi_actions.h>`

Static Public Member Functions

- static void mul_Q (dwfermi_field &psi_out, dwfermi_field &psi_in, gauge_field &U, coefficients &coeff)

6.14.1 Detailed Description

domain wall action fast Notation from ref. hep-lat/0007038 Example:

```
/// gauge_field U(lattice,nc);
/// dwfermi_field psi(lattice,nc);
/// dwfermi_field chi(lattice,nc);
/// coefficients coeff;
/// coeff["m_f"]=0.11; // fermion mass
/// coeff["m_5"]=0.11; // mass in 5th dimension
/// default_dwfermi_action=DWFermiActionFast::mul_Q;
/// mul_Q(chi,psi,U,coeff);
///
```

Note that mul_Q(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)\psi$

6.14.2 Member Function Documentation

6.14.2.1 static void DWFermiActionFast::mul_Q (dwfermi_field & *psi_out*, dwfermi_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_dwfermi_actions.h

6.15 DWFermiActionSlow Class Reference

domain wall action (SORRY THIS IS SLOW)

`#include <fermiqcd_dwfermi_actions.h>`

Static Public Member Functions

- static void mul_Q (dwfermi_field &psi_out, dwfermi_field &psi_in, gauge_field &U, coefficients &coeff)

6.15.1 Detailed Description

domain wall action (SORRY THIS IS SLOW) Notation from ref. hep-lat/0007038 Example:

```
/// gauge_field U(lattice,nc);
/// dwfermi_field psi(lattice,nc);
/// dwfermi_field chi(lattice,nc);
/// coefficients coeff;
/// coeff["m_f"]=0.11; // fermion mass
/// coeff["m_5"]=0.11; // mass in 5th dimension
/// default_dwfermi_action=DWFermiActionSlow::mul_Q;
/// mul_Q(chi,psi,U,coeff);
///
```

Note that mul_Q(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)\psi$

6.15.2 Member Function Documentation

6.15.2.1 static void DWFermiActionSlow::mul_Q (dwfermi_field & *psi_out*, dwfermi_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_dwfermi_actions.h

6.16 em_field Class Reference

the chromo-electr-magnetic field for any SU(n)

`#include <fermiqcd_gauge_field.h>`Inheritance diagram for em_field::

Public Member Functions

- em_field ()
- em_field (mdp_lattice &a, int nc_)
- em_field (em_field &em)
- void allocate_em_field (mdp_lattice &a, int nc_)
- int ordered_index (int mu, int nu) const
- mdp_matrix operator() (site x, int mu, int nu)
- mdp_complex & operator() (site x, int mu, int nu, int i, int j)
- const mdp_complex & operator() (site x, int mu, int nu, int i, int j) const

Public Attributes

- int ndim
- int nc
- int nem

6.16.1 Detailed Description

the chromo-electr-magnetic field for any SU(n) Example:

```
///    int nc=3;
///    int box[]={10,8,8,8};
///    mdp_lattice lattice(4,box);
///    gauge_field U(lattice,nc);
///    mdp_site x(lattice);
///    U.load("myfield");
///    compute_em_field(U);
///    forallsites(x)
///      for(int mu=0; mu<U.ndim; mu++)
///        for(int nu=mu+1; nu<U.ndim; nu++)
///          cout << U.em(x,mu,nu) << endl;
///
```

Note that U.em(x,mu,nu) is $a^2 G_{\mu\nu}$ and it is a color matrix in SU(nc). a is the lattice spacing.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 em_field::em_field () [inline]

6.16.2.2 em_field::em_field (mdp_lattice & *a*, int *nc_*) [inline]

6.16.2.3 em_field::em_field (em_field & *em*) [inline]

6.16.3 Member Function Documentation

6.16.3.1 void em_field::allocate_em_field (mdp_lattice & *a*, int *nc_*) [inline]

6.16.3.2 const mdp_complex& em_field::operator() (site *x*, int *mu*, int *nu*, int *i*, int *j*) const [inline]

6.16.3.3 mdp_complex& em_field::operator() (site *x*, int *mu*, int *nu*, int *i*, int *j*) [inline]

6.16.3.4 mdp_matrix em_field::operator() (site *x*, int *mu*, int *nu*) [inline]

6.16.3.5 int em_field::ordered_index (int *mu*, int *nu*) const [inline]

6.16.4 Member Data Documentation

6.16.4.1 int em_field::nc

6.16.4.2 int em_field::ndim

6.16.4.3 int em_field::nem

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_field.h

6.17 fermi_field Class Reference

wilson fermionic field

`#include <fermiqcd_fermi_field.h>`Inheritance diagram for fermi_field::

Public Member Functions

- fermi_field ()
- fermi_field (mdp_lattice &a, int nc_, int nspin_=4)
- void allocate_fermi_field (mdp_lattice &a, int nc_, int nspin_=4)
- fermi_field (const fermi_field &chi)
- void operator= (const fermi_field &chi)
- mdp_matrix operator() (site x)
- mdp_matrix operator() (site x, int a)
- mdp_complex & operator() (site x, int a, int i)
- const mdp_complex & operator() (site x, int a, int i) const
- void operator= (mdp_complex a)

Public Attributes

- int nspin
- int nc

6.17.1 Detailed Description

wilson fermionic field Example:

```
/// fermi_field psi(lattice,nc);
/// mdp_site x(lattice);
/// forallsites(x)
///    for(int spin=0; spin<4; spin++)
///      for(int i=0; i<nc; i++)
///        psi(x,spin,i)=0.0+0.0*I;
///
```

6.17.2 Constructor & Destructor Documentation

6.17.2.1 fermi_field::fermi_field () [inline]

6.17.2.2 fermi_field::fermi_field (mdp_lattice & *a*, int *nc_*, int *nspin_* = 4) [inline]

6.17.2.3 fermi_field::fermi_field (const fermi_field & *chi*) [inline]

6.17.3 Member Function Documentation

6.17.3.1 void fermi_field::allocate_fermi_field (mdp_lattice & *a*, int *nc_*, int *nspin_* = 4) [inline]

6.17.3.2 const mdp_complex& fermi_field::operator() (site *x*, int *a*, int *i*) const [inline]

6.17.3.3 mdp_complex& fermi_field::operator() (site *x*, int *a*, int *i*) [inline]

6.17.3.4 mdp_matrix fermi_field::operator() (site *x*, int *a*) [inline]

6.17.3.5 mdp_matrix fermi_field::operator() (site *x*) [inline]

6.17.3.6 void fermi_field::operator= (mdp_complex *a*) [inline]

Reimplemented from mdp_field< mdp_complex >.

6.17.3.7 void fermi_field::operator= (const fermi_field & *chi*) [inline]

Reimplemented from mdp_complex_field.

6.17.4 Member Data Documentation

6.17.4.1 int fermi_field::nc

6.17.4.2 int fermi_field::nspin

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_fermi_field.h

6.18 fermi_propagator Class Reference

a Wilson/Clover quark propagator (all 12 components)

`#include <fermiqcd_fermi_propagator.h>`Inheritance diagram for fermi_propagator::

Public Member Functions

- fermi_propagator ()
- fermi_propagator (mdp_lattice &mylattice, int nc_, int nspin_=4)
- void allocate_fermi_propagator (mdp_lattice &mylattice, int nc_, int nspin_=4)
- mdp_matrix operator() (site x, int a, int b)
- mdp_complex & operator() (site x, int a, int b, int i, int j)

Public Attributes

- int nspin
- int nc

Friends

- void generate (fermi_propagator &S, gauge_field &U, coefficients &coeff, mdp_real absolute_precision=fermi_inversion_precision, mdp_real relative_precision=0, int max_steps=2000, void(*smf)(fermi_field &, gauge_field &, coefficients &)=0, coefficients smear_coeff=coefficients(), int comp=0)

6.18.1 Detailed Description

a Wilson/Clover quark propagator (all 12 components) Example of how to make a pion:

```
/// gauge_field U(lattice,nc);
/// U.load("myfield");
/// fermi_propagator S(lattice,nc);
/// coefficients quark;
/// quark["kappa"]=1.12;
/// generate(S,U,quark);
/// vector<float> sum(U.lattice.size(TIME));
/// forallsites(x)
///   for(int alpha=0; alpha<4; alpha++)
///     for(int beta=0; beta<4; beta++)
///        sum(x(0))+=real(trace(S(x,alpha,beta)*
///                   hermitian(S(x,beta,alpha))));
///
```

Note that S(x,alpha,beta,i,j) is $\left\langle 0|\bar{q}^i_\alpha(x), q^j_\beta(0)|\right\rangle$

6.18.2 Constructor & Destructor Documentation

6.18.2.1 fermi_propagator::fermi_propagator () [inline]

6.18.2.2 fermi_propagator::fermi_propagator (mdp_lattice & *mylattice*, int *nc_*, int *nspin_* = 4) [inline]

6.18.3 Member Function Documentation

6.18.3.1 void fermi_propagator::allocate_fermi_propagator (mdp_lattice & *mylattice*, int *nc_*, int *nspin_* = 4) [inline]

6.18.3.2 mdp_complex& fermi_propagator::operator() (site *x*, int *a*, int *b*, int *i*, int *j*) [inline]

6.18.3.3 mdp_matrix fermi_propagator::operator() (site *x*, int *a*, int *b*) [inline]

6.18.4 Friends And Related Function Documentation

6.18.4.1 void generate (fermi_propagator & *S*, gauge_field & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = fermi_inversion_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000, void(∗)(fermi_field &, gauge_field &, coefficients &) *smf* = 0, coefficients *smear_coeff* = coefficients(), int *comp* = 0) [friend]

makes the quark propagator

Parameters:

S the output propagator

U the input gauge configuration

coeff the parameters to be passed to the action

absolute_precision the target absolute precision for inversion

relative_precision the target relative precision for invcersion

max_steps the max number of steps in inversion

smf pointer to smearing function (smear sources)

smear_coeff parameters for smearing

6.18.5 Member Data Documentation

6.18.5.1 int fermi_propagator::nc

6.18.5.2 int fermi_propagator::nspin

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_fermi_propagator.h

6.19 FermiCloverActionFast Class Reference

Wilson/Clover action.

`#include <fermiqcd_fermi_actions.h>`

Static Public Member Functions

- static void mul_Q (fermi_field &psi_out, fermi_field &psi_in, gauge_field &U, coefficients &coeff, int parity=EVENODD)

6.19.1 Detailed Description

Wilson/Clover action. Example:

```
/// gauge_field U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// fermi_field chi(lattice,nc);
/// coefficients coeff;
/// coeff["kappa_s"]=0.11;
/// coeff["kappa_t"]=0.11;
/// coeff["r_s"]=1.0;
/// coeff["r_t"]=1.0;
/// coeff["c_{sw}"]=1.0;
/// coeff["c_E"]=1.0;
/// coeff["c_B"]=1.0;
/// default_fermi_action=FermiCloverActionFast::mul_Q;
/// if(coeff["c_{sw}"]!=0) compute_em_field(U);
/// mul_Q(chi,psi,U,coeff);
///
```

6.19.2 Member Function Documentation

6.19.2.1 static void FermiCloverActionFast::mul_Q (fermi_field & *psi_out*, fermi_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, int *parity* = EVENODD) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_fermi_actions.h

6.20 FermiCloverActionSlow Class Reference

Wilson/Clover action (SLOW: DO NOT USE IN PRODUCTION).

`#include <fermiqcd_fermi_actions.h>`

Static Public Member Functions

- static void mul_Q (fermi_field &psi_out, fermi_field &psi_in, gauge_field &U, coefficients &coeff, int parity=EVENODD)

6.20.1 Detailed Description

Wilson/Clover action (SLOW: DO NOT USE IN PRODUCTION). Example:

```
/// gauge_field U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// fermi_field chi(lattice,nc);
/// coefficients coeff;
/// coeff["kappa_s"]=0.11;
/// coeff["kappa_t"]=0.11;
/// coeff["r_s"]=1.0;
/// coeff["r_t"]=1.0;
/// coeff["c_{sw}"]=1.0;
/// coeff["c_E"]=1.0;
/// coeff["c_B"]=1.0;
/// default_fermi_action=FermiCloverActionSlow::mul_Q;
/// if(coeff["c_{sw}"]!=0) compute_em_field(U);
/// mul_Q(chi,psi,U,coeff);
///
```

Note that mul_Q(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)\psi$

6.20.2 Member Function Documentation

6.20.2.1 static void FermiCloverActionSlow::mul_Q (fermi_field & *psi_out*, fermi_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, int *parity* = EVENODD) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_fermi_actions.h

6.21 gauge_field Class Reference

the gauge field for any SU(n)

`#include <fermiqcd_gauge_field.h>`Inheritance diagram for gauge_field::

Public Member Functions

- gauge_field ()
- gauge_field (const gauge_field &U)
- void operator= (const gauge_field &U)
- gauge_field (mdp_lattice &a, int nc_)
- void allocate_gauge_field (mdp_lattice &a, int nc_)
- mdp_matrix operator() (site x, int mu)
- const mdp_matrix operator() (site x, int mu) const
- mdp_complex & operator() (site x, int mu, int i, int j)
- const mdp_complex & operator() (site x, int mu, int i, int j) const
- mdp_matrix operator() (site x, int sign, int mu)
- const mdp_matrix operator() (site x, int sign, int mu) const
- const mdp_complex operator() (site x, int sign, int mu, int i, int j) const

Public Attributes

- em_field em
- mdp_nmatrix_field long_links
- mdp_field< mdp_int > i_jump
- mdp_matrix_field swirls
- int ndim
- int nc

6.21.1 Detailed Description

the gauge field for any SU(n) Example:

```
///    int nc=3;
///    int box[]={10,8,8,8};
///    mdp_lattice lattice(4,box);
///    gauge_field U(lattice,nc);
///    mdp_site x(lattice);
///    // set_cold(U);
///    forallsites(x)
///       for(int mu=0; mu<U.ndim; mu++)
///          U(x,mu)=1;
///    U.update(); // synchronization
///    U.save("myfield");
///    U.load("myfield");
///
```

Note that U(x,mu) is $\exp iaA_\mu$ and it is a color matrix in SU(nc). a is the lattice spacing.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 gauge_field::gauge_field () [inline]

6.21.2.2 gauge_field::gauge_field (const gauge_field & *U*) [inline]

6.21.2.3 gauge_field::gauge_field (mdp_lattice & *a*, int *nc_*) [inline]

6.21.3 Member Function Documentation

6.21.3.1 void gauge_field::allocate_gauge_field (mdp_lattice & *a*, int *nc_*) [inline]

6.21.3.2 const mdp_complex gauge_field::operator() (site *x*, int *sign*, int *mu*, int *i*, int *j*) const [inline]

6.21.3.3 const mdp_matrix gauge_field::operator() (site *x*, int *sign*, int *mu*) const [inline]

6.21.3.4 mdp_matrix gauge_field::operator() (site *x*, int *sign*, int *mu*) [inline]

6.21.3.5 const mdp_complex& gauge_field::operator() (site *x*, int *mu*, int *i*, int *j*) const [inline]

6.21.3.6 mdp_complex& gauge_field::operator() (site *x*, int *mu*, int *i*, int *j*) [inline]

6.21.3.7 const mdp_matrix gauge_field::operator() (site *x*, int *mu*) const [inline]

6.21.3.8 mdp_matrix gauge_field::operator() (site *x*, int *mu*) [inline]

6.21.3.9 void gauge_field::operator= (const gauge_field & *U*) [inline]

Reimplemented from mdp_complex_field.

6.21.4 Member Data Documentation

6.21.4.1 em_field gauge_field::em

6.21.4.2 mdp_field<mdp_int> gauge_field::i_jump

6.21.4.3 mdp_nmatrix_field gauge_field::long_links

6.21.4.4 int gauge_field::nc

6.21.4.5 int gauge_field::ndim

6.21.4.6 mdp_matrix_field gauge_field::swirls

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_field.h

6.22 gauge_stats Class Reference

(unused)

`#include <fermiqcd_gauge_actions.h>`

6.22.1 Detailed Description

(unused)

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_actions.h

6.23 GaugeFixing Class Reference

the main gaugefixing algorithm

`#include <fermiqcd_gauge_fixing.h>`

Static Public Member Functions

- static void hit (gauge_field &U, int mu, int parity, int i, int j, mdp_real overrelaxation_boost=1)
- static void z3_fix (gauge_field &U, int mu)
- static gaugefixing_stats fix (gauge_field &U, int mu=0, int max_steps=1, mdp_real target_-precision=1e-5, mdp_real overrelaxation_boost=1, bool z3=false)

Static Public Attributes

- static const int Coulomb = 0
- static const int Landau = 10

6.23.1 Detailed Description

the main gaugefixing algorithm Example:

```
///    gauge_field U(lattice,nc);
///    gaugefixing_stats stats;
///    U.load("myfield");
///    stats=GaugeFixing::fix(U,GaugeFixing::Coulomb,100);
///    U.save("myfield_gaugefixed");
///
```

6.23.2 Member Function Documentation

6.23.2.1 static gaugefixing_stats GaugeFixing::fix (gauge_field & *U*, int *mu* = `0`, int *max_steps* = `1`, mdp_real *target_precision* = `1e-5`, mdp_real *overrelaxation_boost* = `1`, bool *z3* = `false`) `[inline, static]`

performs the gauge fixing

Parameters:

U the gauge field

mu = GaugeFixing::Coulomb or GaugeFixing::Landau or other direction

max_steps maximum number of gaugefixing steps

parget_precision precision in gaugefixing

overrelaxation_boost

z3 if set to true fixes residual Z(n) symmatry due to lattice torus topology

6.23.2.2 static void GaugeFixing::hit (gauge_field & *U*, int *mu*, int *parity*, int *i*, int *j*, mdp_real *overrelaxation_boost* = 1) [inline, static]

6.23.2.3 static void GaugeFixing::z3_fix (gauge_field & *U*, int *mu*) [inline, static]

6.23.3 Member Data Documentation

6.23.3.1 const int GaugeFixing::Coulomb = 0 [static]

6.23.3.2 const int GaugeFixing::Landau = 10 [static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_fixing.h

6.24 gaugefixing_stats Class Reference

Structure for gaugefixing stats.

`#include <fermiqcd_gauge_fixing.h>`

Public Attributes

- uint max_steps
- mdp_real target_precision
- uint steps
- mdp_real precision
- mdp_real action

6.24.1 Detailed Description

Structure for gaugefixing stats.

6.24.2 Member Data Documentation

6.24.2.1 mdp_real gaugefixing_stats::action

6.24.2.2 uint gaugefixing_stats::max_steps

6.24.2.3 mdp_real gaugefixing_stats::precision

6.24.2.4 uint gaugefixing_stats::steps

6.24.2.5 mdp_real gaugefixing_stats::target_precision

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_fixing.h

6.25 HMC< GaugeClass, FermiClass > Class Template Reference

`#include <fermiqcd_hmc.h>`

Public Member Functions

- HMC (GaugeClass &U, FermiClass &F, coefficients &coeff)
- ~HMC ()
- void step ()
- mdp_real acceptance_rate ()
- void initialize ()
- mdp_real compute_gaussian_momenta (GaugeClass &U)
- void set_gaussian (FermiClass &F)
- mdp_real compute_kinetic_energy (GaugeClass &p_U, FermiClass &p_F)
- void compute_effective_links (GaugeClass &U, GaugeClass &V)
- mdp_real compute_action (GaugeClass &U, GaugeClass &V, FermiClass &F)
- void compute_fields_evolution (GaugeClass &U, GaugeClass &p_U, GaugeClass &f_U, FermiClass &F, FermiClass &p_F, FermiClass &f_F)
- void compute_force (GaugeClass &U, GaugeClass &f_U, FermiClass &F, FermiClass &f_F)
- void compute_fermion_forces (GaugeClass &U, GaugeClass &f_U, FermiClass &sol, FermiClass &psol)

Static Public Member Functions

- static mdp_matrix spinor (FermiClass &psi, mdp_site x, int b)

Public Attributes

- coefficients coeff
- double bs
- double bs_old
- double fs
- double fs_old
- mdp_real s_old
- int dimrep
- int accepted
- int steps
- vector< mdp_matrix > S
- vector< mdp_matrix > lambda

Static Public Attributes

- static const int FUNDAMENTAL = 0
- static const int SYMMETRIC = 1
- static const int ANTISYMMETRIC = 2

template<class GaugeClass, class FermiClass> class HMC< GaugeClass, FermiClass >

6.25.1 Constructor & Destructor Documentation

6.25.1.1 **template<class GaugeClass , class FermiClass > HMC< GaugeClass, FermiClass >::HMC (GaugeClass & *U*, FermiClass & *F*, coefficients & *coeff*) `[inline]`**

6.25.1.2 **template<class GaugeClass , class FermiClass > HMC< GaugeClass, FermiClass >::~HMC () `[inline]`**

6.25.2 Member Function Documentation

6.25.2.1 **template<class GaugeClass , class FermiClass > mdp_real HMC< GaugeClass, FermiClass >::acceptance_rate () `[inline]`**

6.25.2.2 **template<class GaugeClass , class FermiClass > mdp_real HMC< GaugeClass, FermiClass >::compute_action (GaugeClass & *U*, GaugeClass & *V*, FermiClass & *F*) `[inline]`**

6.25.2.3 **template<class GaugeClass , class FermiClass > void HMC< GaugeClass, FermiClass >::compute_effective_links (GaugeClass & *U*, GaugeClass & *V*) `[inline]`**

6.25.2.4 **template<class GaugeClass , class FermiClass > void HMC< GaugeClass, FermiClass >::compute_fermion_forces (GaugeClass & *U*, GaugeClass & *f_U*, FermiClass & *sol*, FermiClass & *psol*) `[inline]`**

6.25.2.5 **template<class GaugeClass , class FermiClass > void HMC< GaugeClass, FermiClass >::compute_fields_evolution (GaugeClass & *U*, GaugeClass & *p_U*, GaugeClass & *f_U*, FermiClass & *F*, FermiClass & *p_F*, FermiClass & *f_F*) `[inline]`**

6.25.2.6 **template<class GaugeClass , class FermiClass > void HMC< GaugeClass, FermiClass >::compute_force (GaugeClass & *U*, GaugeClass & *f_U*, FermiClass & *F*, FermiClass & *f_F*) `[inline]`**

6.25.2.7 **template<class GaugeClass , class FermiClass > mdp_real HMC< GaugeClass, FermiClass >::compute_gaussian_momenta (GaugeClass & *U*) `[inline]`**

6.25.2.8 **template<class GaugeClass , class FermiClass > mdp_real HMC< GaugeClass, FermiClass >::compute_kinetic_energy (GaugeClass & *p_U*, FermiClass & *p_F*) `[inline]`**

6.25.2.9 **template<class GaugeClass , class FermiClass > void HMC< GaugeClass, FermiClass >::initialize () `[inline]`**

6.25.2.10 **template<class GaugeClass , class FermiClass > void HMC< GaugeClass, FermiClass >::set_gaussian (FermiClass & *F*) `[inline]`**

6.25.2.11 **template<class GaugeClass , class FermiClass > static mdp_matrix HMC< GaugeClass, FermiClass >::spinor (FermiClass & *psi*, mdp_site *x*, int *b*) `[inline, static]`**

6.25.2.12 **template<class GaugeClass , class FermiClass > void HMC< GaugeClass, FermiClass >::step () `[inline]`**

CHECKED UP TO HERE!

6.25.3 Member Data Documentation

6.25.3.1 template<class GaugeClass , class FermiClass > int HMC< GaugeClass, FermiClass >::accepted

6.25.3.2 template<class GaugeClass , class FermiClass > const int HMC< GaugeClass, FermiClass >::ANTISYMMETRIC = 2 [static]

6.25.3.3 template<class GaugeClass , class FermiClass > double HMC< GaugeClass, FermiClass >::bs

6.25.3.4 template<class GaugeClass , class FermiClass > double HMC< GaugeClass, FermiClass >::bs_old

6.25.3.5 template<class GaugeClass , class FermiClass > coefficients HMC< GaugeClass, FermiClass >::coeff

6.25.3.6 template<class GaugeClass , class FermiClass > int HMC< GaugeClass, FermiClass >::dimrep

6.25.3.7 template<class GaugeClass , class FermiClass > double HMC< GaugeClass, FermiClass >::fs

6.25.3.8 template<class GaugeClass , class FermiClass > double HMC< GaugeClass, FermiClass >::fs_old

6.25.3.9 template<class GaugeClass , class FermiClass > const int HMC< GaugeClass, FermiClass >::FUNDAMENTAL = 0 [static]

6.25.3.10 template<class GaugeClass , class FermiClass > vector<mdp_matrix> HMC< GaugeClass, FermiClass >::lambda

6.25.3.11 template<class GaugeClass , class FermiClass > vector<mdp_matrix> HMC< GaugeClass, FermiClass >::S

6.25.3.12 template<class GaugeClass , class FermiClass > mdp_real HMC< GaugeClass, FermiClass >::s_old

6.25.3.13 template<class GaugeClass , class FermiClass > int HMC< GaugeClass, FermiClass >::steps

6.25.3.14 template<class GaugeClass , class FermiClass > const int HMC< GaugeClass, FermiClass >::SYMMETRIC = 1 [static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_hmc.h

6.26 ImprovedGaugeAction Class Reference

the $O(a^2)$ Improved Gauge Action

`#include <fermiqcd_gauge_actions.h>`Inheritance diagram for ImprovedGaugeAction::

Static Public Member Functions

- static gauge_stats heatbath (gauge_field &U, coefficients &coeff, int n_iter=1, string model="MILC")

6.26.1 Detailed Description

the $O(a^2)$ Improved Gauge Action Example using the MILC improved action:

```
///    int ns=2, steps=10;
///    gauge_field U(lattice,nc);
///    coefficients gauge;
///    U.load("myfield.0000");
///    gauge["beta"]=6.0;
///    gauge["zeta"]=1.0; // MUST BE ONE
///    gauge["u_t"]=1.0;
///    gauge["u_s"]=1.0;
///    ImprovedGaugeAction::heatbath(U,gauge,steps,"MILC");
///    U.save("myfield.0001");
///
```

Example using the Morningstar unisotropic improved action:

```
///    int ns=2, steps=10;
///    gauge_field U(lattice,nc);
///    coefficients gauge;
///    U.load("myfield.0000");
///    gauge["beta"]=6.0;
///    gauge["zeta"]=1.0; // CAN BE != ONE
///    gauge["u_t"]=1.0;
///    gauge["u_s"]=1.0;
///    ImprovedGaugeAction::heatbath(U,gauge,steps,"Morningstar");
///    U.save("myfield.0001");
///
```

6.26.2 Member Function Documentation

6.26.2.1 static gauge_stats ImprovedGaugeAction::heatbath (gauge_field & *U*, coefficients & *coeff*, int *n_iter* = 1, string *model* = "MILC") [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_actions.h

6.27 ImprovedGaugeActionSSE2 Class Reference

the $O(a^2)$ Improved Gauge Action for SU3 with SSE2 and double precision (UNTESTED)

`#include <fermiqcd_gauge_actions_sse2.h>`Inheritance diagram for ImprovedGaugeActionSSE2::

Static Public Member Functions

- static gauge_stats heatbath (gauge_field &U, coefficients &coeff, int n_iter=1, string model="MILC")

6.27.1 Detailed Description

the $O(a^2)$ Improved Gauge Action for SU3 with SSE2 and double precision (UNTESTED) Example using the MILC improved action:

```
///    int ns=2, steps=10;
///    gauge_field U(lattice,nc);
///    coefficients gauge;
///    U.load("myfield.0000");
///    gauge["beta"]=6.0;
///    gauge["zeta"]=1.0; // MUST BE ONE
///    gauge["u_t"]=1.0;
///    gauge["u_s"]=1.0;
///    ImprovedGaugeActionSSE2::heatbath(U,gauge,steps,"MILC");
///    U.save("myfield.0001");
///
```

Example using the Morningstar unisotropic improved action:

```
///    int ns=2, steps=10;
///    gauge_field U(lattice,nc);
///    coefficients gauge;
///    U.load("myfield.0000");
///    gauge["beta"]=6.0;
///    gauge["zeta"]=1.0; // CAN BE != ONE
///    gauge["u_t"]=1.0;
///    gauge["u_s"]=1.0;
///    ImprovedGaugeActionSSE2::heatbath(U,gauge,steps,"Morningstar");
///    U.save("myfield.0001");
///
```

6.27.2 Member Function Documentation

6.27.2.1 static gauge_stats ImprovedGaugeActionSSE2::heatbath (gauge_field & *U*, coefficients & *coeff*, int *n_iter* = 1, string *model* = "MILC") [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_actions_sse2.h

6.28 Instanton4D Class Reference

`#include <fermiqcd_instanton4d.h>`

Public Member Functions

- Instanton4D (int nc, int sub_i, int sub_j, mdp_real charge, mdp_real lambda, vector< mdp_real > &p)
- mdp_matrix operator() (mdp_site &x, int mu)

Public Attributes

- vector< mdp_real > p
- int nc
- int sub_i
- int sub_j
- mdp_real charge
- mdp_real lambda
- mdp_matrix eta [4][4]

6.28.1 Constructor & Destructor Documentation

6.28.1.1 Instanton4D::Instanton4D (int *nc*, int *sub_i*, int *sub_j*, mdp_real *charge*, mdp_real *lambda*, vector< mdp_real > & *p*) [inline]

6.28.2 Member Function Documentation

6.28.2.1 mdp_matrix Instanton4D::operator() (mdp_site & *x*, int *mu*) [inline]

6.28.3 Member Data Documentation

6.28.3.1 mdp_real Instanton4D::charge

6.28.3.2 mdp_matrix Instanton4D::eta[4][4]

6.28.3.3 mdp_real Instanton4D::lambda

6.28.3.4 int Instanton4D::nc

6.28.3.5 vector<mdp_real> Instanton4D::p

6.28.3.6 int Instanton4D::sub_i

6.28.3.7 int Instanton4D::sub_j

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_instanton4d.h

6.29 inversion_stats Class Reference

structure for inverstion stats

`#include <fermiqcd_minres_inverter.h>`

Public Attributes

- mdp_real target_absolute_precision
- mdp_real target_relative_precision
- int max_steps
- mdp_real residue
- mdp_real absolute_precision
- mdp_real relative_precision
- int steps
- int mul_Q_steps
- mdp_real time

6.29.1 Detailed Description

structure for inverstion stats Returned by the inverters

6.29.2 Member Data Documentation

6.29.2.1 mdp_real inversion_stats::absolute_precision

6.29.2.2 int inversion_stats::max_steps

6.29.2.3 int inversion_stats::mul_Q_steps

6.29.2.4 mdp_real inversion_stats::relative_precision

6.29.2.5 mdp_real inversion_stats::residue

6.29.2.6 int inversion_stats::steps

6.29.2.7 mdp_real inversion_stats::target_absolute_precision

6.29.2.8 mdp_real inversion_stats::target_relative_precision

6.29.2.9 mdp_real inversion_stats::time

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_minres_inverter.h

6.30 Lanczos< fieldT > Class Template Reference

Lanczos algorithms.

`#include <fermiqcd_lanczos.h>`

Static Public Member Functions

- static mdp_complex step (fieldT &psi, gauge_field &U, coefficients &coeff, bool force=false, bool output_check=false)

6.30.1 Detailed Description

template<class fieldT> class Lanczos< fieldT >

Lanczos algorithms. Example:

```
/// mdp_gauge U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// coefficients coeff;
/// coeff["kappa"]=1.12;
/// for(int k=0; k<100; k++)
///    mdp << Lanczos::step(psi,U,coeff) << endl;
///
```

return mdp_complex(alpha,eta)

6.30.2 Member Function Documentation

6.30.2.1 template<class fieldT > static mdp_complex Lanczos< fieldT >::step (fieldT & *psi*, gauge_field & *U*, coefficients & *coeff*, bool *force* = `false`, bool *output_check* = `false`) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_lanczos.h

6.31 mdp_array< T, nc_ > Class Template Reference

generic container for multidimensional arrays

`#include <mdp_array.h>`

Public Member Functions

- const int & ndim () const
- T * address ()
- uint * size_address ()
- T & operator[ ] (const uint i)
- const T & operator[ ] (const uint i) const
- uint length (const uint i) const
- uint length () const
- uint size (uint i) const
- uint size () const
- void dimension (const uint *p)
- void dimension (const uint c0_=1, const uint c1_=1, const uint c2_=1, const uint c3_=1, const uint c4_=1)
- mdp_array (const uint c0_=1, const uint c1_=1, const uint c2_=1, const uint c3_=1, const uint c4_=1)
- mdp_array (const uint *p)
- mdp_array (const T *m0, const uint c0_=1, const uint c1_=1, const uint c2_=1, const uint c3_=1, const uint c4_=1)
- mdp_array (const T *m0, const uint *p)
- mdp_array (const mdp_array &a)
- virtual ~mdp_array ()
- void operator= (const mdp_array &a)
- T & operator() (const uint i0, const uint i1=0, const uint i2=0, const uint i3=0, const uint i4=0)
- const T & operator() (const uint i0, const uint i1=0, const uint i2=0, const uint i3=0, const uint i4=0) const

Friends

- void prepare (const mdp_array &a)
- mdp_array operator+ (const mdp_array &a, const mdp_array &b)
- mdp_array operator- (const mdp_array &a, const mdp_array &b)
- template<class T2 >
  mdp_array operator* (T2 x, const mdp_array &a)
- mdp_array applytoall (const mdp_array &a, T(*fptr)(T, void *), void *x=0)
- mdp_array applytoall (const mdp_array &a, const mdp_array &b, T(*fptr)(T, T, void *), void *x=0)
- ostream & operator<< (ostream &os, const mdp_array &a)

6.31.1 Detailed Description

template<class T, uint nc_> class mdp_array< T, nc_ >

generic container for multidimensional arrays Example:

```
///     mdp_array<float,3> a(5,5,5);
///     a(0,0,0)=3.15;
///
```

6.31.2 Constructor & Destructor Documentation

6.31.2.1 template<class T, uint nc_> mdp_array< T, nc_ >::mdp_array (const uint *c0_* = 1, const uint *c1_* = 1, const uint *c2_* = 1, const uint *c3_* = 1, const uint *c4_* = 1) [inline]

6.31.2.2 template<class T, uint nc_> mdp_array< T, nc_ >::mdp_array (const uint * *p*) [inline]

6.31.2.3 template<class T, uint nc_> mdp_array< T, nc_ >::mdp_array (const T * *m0*, const uint *c0_* = 1, const uint *c1_* = 1, const uint *c2_* = 1, const uint *c3_* = 1, const uint *c4_* = 1) [inline]

6.31.2.4 template<class T, uint nc_> mdp_array< T, nc_ >::mdp_array (const T * *m0*, const uint * *p*) [inline]

6.31.2.5 template<class T, uint nc_> mdp_array< T, nc_ >::mdp_array (const mdp_array< T, nc_ > & *a*) [inline]

6.31.2.6 template<class T, uint nc_> virtual mdp_array< T, nc_ >::~mdp_array () [inline, virtual]

6.31.3 Member Function Documentation

6.31.3.1 template<class T, uint nc_> T* mdp_array< T, nc_ >::address () [inline]

6.31.3.2 template<class T, uint nc_> void mdp_array< T, nc_ >::dimension (const uint *c0_* = 1, const uint *c1_* = 1, const uint *c2_* = 1, const uint *c3_* = 1, const uint *c4_* = 1) [inline]

6.31.3.3 template<class T, uint nc_> void mdp_array< T, nc_ >::dimension (const uint * *p*) [inline]

6.31.3.4 template<class T, uint nc_> uint mdp_array< T, nc_ >::length () const [inline]

6.31.3.5 template<class T, uint nc_> uint mdp_array< T, nc_ >::length (const uint *i*) const [inline]

6.31.3.6 template<class T, uint nc_> const int& mdp_array< T, nc_ >::ndim () const [inline]

6.31.3.7 template<class T, uint nc_> const T& mdp_array< T, nc_ >::operator() (const uint *i0*, const uint *i1* = 0, const uint *i2* = 0, const uint *i3* = 0, const uint *i4* = 0) const [inline]

6.31.3.8 template<class T, uint nc_> T& mdp_array< T, nc_ >::operator() (const uint *i0*, const uint *i1* = 0, const uint *i2* = 0, const uint *i3* = 0, const uint *i4* = 0) [inline]

6.31.3.9 template<class T, uint nc_> void mdp_array< T, nc_ >::operator= (const mdp_array< T, nc_ > & *a*) [inline]

6.31.3.10 template<class T, uint nc_> const T& mdp_array< T, nc_ >::operator[ ] (const uint *i*) const [inline]

6.31.3.11 template<class T, uint nc_> T& mdp_array< T, nc_ >::operator[ ] (const uint *i*) [inline]

6.31.3.12 template<class T, uint nc_> uint mdp_array< T, nc_ >::size () const [inline]

6.31.3.13 template<class T, uint nc_> uint mdp_array< T, nc_ >::size (uint *i*) const [inline]

6.31.3.14 template<class T, uint nc_> uint* mdp_array< T, nc_ >::size_address () [inline]

6.31.4 Friends And Related Function Documentation

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_array.h

6.32 mdp_communicator Class Reference

DO NOT INSTANTIATE use object mdp instead.

`#include <mdp_communicator.h>`Inheritance diagram for mdp_communicator::

Public Member Functions

- mdp_communicator ()
  time spent in communications
- template<class T >
  void put (T &obj, int destination)
- template<class T >
  void put (T &obj, int destination, mdp_request &r)
- template<class T >
  void get (T &obj, int source)
- template<class T >
  void put (T *objptr, mdp_int length, int destination)
- template<class T >
  void put (T *objptr, mdp_int length, int destination, mdp_request &r)
- template<class T >
  void get (T *objptr, mdp_int length, int source)
- void add (float &obj1, float &obj2)
- void add (float *obj1, float *obj2, mdp_int length)
- void add (double &obj1, double &obj2)
- void add (double *obj1, double *obj2, mdp_int length)
- void add (mdp_int &obj1)
- void add (float &obj1)
- void add (double &obj1)
- void add (mdp_int *obj1, mdp_int length)
- void add (float *obj1, mdp_int length)
- void add (double *obj1, mdp_int length)
- void add (mdp_complex &obj1)
- void add (mdp_complex *obj1, mdp_int length)
- void add (mdp_matrix &a)
- void add (mdp_matrix *a, mdp_int length)
- template<class T >
  void add (vector< T > &a)
- template<class T >
  void broadcast (T &obj, int p)
- template<class T >
  void broadcast (T *obj, mdp_int length, int p)
- void wait (mdp_request &r)
- void wait (mdp_request *r, int length)
- const int me ()
- const int nproc ()
- void barrier ()
- int tag (int i, int j)

- void reset_time ()
- double time ()

  returns the time in seconds since call to mdp_communicator::open_wormholes

- void open_wormholes (int argc, char **argv)
- void print_stats ()

  prints statistics about parallel processes

- void close_wormholes ()

  closes parallel communications

- void abort ()

  forces the process to exit(-1)

Public Attributes

- double comm_time

6.32.1 Detailed Description

DO NOT INSTANTIATE use object mdp instead. Example:

```
/// int main(int argc, char**argv) {
///    mdp.open_wormholes(argc,argv);
///    // your code here
///    mdp << 3.14 << endl;  // only process 0 prints
///    mdp.close_wormholes();
///    return 0;
/// }
///
```

6.32.2 Constructor & Destructor Documentation

6.32.2.1 mdp_communicator::mdp_communicator () [inline]

time spent in communications

6.32.3 Member Function Documentation

6.32.3.1 void mdp_communicator::abort () [inline]

forces the process to exit(-1)

Reimplemented from mdp_log.

6.32.3.2 template<class T > void mdp_communicator::add (vector< T > & *a*) [inline]

6.32.3.3 void mdp_communicator::add (mdp_matrix * *a*, mdp_int *length*) [inline]

6.32.3.4 void mdp_communicator::add (mdp_matrix & *a*) [inline]

6.32.3.5 void mdp_communicator::add (mdp_complex * *obj1*, mdp_int *length*) [inline]

6.32.3.6 void mdp_communicator::add (mdp_complex & *obj1*) [inline]

6.32.3.7 void mdp_communicator::add (double * *obj1*, mdp_int *length*) [inline]

6.32.3.8 void mdp_communicator::add (float * *obj1*, mdp_int *length*) [inline]

6.32.3.9 void mdp_communicator::add (mdp_int * *obj1*, mdp_int *length*) [inline]

6.32.3.10 void mdp_communicator::add (double & *obj1*) [inline]

6.32.3.11 void mdp_communicator::add (float & *obj1*) [inline]

6.32.3.12 void mdp_communicator::add (mdp_int & *obj1*) [inline]

6.32.3.13 void mdp_communicator::add (double * *obj1*, double * *obj2*, mdp_int *length*) [inline]

6.32.3.14 void mdp_communicator::add (double & *obj1*, double & *obj2*) [inline]

6.32.3.15 void mdp_communicator::add (float * *obj1*, float * *obj2*, mdp_int *length*) [inline]

6.32.3.16 void mdp_communicator::add (float & *obj1*, float & *obj2*) [inline]

6.32.3.17 void mdp_communicator::barrier () [inline]

6.32.3.18 template<class T > void mdp_communicator::broadcast (T * *obj*, mdp_int *length*, int *p*) [inline]

6.32.3.19 template<class T > void mdp_communicator::broadcast (T & *obj*, int *p*) [inline]

6.32.3.20 void mdp_communicator::close_wormholes () [inline]

closes parallel communications

6.32.3.21 template<class T > void mdp_communicator::get (T * *objptr*, mdp_int *length*, int *source*) [inline]

6.32.3.22 template<class T > void mdp_communicator::get (T & *obj*, int *source*) [inline]

6.32.3.23 const int mdp_communicator::me () [inline]

6.32.3.24 const int mdp_communicator::nproc () [inline]

6.32.3.25 void mdp_communicator::open_wormholes (int *argc*, char ** *argv*) [inline]

starts communications parses command line argument for MPI or PSIM parameters

6.32.3.26 void mdp_communicator::print_stats () [inline]

prints statistics about parallel processes

6.32.3.27 template<class T > void mdp_communicator::put (T * *objptr*, mdp_int *length*, int *destination*, mdp_request & *r*) [inline]

6.32.3.28 template<class T > void mdp_communicator::put (T * *objptr*, mdp_int *length*, int *destination*) [inline]

6.32.3.29 template<class T > void mdp_communicator::put (T & *obj*, int *destination*, mdp_request & *r*) [inline]

6.32.3.30 template<class T > void mdp_communicator::put (T & *obj*, int *destination*) [inline]

6.32.3.31 void mdp_communicator::reset_time () [inline]

6.32.3.32 int mdp_communicator::tag (int *i*, int *j*) [inline]

6.32.3.33 double mdp_communicator::time () [inline]

returns the time in seconds since call to mdp_communicator::open_wormholes

6.32.3.34 void mdp_communicator::wait (mdp_request * *r*, int *length*) [inline]

6.32.3.35 void mdp_communicator::wait (mdp_request & *r*) [inline]

6.32.4 Member Data Documentation

6.32.4.1 double mdp_communicator::comm_time

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_communicator.h

6.33 mdp_complex Class Reference

portable complex numbers

`#include <mdp_complex.h>`

Public Member Functions

- mdp_real & real ()
- mdp_real & imag ()
- const mdp_real & real () const
- const mdp_real & imag () const
- mdp_complex (const mdp_real a=0.0, const mdp_real b=0.0)
- mdp_complex (const mdp_complex &c)
- bool operator== (const mdp_complex &c)
- bool operator!= (const mdp_complex &c)
- void operator+= (const mdp_complex &c)
- void operator-= (const mdp_complex &c)
- void operator*= (const mdp_complex &c)
- void operator/= (const mdp_complex &c)
- void operator+= (const mdp_real c)
- void operator-= (const mdp_real c)
- void operator*= (const mdp_real c)
- void operator/= (const mdp_real c)

Public Attributes

- mdp_real re
- mdp_real im

Friends

- mdp_real real (const mdp_complex &c)
- mdp_real imag (const mdp_complex &c)
- mdp_real abs (const mdp_complex &c)
- mdp_real arg (const mdp_complex &c)
- mdp_complex pow (const mdp_complex &c, mdp_real z)
- mdp_complex sqrt (const mdp_complex &c)
- mdp_complex exp (const mdp_complex &c)
- mdp_complex sin (const mdp_complex &c)
- mdp_complex cos (const mdp_complex &c)
- mdp_complex times_i (const mdp_complex &c)
- mdp_complex times_minus_i (const mdp_complex &c)
- mdp_complex operator- (const mdp_complex &c)
- mdp_complex operator+ (const mdp_complex &c)
- mdp_real phase (const mdp_complex &c)
- mdp_complex conj (const mdp_complex &a)

6.33.1 Detailed Description

portable complex numbers Example:

```
///     mdp_complex x=3+5*I;
///     cout << x.read() << "," << x.imag() << endl;
///     cout << sin(x) << endl;
///
```

6.33.2 Constructor & Destructor Documentation

6.33.2.1 mdp_complex::mdp_complex (const mdp_real *a* = 0.0, const mdp_real *b* = 0.0) [inline]

6.33.2.2 mdp_complex::mdp_complex (const mdp_complex & *c*) [inline]

6.33.3 Member Function Documentation

6.33.3.1 const mdp_real& mdp_complex::imag () const [inline]

6.33.3.2 mdp_real& mdp_complex::imag () [inline]

6.33.3.3 bool mdp_complex::operator!= (const mdp_complex & *c*) [inline]

6.33.3.4 void mdp_complex::operator*= (const mdp_real *c*) [inline]

6.33.3.5 void mdp_complex::operator*= (const mdp_complex & *c*) [inline]

6.33.3.6 void mdp_complex::operator+= (const mdp_real *c*) [inline]

6.33.3.7 void mdp_complex::operator+= (const mdp_complex & *c*) [inline]

6.33.3.8 void mdp_complex::operator-= (const mdp_real *c*) [inline]

6.33.3.9 void mdp_complex::operator-= (const mdp_complex & *c*) [inline]

6.33.3.10 void mdp_complex::operator/= (const mdp_real *c*) [inline]

6.33.3.11 void mdp_complex::operator/= (const mdp_complex & *c*) [inline]

6.33.3.12 bool mdp_complex::operator== (const mdp_complex & *c*) [inline]

6.33.3.13 const mdp_real& mdp_complex::real () const [inline]

6.33.3.14 mdp_real& mdp_complex::real () [inline]

6.33.4 Friends And Related Function Documentation

6.33.4.1 mdp_real abs (const mdp_complex & *c*) [friend]

6.33.4.2 mdp_real arg (const mdp_complex & *c*) [friend]

6.33.4.3 mdp_complex conj (const mdp_complex & *a*) [friend]

6.33.4.4 mdp_complex cos (const mdp_complex & *c*) [friend]

6.33.4.5 mdp_complex exp (const mdp_complex & *c*) [friend]

6.33.4.6 mdp_real imag (const mdp_complex & *c*) [friend]

6.33.4.7 mdp_complex operator+ (const mdp_complex & *c*) [friend]

6.33.4.8 mdp_complex operator- (const mdp_complex & *c*) [friend]

6.33.4.9 mdp_real phase (const mdp_complex & *c*) [friend]

6.33.4.10 mdp_complex pow (const mdp_complex & *c*, mdp_real *z*) [friend]

6.33.4.11 mdp_real real (const mdp_complex & *c*) [friend]

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_complex.h

6.34 mdp_complex_field Class Reference

field of complex numbers or vectors of complex numbers

`#include <mdp_complex_field.h>`Inheritance diagram for mdp_complex_field::

Public Member Functions

- mdp_complex_field ()
- mdp_complex_field (mdp_lattice &lattice, int n=1)
- mdp_complex_field (const mdp_complex_field &other)
- void operator= (const mdp_complex_field &psi)
- void operator∗= (const mdp_complex alpha)
- void operator/= (const mdp_complex alpha)
- void operator∗= (const mdp_real alpha)
- void operator/= (const mdp_real alpha)
- void operator+= (mdp_complex_field &psi)
- void operator-= (mdp_complex_field &psi)
- bool save_as_float (string filename, int processIO=0, mdp_int max_buffer_size=1024, bool load_-header=true, mdp_int skip_bytes=0)
- bool load_as_float (string filename, int processIO=0, mdp_int max_buffer_size=1024, bool load_-header=true, mdp_int skip_bytes=0)
- bool load_as_double (string filename, int processIO=0, mdp_int max_buffer_size=1024, bool load_-header=true, mdp_int skip_bytes=0)
- bool save_as_double (string filename, int processIO=0, mdp_int max_buffer_size=1024, bool load_-header=true, mdp_int skip_bytes=0)

Friends

- mdp_real norm_square (mdp_complex_field &psi, int parity=EVENODD)
- mdp_complex scalar_product (mdp_complex_field &psi, mdp_complex_field &chi, int parity=EVENODD)
- mdp_real real_scalar_product (mdp_complex_field &psi, mdp_complex_field &chi, int parity=EVENODD)
- mdp_real imag_scalar_product (mdp_complex_field &psi, mdp_complex_field &chi, int parity=EVENODD)
- void mdp_add_scaled_field (mdp_complex_field &psi, mdp_real alpha, mdp_complex_field &chi, int parity=EVENODD)
- void mdp_add_scaled_field (mdp_complex_field &psi, mdp_complex alpha, mdp_complex_field &chi, int parity=EVENODD)
- mdp_complex operator∗ (mdp_complex_field &psi, mdp_complex_field &chi)
- mdp_real relative_residue (mdp_complex_field &p, mdp_complex_field &q, int parity=EVENODD)

6.34.1 Detailed Description

field of complex numbers or vectors of complex numbers Example:

```
///    int box[]={10,10,10};
///    mdp_lattice lattice(3,box);
///    mdp_complex_field psi(lattice,10);
///    mdp_site x(lattice);
///    forallsites(x)
///      for(int i=0; i<10; i++)
///         psi(x,i)=0.0+0.0*I;
///
```

6.34.2 Constructor & Destructor Documentation

6.34.2.1 mdp_complex_field::mdp_complex_field () [inline]

6.34.2.2 mdp_complex_field::mdp_complex_field (mdp_lattice & *lattice*, int *n* = 1) [inline]

6.34.2.3 mdp_complex_field::mdp_complex_field (const mdp_complex_field & *other*) [inline]

6.34.3 Member Function Documentation

6.34.3.1 bool mdp_complex_field::load_as_double (string *filename*, int *processIO* = 0, mdp_int *max_buffer_size* = 1024, bool *load_header* = true, mdp_int *skip_bytes* = 0) [inline]

6.34.3.2 bool mdp_complex_field::load_as_float (string *filename*, int *processIO* = 0, mdp_int *max_buffer_size* = 1024, bool *load_header* = true, mdp_int *skip_bytes* = 0) [inline]

6.34.3.3 void mdp_complex_field::operator*= (const mdp_real *alpha*) [inline]

6.34.3.4 void mdp_complex_field::operator*= (const mdp_complex *alpha*) [inline]

6.34.3.5 void mdp_complex_field::operator+= (mdp_complex_field & *psi*) [inline]

6.34.3.6 void mdp_complex_field::operator-= (mdp_complex_field & *psi*) [inline]

6.34.3.7 void mdp_complex_field::operator/= (const mdp_real *alpha*) [inline]

6.34.3.8 void mdp_complex_field::operator/= (const mdp_complex *alpha*) [inline]

6.34.3.9 void mdp_complex_field::operator= (const mdp_complex_field & *psi*) [inline]

Reimplemented in fermi_field, gauge_field, and staggered_field.

6.34.3.10 bool mdp_complex_field::save_as_double (string *filename*, int *processIO* = 0, mdp_int *max_buffer_size* = 1024, bool *load_header* = true, mdp_int *skip_bytes* = 0) [inline]

6.34.3.11 bool mdp_complex_field::save_as_float (string *filename*, int *processIO* = 0, mdp_int *max_buffer_size* = 1024, bool *load_header* = true, mdp_int *skip_bytes* = 0) [inline]

6.34.4 Friends And Related Function Documentation

6.34.4.1 mdp_real imag_scalar_product (mdp_complex_field & *psi*, mdp_complex_field & *chi*, int *parity* = EVENODD) [friend]

6.34.4.2 void mdp_add_scaled_field (mdp_complex_field & *psi*, mdp_complex *alpha*, mdp_complex_field & *chi*, int *parity* = EVENODD) [friend]

6.34.4.3 void mdp_add_scaled_field (mdp_complex_field & *psi*, mdp_real *alpha*, mdp_complex_field & *chi*, int *parity* = EVENODD) [friend]

6.34.4.4 mdp_real norm_square (mdp_complex_field & *psi*, int *parity* = EVENODD) [friend]

6.34.4.5 mdp_complex operator* (mdp_complex_field & *psi*, mdp_complex_field & *chi*) [friend]

6.34.4.6 mdp_real real_scalar_product (mdp_complex_field & *psi*, mdp_complex_field & *chi*, int *parity* = EVENODD) [friend]

6.34.4.7 mdp_real relative_residue (mdp_complex_field & *p*, mdp_complex_field & *q*, int *parity* = EVENODD) [friend]

6.34.4.8 mdp_complex scalar_product (mdp_complex_field & *psi*, mdp_complex_field & *chi*, int *parity* = EVENODD) [friend]

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_complex_field.h

6.35 mdp_field< T > Class Template Reference

most generic field object

`#include <mdp_field.h>`

Public Member Functions

- mdp_field ()
  declare empty field (zero size)
- mdp_field (mdp_lattice &a, int n=1)
  declares a field on lattice a and allocates a vector of n T at each site
- mdp_field (const mdp_field &field)
- bool allocated ()
  checks if a field is allocated of has zero-size
- void allocate_field (mdp_lattice &a, int n=0)
  Allows dynamical allocation of a field that is not allocated.
- void fill_header ()
- void deallocate_memory ()
- void reset_field ()
  do not use, may cause memory leaks
- void deallocate_field ()
  dynamically deallocate field
- virtual ~mdp_field ()
- T & operator() (mdp_site x, int i=0)
  returns component i of the vector of objects T stored at site x
- T & operator() (int idx, int i=0)
- T * operator[ ] (mdp_site x)
  retruns the address of the vector of objects T stored at site x
- T & operator[ ] (mdp_int i)
- T * address (mdp_site x, int i=0) const
- void shift (int i, int mu)
- void operator= (const mdp_field &a)
- void operator= (const T a)
- void operator+= (const mdp_field &a)
- void operator-= (const mdp_field &a)
- template<class T2 >
  void operator*= (const T2 a)
- template<class T2 >
  void operator/= (const T2 a)
- mdp_lattice & lattice () const
  returns by reference the lattice this field is defined on

- mdp_int field_size ()

  returns the total memory in bytes occupied by the field

- mdp_int file_size ()

  returns the total space in bytes required to store the field

- int where_global (mdp_int i)

  only used by mdp_field::load() and mdp_field::save()

- void switch_endianess_4bytes ()
- void switch_endianess_8bytes ()
- mdp_int global_size ()

  lattice size in units of sizeof(T)

- mdp_int physical_size ()
- mdp_int size_per_site ()
- mdp_int physical_local_start (int i=2)
- mdp_int physical_local_stop (int i=2)
- T * physical_address (mdp_int i=0)
- void update (int np=2, int d=-1, int size=1)
- bool load (string filename, int processIO=0, mdp_int max_buffer_size=1024, bool load_header=true, mdp_int skip_bytes=0, bool(*user_read)(FILE *, void *, mdp_int, mdp_int, mdp_int, const mdp_lattice &)=0, bool try_switch_endianess=true)

  Best way to load a field.

- bool save (string filename, int processIO=0, mdp_int max_buffer_size=1024, bool load_header=true, mdp_int skip_bytes=0, bool(*user_write)(FILE *, void *, mdp_int, mdp_int, mdp_int, const mdp_lattice &)=0)

  Best way to save a field.

- bool save_vtk (string filename, int t=-1, int component=-1, int processIO=0, bool ASCII=false)

  Best way to save a field.

Public Attributes

- mdp_field_file_header header

  the field file header, contains data only if field was read from file

Protected Attributes

- mdp_lattice * ptr
- T * m
- mdp_int Tsize
- mdp_int size
- int field_components

6.35.1 Detailed Description

template<class T> class mdp_field< T >

most generic field object Example:

```
///    int box[]={10,10,10};
///    mdp_lattice lattice(3,box);
///    mdp_field<float> psi(lattice,10);
///    mdp_site x(lattice);
///    forallsites(x)
///      for(int i=0; i<10; i++)
///         psi(x,i)=0.0;
///    psi.update(); // synchronization
///    psi.save("myfield");
///    psi.load("myfield");
///
```

6.35.2 Constructor & Destructor Documentation

6.35.2.1 template<class T> mdp_field< T >::mdp_field () [inline]

declare empty field (zero size)

6.35.2.2 template<class T> mdp_field< T >::mdp_field (mdp_lattice & *a*, int *n* = 1) [inline]

declares a field on lattice a and allocates a vector of n T at each site

6.35.2.3 template<class T> mdp_field< T >::mdp_field (const mdp_field< T > & *field*) [inline]

6.35.2.4 template<class T> virtual mdp_field< T >::~mdp_field () [inline, virtual]

6.35.3 Member Function Documentation

6.35.3.1 template<class T> T* mdp_field< T >::address (mdp_site *x*, int *i* = 0) const [inline]

6.35.3.2 template<class T> void mdp_field< T >::allocate_field (mdp_lattice & *a*, int *n* = 0) [inline]

Allows dynamical allocation of a field that is not allocated.

6.35.3.3 template<class T> bool mdp_field< T >::allocated () [inline]

checks if a field is allocated of has zero-size

6.35.3.4 template<class T> void mdp_field< T >::deallocate_field () [inline]

dynamically deallocate field

6.35.3.5 template<class T> void mdp_field< T >::deallocate_memory () [inline]

6.35.3.6 template<class T> mdp_int mdp_field< T >::field_size () [inline]

returns the total memory in bytes occupied by the field

6.35.3.7 template<class T> mdp_int mdp_field< T >::file_size () [inline]

returns the total space in bytes required to store the field

6.35.3.8 template<class T> void mdp_field< T >::fill_header () [inline]

6.35.3.9 template<class T> mdp_int mdp_field< T >::global_size () [inline]

lattice size in units of sizeof(T)

6.35.3.10 template<class T> mdp_lattice& mdp_field< T >::lattice () const [inline]

returns by reference the lattice this field is defined on

6.35.3.11 template<class T > bool mdp_field< T >::load (string *filename*, int *processIO* = 0, mdp_int *max_buffer_size* = 1024, bool *load_header* = true, mdp_int *skip_bytes* = 0, bool(∗)(FILE ∗, void ∗, mdp_int, mdp_int, mdp_int, const mdp_lattice &) *user_read* = 0, bool *try_switch_endianess* = true) [inline]

Best way to load a field.

6.35.3.12 template<class T> T& mdp_field< T >::operator() (int *idx*, int *i* = 0) [inline]

6.35.3.13 template<class T> T& mdp_field< T >::operator() (mdp_site *x*, int *i* = 0) [inline]

returns component i of the vector of objects T stored at site x

Reimplemented in mdp_nmatrix_field, mdp_nvector_field, and mdp_vector_field.

6.35.3.14 template<class T> template<class T2 > void mdp_field< T >::operator∗= (const T2 *a*) [inline]

6.35.3.15 template<class T> void mdp_field< T >::operator+= (const mdp_field< T > & *a*) [inline]

6.35.3.16 template<class T> void mdp_field< T >::operator-= (const mdp_field< T > & *a*) [inline]

6.35.3.17 template<class T> template<class T2 > void mdp_field< T >::operator/= (const T2 *a*) [inline]

6.35.3.18 template<class T> void mdp_field< T >::operator= (const T *a*) [inline]

Reimplemented in dwfermi_field, fermi_field, sdwf_field, and staggered_field.

6.35.3.19 template<class T> void mdp_field< T >::operator= (const mdp_field< T > & *a*) [inline]

6.35.3.20 template<class T> T& mdp_field< T >::operator[ ] (mdp_int *i*) [inline]

6.35.3.21 template<class T> T* mdp_field< T >::operator[ ] (mdp_site *x*) [inline]

retruns the address of the vector of objects T stored at site x

6.35.3.22 template<class T> T* mdp_field< T >::physical_address (mdp_int *i* = 0) [inline]

6.35.3.23 template<class T> mdp_int mdp_field< T >::physical_local_start (int *i* = 2) [inline]

6.35.3.24 template<class T> mdp_int mdp_field< T >::physical_local_stop (int *i* = 2) [inline]

6.35.3.25 template<class T> mdp_int mdp_field< T >::physical_size () [inline]

6.35.3.26 template<class T> void mdp_field< T >::reset_field () [inline]

do not use, may cause memory leaks

6.35.3.27 template<class T > bool mdp_field< T >::save (string *filename*, int *processIO* = 0, mdp_int *max_buffer_size* = 1024, bool *load_header* = true, mdp_int *skip_bytes* = 0, bool(*)(FILE *, void *, mdp_int, mdp_int, mdp_int, const mdp_lattice &) *user_write* = 0) [inline]

Best way to save a field.

6.35.3.28 template<class T > bool mdp_field< T >::save_vtk (string *filename*, int *t* = -1, int *component* = -1, int *processIO* = 0, bool *ASCII* = false) [inline]

Best way to save a field.

6.35.3.29 template<class T> void mdp_field< T >::shift (int *i*, int *mu*) [inline]

shifts the entire fields in direction mu of i steps (i can be positive or negative) note that if i=1, field(x-mu) is assigned to field(x) function requires communication

6.35.3.30 template<class T> mdp_int mdp_field< T >::size_per_site () [inline]

6.35.3.31 template<class T> void mdp_field< T >::switch_endianess_4bytes () [inline]

6.35.3.32 template<class T> void mdp_field< T >::switch_endianess_8bytes () [inline]

6.35.3.33 template<class T > void mdp_field< T >::update (int *np* = 2, int *d* = -1, int *ncomp* = 1) [inline]

the most important communication function in MDP. it must be called after each field variables are modified. it restores the syncronization between parallel processes.

The only communication function for a field object To be invoked every time field variables are assigned and Need to be sinchronized between the parallel processes

6.35.3.34 template<class T> int mdp_field< T >::where_global (mdp_int *i*) [inline]

only used by mdp_field::load() and mdp_field::save()

6.35.4 Member Data Documentation

6.35.4.1 template<class T> int mdp_field< T >::field_components [protected]

6.35.4.2 template<class T> mdp_field_file_header mdp_field< T >::header

the field file header, contains data only if field was read from file

6.35.4.3 template<class T> T* mdp_field< T >::m [protected]

6.35.4.4 template<class T> mdp_lattice* mdp_field< T >::ptr [protected]

6.35.4.5 template<class T> mdp_int mdp_field< T >::size [protected]

6.35.4.6 template<class T> mdp_int mdp_field< T >::Tsize [protected]

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_field.h
- /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_load.h
- /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_save.h
- /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_save_vtk.h
- /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_update.h

6.36 mdp_field_file_header Class Reference

header for field file IO

`#include <mdp_field.h>`

Public Member Functions

- mdp_field_file_header ()
- void reset ()
- void set_time ()

Public Attributes

- char file_id [60]
- char program_version [60]
- char creation_date [60]
- uint32_t endianess
- int32_t ndim
- int32_t box [10]
- int32_t bytes_per_site
- int32_t sites

Friends

- bool switch_header_endianess (mdp_field_file_header &header)

  tries to swicth the endianess of the numerical members of the header

6.36.1 Detailed Description

header for field file IO Used to store the binary file haeader that precedes the data When storing an object of class mdp_field<> in a file

6.36.2 Constructor & Destructor Documentation

6.36.2.1 mdp_field_file_header::mdp_field_file_header () [inline]

6.36.3 Member Function Documentation

6.36.3.1 void mdp_field_file_header::reset () [inline]

6.36.3.2 void mdp_field_file_header::set_time () [inline]

6.36.4 Friends And Related Function Documentation

6.36.4.1 bool switch_header_endianess (mdp_field_file_header & *header*) [friend]

tries to swicth the endianess of the numerical members of the header

6.36.5 Member Data Documentation

6.36.5.1 int32_t mdp_field_file_header::box[10]

6.36.5.2 int32_t mdp_field_file_header::bytes_per_site

6.36.5.3 char mdp_field_file_header::creation_date[60]

6.36.5.4 uint32_t mdp_field_file_header::endianess

6.36.5.5 char mdp_field_file_header::file_id[60]

6.36.5.6 int32_t mdp_field_file_header::ndim

6.36.5.7 char mdp_field_file_header::program_version[60]

6.36.5.8 int32_t mdp_field_file_header::sites

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_field.h

6.37 mdp_jackboot Class Reference

coniatiner class for jackknife and boostrap analysis

`#include <mdp_jackboot.h>`

Public Member Functions

- mdp_jackboot ()
- mdp_jackboot (int nconf_, int narg_=1)

  allocate container for nconf_ datasets of nargs_ numbers each

- void dimension (int nconf_, int narg_=1)
- virtual ~mdp_jackboot ()
- float * address (int conf)
- float & operator() (int present_conf, int arg)
- float & operator() (int arg)
- void plain (int i)
- void makesample (int *p, int nboot)
- float mean ()
- float j_err ()
- float b_err (int nboot=100)

Public Attributes

- float(* f)(float *, void *)
- int nconf
- int narg
- int conf
- float * m
- void * handle

Friends

- float mdp_jackboot_plain (float *x, void *a)

6.37.1 Detailed Description

coniatiner class for jackknife and boostrap analysis Example:

```
///    mdp_jackboot jb(10,2);
///    for(int k=0; k<10; k++) {
///       jb(k,0)=mdp_random.plain();
///       jb(k,1)=mdp_random.plain();
///    }
///    jb.plain(0);
///    cout << "mean of jb(k,0) =" << mean() << endl;
///    cout << "jackknife of mean jb(k,0) =" << j_err() << endl;
///    cout << "boostrap of mean jb(k,0) =" << b_err() << endl;
///    jb.plain(1);
///    cout << "mean of jb(k,1) =" << mean() << endl;
///    cout << "jackknife of mean jb(k,1) =" << j_err() << endl;
///    cout << "boostrap of mean jb(k,1) =" << b_err() << endl;
///
```

6.37.2 Constructor & Destructor Documentation

6.37.2.1 mdp_jackboot::mdp_jackboot () [inline]

6.37.2.2 mdp_jackboot::mdp_jackboot (int *nconf_*, int *narg_* = 1) [inline]

allocate container for nconf_ datasets of nargs_ numbers each

6.37.2.3 virtual mdp_jackboot::~mdp_jackboot () [inline, virtual]

6.37.3 Member Function Documentation

6.37.3.1 float* mdp_jackboot::address (int *conf*) [inline]

6.37.3.2 float mdp_jackboot::b_err (int *nboot* = 100) [inline]

6.37.3.3 void mdp_jackboot::dimension (int *nconf_*, int *narg_* = 1) [inline]

6.37.3.4 float mdp_jackboot::j_err () [inline]

6.37.3.5 void mdp_jackboot::makesample (int * *p*, int *nboot*) [inline]

6.37.3.6 float mdp_jackboot::mean () [inline]

6.37.3.7 float& mdp_jackboot::operator() (int *arg*) [inline]

6.37.3.8 float& mdp_jackboot::operator() (int *present_conf*, int *arg*) [inline]

6.37.3.9 void mdp_jackboot::plain (int *i*) [inline]

6.37.4 Friends And Related Function Documentation

6.37.4.1 float mdp_jackboot_plain (float * *x*, void * *a*) [friend]

6.37.5 Member Data Documentation

6.37.5.1 int mdp_jackboot::conf

6.37.5.2 float(* mdp_jackboot::f)(float *, void *)

6.37.5.3 void* mdp_jackboot::handle

6.37.5.4 float* mdp_jackboot::m

6.37.5.5 int mdp_jackboot::narg

6.37.5.6 int mdp_jackboot::nconf

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_jackboot.h

6.38 mdp_lattice Class Reference

distributed lattice object

`#include <mdp_lattice.h>`

Public Member Functions

- mdp_int global_coordinate (int *x)
- void global_coordinate (mdp_int global_idx, int *x)
- int compute_parity (int *x)
- mdp_lattice ()
- mdp_lattice (int ndim_, int nx_[ ], int(*where_)(int *, int, int *)=default_partitioning0, void(*neighbour_)(int, int *, int *, int *, int, int *)=torus_topology, mdp_int random_seed_=0, int next_next_=1, bool local_random_=true)
- mdp_lattice (int ndim_, int ndir_, int nx_[ ], int(*where_)(int *, int, int *)=default_partitioning0, void(*neighbour_)(int, int *, int *, int *, int, int *)=torus_topology, mdp_int random_seed_=0, int next_next_=1, bool local_random_=true)

  for weird stuff

- void allocate_lattice (int ndim_, int nx_[ ], int(*where_)(int *, int, int *)=default_partitioning0, void(*neighbour_)(int, int *, int *, int *, int, int *)=torus_topology, mdp_int random_seed_=0, int next_next_=1, bool local_random_=true)
- void allocate_lattice (int ndim_, int ndir_, int nx_[ ], int(*where_)(int *, int, int *)=default_partitioning0, void(*neighbour_)(int, int *, int *, int *, int, int *)=torus_topology, mdp_int random_seed_=0, int next_next_=1, bool local_random_=true)

  for weird stuff

- virtual ~mdp_lattice ()
- void deallocate_memory ()

  dynamically deallocate a lattice

- void initialize_random (mdp_int random_seed_=0)
- mdp_prng & random (mdp_site)

  Returns the local object mdp_prng at site x of the lattice.

- int n_dimensions () const

  *number of dimensions of the lattice (deprecated_*

- int n_directions () const

  number of directions one can move on the lattice; usually same as ndim

- mdp_int size () const

  number of sites of the lattice

- mdp_int size (const int mu) const

  size of the lattice in direction mu

- mdp_int local_volume () const

  number of lattice sites stored locally by current process

- mdp_int global_volume () const

  total lattice volume (deprecated)

- mdp_int move_up (const mdp_int idx, const int mu) const
- mdp_int move_down (const mdp_int idx, const int mu) const
- mdp_int local (mdp_int idx) const
- mdp_int global (mdp_int idx) const
- int site_parity (const mdp_int idx) const
- mdp_int start_index (const int process, int p=EVENODD) const
- mdp_int stop_index (const int process, int p=EVENODD) const

Public Attributes

- int ndim
- int ndir
- int next_next
- int * nx
- mdp_int nvol
- mdp_int nvol_gl
- mdp_int nvol_in
- mdp_int * gl
- mdp_int * lg
- FILE * lg_file
- mdp_int ** up
- mdp_int ** dw
- int ** co
- int * wh
- int * parity
- mdp_int start [_NprocMax_][2]
- mdp_int stop [_NprocMax_][2]
- mdp_int len_to_send [_NprocMax_][2]
- mdp_int * to_send [_NprocMax_]
- bool local_random_generator
- int(* where)(int *, int, int *)
- void(* neighbour)(int, int *, int *, int *, int, int *)

6.38.1 Detailed Description

distributed lattice object Example:

```
///    int box[]={3,3,3};
///    int seed=0, border_width=1;
///    mdp_lattice lattice(3,box,default_partitioning0,
///                        torus_topology,seed,border_width);
///    mdp_site x(lattice);
///    forallsites(x)
///      cout << lattice.random(x).plain() << endl;
///
```

6.38.2 Constructor & Destructor Documentation

6.38.2.1 mdp_lattice::mdp_lattice () [inline]

6.38.2.2 mdp_lattice::mdp_lattice (int *ndim_*, int *nx_*[ ], int(∗)(int ∗, int, int ∗) *where_* = default_partitioning0, void(∗)(int, int ∗, int ∗, int ∗, int, int ∗) *neighbour_* = torus_topology, mdp_int *random_seed_* = 0, int *next_next_* = 1, bool *local_random_* = true) [inline]

declares a lattice object

Parameters:

ndim_ dimensions of the lattice

nx_ size of the lattice

where pointer to a partitioning function neighbour_ pointer to a topology function.

random_seed_ seed to be used by the parallel prng

next_next_ size of the buffer between neighbour processes

local_random_ true is local random generator is required

6.38.2.3 mdp_lattice::mdp_lattice (int *ndim_*, int *ndir_*, int *nx_*[ ], int(∗)(int ∗, int, int ∗) *where_* = default_partitioning0, void(∗)(int, int ∗, int ∗, int ∗, int, int ∗) *neighbour_* = torus_topology, mdp_int *random_seed_* = 0, int *next_next_* = 1, bool *local_random_* = true) [inline]

for weird stuff

6.38.2.4 virtual mdp_lattice::∼mdp_lattice () [inline, virtual]

6.38.3 Member Function Documentation

6.38.3.1 void mdp_lattice::allocate_lattice (int *ndim_*, int *ndir_*, int *nx_*[ ], int(∗)(int ∗, int, int ∗) *where_* = default_partitioning0, void(∗)(int, int ∗, int ∗, int ∗, int, int ∗) *neighbour_* = torus_topology, mdp_int *random_seed_* = 0, int *next_next_* = 1, bool *local_random_* = true) [inline]

for weird stuff

6.38.3.2 void mdp_lattice::allocate_lattice (int *ndim_*, int *nx_*[ ], int(∗)(int ∗, int, int ∗) *where_* = default_partitioning0, void(∗)(int, int ∗, int ∗, int ∗, int, int ∗) *neighbour_* = torus_topology, mdp_int *random_seed_* = 0, int *next_next_* = 1, bool *local_random_* = true) [inline]

reallocate a lattice dynamically

Parameters:

ndim_ dimensions of the lattice

nx_ size of the lattice

where pointer to a partitioning function neighbour_ pointer to a topology function.

random_seed_ seed to be used by the parallel prng

next_next_ size of the buffer between neighbour processes

local_random_ true is local random generator is required

6.38.3.3 int mdp_lattice::compute_parity (int * *x*) [inline]

6.38.3.4 void mdp_lattice::deallocate_memory () [inline]

dynamically deallocate a lattice

6.38.3.5 mdp_int mdp_lattice::global (mdp_int *idx*) const [inline]

6.38.3.6 void mdp_lattice::global_coordinate (mdp_int *global_idx*, int * *x*) [inline]

6.38.3.7 mdp_int mdp_lattice::global_coordinate (int * *x*) [inline]

6.38.3.8 mdp_int mdp_lattice::global_volume () const [inline]

total lattice volume (deprecated)

6.38.3.9 void mdp_lattice::initialize_random (mdp_int *random_seed_* = 0) [inline]

6.38.3.10 mdp_int mdp_lattice::local (mdp_int *idx*) const [inline]

6.38.3.11 mdp_int mdp_lattice::local_volume () const [inline]

number of lattice sites stored locally by current process

6.38.3.12 mdp_int mdp_lattice::move_down (const mdp_int *idx*, const int *mu*) const [inline]

6.38.3.13 mdp_int mdp_lattice::move_up (const mdp_int *idx*, const int *mu*) const [inline]

6.38.3.14 int mdp_lattice::n_dimensions () const [inline]

number of dimensions of the lattice (deprecated_

6.38.3.15 int mdp_lattice::n_directions () const [inline]

number of directions one can move on the lattice; usually same as ndim

6.38.3.16 mdp_prng & mdp_lattice::random (mdp_site *x*) [inline]

Returns the local object mdp_prng at site x of the lattice.

6.38.3.17 int mdp_lattice::site_parity (const mdp_int *idx*) const [inline]

6.38.3.18 mdp_int mdp_lattice::size (const int *mu*) const [inline]

size of the lattice in direction mu

6.38.3.19 mdp_int mdp_lattice::size () const [inline]

number of sites of the lattice

6.38.3.20 mdp_int mdp_lattice::start_index (const int *process*, int *p* = EVENODD) const [inline]

6.38.3.21 mdp_int mdp_lattice::stop_index (const int *process*, int *p* = EVENODD) const [inline]

6.38.4 Member Data Documentation

6.38.4.1 int** mdp_lattice::co

6.38.4.2 mdp_int** mdp_lattice::dw

6.38.4.3 mdp_int* mdp_lattice::gl

6.38.4.4 mdp_int mdp_lattice::len_to_send[_NprocMax_][2]

6.38.4.5 mdp_int* mdp_lattice::lg

6.38.4.6 FILE* mdp_lattice::lg_file

6.38.4.7 bool mdp_lattice::local_random_generator

6.38.4.8 int mdp_lattice::ndim

6.38.4.9 int mdp_lattice::ndir

6.38.4.10 void(* mdp_lattice::neighbour)(int, int *, int *, int, int *)

6.38.4.11 int mdp_lattice::next_next

6.38.4.12 mdp_int mdp_lattice::nvol

6.38.4.13 mdp_int mdp_lattice::nvol_gl

6.38.4.14 mdp_int mdp_lattice::nvol_in

6.38.4.15 int* mdp_lattice::nx

6.38.4.16 int* mdp_lattice::parity

6.38.4.17 mdp_int mdp_lattice::start[_NprocMax_][2]

6.38.4.18 mdp_int mdp_lattice::stop[_NprocMax_][2]

6.38.4.19 mdp_int* mdp_lattice::to_send[_NprocMax_]

6.38.4.20 mdp_int** mdp_lattice::up

6.38.4.21 int* mdp_lattice::wh

6.38.4.22 int(* mdp_lattice::where)(int *, int, int *)

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_lattice.h
- /Users/mdipierro/fermiqcd/development/Libraries/mdp_site.h

6.39 mdp_log Class Reference

base class of class mdp_communicator (DO NOT INSTANTIATE)

`#include <mdp_log.h>`Inheritance diagram for mdp_log::

Public Member Functions

- void abort ()
- void set_level (int i)
- mdp_log ()
- void connect (ostream &os1)
- void connect (ofstream &os2)
- void error_message (string s, string file, int line)
- void begin_function (string s)
- void end_function (string s)
- template<class T >
  mdp_log & operator<< (const T x)

Public Attributes

- bool print

6.39.1 Detailed Description

base class of class mdp_communicator (DO NOT INSTANTIATE)

See also:

class mdp_communicator

6.39.2 Constructor & Destructor Documentation

6.39.2.1 mdp_log::mdp_log () `[inline]`

6.39.3 Member Function Documentation

6.39.3.1 void mdp_log::abort () `[inline]`

Reimplemented in mdp_communicator.

6.39.3.2 void mdp_log::begin_function (string *s*) [inline]

6.39.3.3 void mdp_log::connect (ofstream & *os2*) [inline]

6.39.3.4 void mdp_log::connect (ostream & *os1*) [inline]

6.39.3.5 void mdp_log::end_function (string *s*) [inline]

6.39.3.6 void mdp_log::error_message (string *s*, string *file*, int *line*) [inline]

6.39.3.7 template<class T > mdp_log& mdp_log::operator<< (const T *x*) [inline]

6.39.3.8 void mdp_log::set_level (int *i*) [inline]

6.39.4 Member Data Documentation

6.39.4.1 bool mdp_log::print

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_log.h

6.40 mdp_matrix Class Reference

matrices of complex numbers

`#include <mdp_matrix.h>`

Public Member Functions

- void allocate ()
- void reallocate ()
- void deallocate ()
- void dimension (const uint, const uint)
- mdp_matrix ()
- mdp_matrix (const mdp_matrix &a)
- mdp_matrix (const uint r, const uint c)
- mdp_matrix (mdp_complex *z, const uint r, const uint c)
- virtual ~mdp_matrix ()
- const mdp_matrix & operator= (const mdp_matrix &a)
- mdp_complex & operator[ ] (const uint i)
- const mdp_complex & operator[ ] (const uint i) const
- mdp_complex & operator() (const uint i, const uint j)
- const mdp_complex & operator() (const uint i, const uint j) const
- mdp_matrix operator() (const uint i)
- mdp_complex * address ()
- const uint rows () const
- const uint cols () const
- const uint size () const
- uint rowmax () const
- uint colmax () const
- mdp_matrix operator+= (const mdp_matrix &a)
- mdp_matrix operator-= (const mdp_matrix &a)
- mdp_matrix operator*= (const mdp_matrix &a)
- mdp_matrix operator/= (const mdp_matrix &a)
- mdp_matrix operator+= (mdp_complex a)
- mdp_matrix operator-= (mdp_complex a)
- mdp_matrix operator*= (mdp_complex a)
- mdp_matrix operator/= (mdp_complex a)
- mdp_matrix operator+= (mdp_real a)
- mdp_matrix operator-= (mdp_real a)
- mdp_matrix operator*= (mdp_real a)
- mdp_matrix operator/= (mdp_real a)
- void operator= (mdp_complex a)
- void operator= (mdp_real a)

Friends

- void prepare (mdp_matrix &)
- mdp_matrix operator+ (const mdp_matrix &a)
- mdp_matrix operator- (const mdp_matrix &a)
- mdp_matrix operator+ (const mdp_matrix &a, const mdp_matrix &b)
- mdp_matrix operator- (const mdp_matrix &a, const mdp_matrix &b)
- mdp_matrix operator* (const mdp_matrix &a, const mdp_matrix &b)
- mdp_matrix operator/ (const mdp_matrix &a, const mdp_matrix &b)
- mdp_matrix operator+ (const mdp_matrix &a, mdp_complex b)
- mdp_matrix operator- (const mdp_matrix &a, mdp_complex b)
- mdp_matrix operator* (const mdp_matrix &a, mdp_complex b)
- mdp_matrix operator/ (const mdp_matrix &a, mdp_complex b)
- mdp_matrix operator+ (mdp_complex a, const mdp_matrix &b)
- mdp_matrix operator- (mdp_complex a, const mdp_matrix &b)
- mdp_matrix operator* (mdp_complex a, const mdp_matrix &b)
- mdp_matrix operator/ (mdp_complex a, const mdp_matrix &b)
- mdp_matrix operator+ (const mdp_matrix &a, mdp_real b)
- mdp_matrix operator- (const mdp_matrix &a, mdp_real b)
- mdp_matrix operator* (const mdp_matrix &a, mdp_real b)
- mdp_matrix operator/ (const mdp_matrix &a, mdp_real b)
- mdp_matrix operator+ (mdp_real a, const mdp_matrix &b)
- mdp_matrix operator- (mdp_real a, const mdp_matrix &b)
- mdp_matrix operator* (mdp_real a, const mdp_matrix &b)
- mdp_matrix operator/ (mdp_real a, const mdp_matrix &b)
- mdp_matrix inv (const mdp_matrix &a)
- mdp_matrix pow (const mdp_matrix &a, uint b)
- mdp_matrix exp (const mdp_matrix &a)
- mdp_matrix log (const mdp_matrix &a)
- mdp_matrix sin (const mdp_matrix &a)
- mdp_matrix cos (const mdp_matrix &a)
- mdp_matrix mdp_identity ()
- mdp_matrix mdp_zero ()
- mdp_real max (const mdp_matrix &a)
- mdp_matrix submatrix (const mdp_matrix &a, uint i, uint j)
- mdp_complex det (const mdp_matrix &a)
- mdp_complex trace (const mdp_matrix &a)
- mdp_matrix hermitian (const mdp_matrix &a)
- mdp_matrix transpose (const mdp_matrix &a)
- mdp_matrix conj (const mdp_matrix &a)

6.40.1 Detailed Description

matrices of complex numbers Example:

```
///    mdp_matrix A,B;
///    A.dimension(3,3);
///    A(0,0)=A(1,1)=A(2,2)=A(1,2)=1.0+I/2;
///    B=A+inv(A)+exp(A+5);
///
```

6.40.2 Constructor & Destructor Documentation

6.40.2.1 mdp_matrix::mdp_matrix ()

6.40.2.2 mdp_matrix::mdp_matrix (const mdp_matrix & *a*)

6.40.2.3 mdp_matrix::mdp_matrix (const uint *r*, const uint *c*)

6.40.2.4 mdp_matrix::mdp_matrix (mdp_complex * *z*, const uint *r*, const uint *c*)

6.40.2.5 mdp_matrix::~mdp_matrix () [virtual]

6.40.3 Member Function Documentation

6.40.3.1 mdp_complex * mdp_matrix::address () [inline]

6.40.3.2 void mdp_matrix::allocate () [inline]

6.40.3.3 uint mdp_matrix::colmax () const [inline]

6.40.3.4 const uint mdp_matrix::cols () const [inline]

6.40.3.5 void mdp_matrix::deallocate () [inline]

6.40.3.6 void mdp_matrix::dimension (const uint *a*, const uint *b*)

6.40.3.7 mdp_matrix mdp_matrix::operator() (const uint *i*) [inline]

6.40.3.8 const mdp_complex & mdp_matrix::operator() (const uint *i*, const uint *j*) const [inline]

6.40.3.9 mdp_complex & mdp_matrix::operator() (const uint *i*, const uint *j*) [inline]

6.40.3.10 mdp_matrix mdp_matrix::operator*= (mdp_real *a*) [inline]

6.40.3.11 mdp_matrix mdp_matrix::operator*= (mdp_complex *a*) [inline]

6.40.3.12 mdp_matrix mdp_matrix::operator*= (const mdp_matrix & *a*) [inline]

6.40.3.13 mdp_matrix mdp_matrix::operator+= (mdp_real *a*) [inline]

6.40.3.14 mdp_matrix mdp_matrix::operator+= (mdp_complex *a*) [inline]

6.40.3.15 mdp_matrix mdp_matrix::operator+= (const mdp_matrix & *a*) [inline]

6.40.3.16 mdp_matrix mdp_matrix::operator-= (mdp_real *a*) [inline]

6.40.3.17 mdp_matrix mdp_matrix::operator-= (mdp_complex *a*) [inline]

6.40.3.18 mdp_matrix mdp_matrix::operator-= (const mdp_matrix & *a*) [inline]

6.40.3.19 mdp_matrix mdp_matrix::operator/= (mdp_real *a*) [inline]

6.40.3.20 mdp_matrix mdp_matrix::operator/= (mdp_complex *a*) [inline]

6.40.3.21 mdp_matrix mdp_matrix::operator/= (const mdp_matrix & *a*) [inline]

6.40.3.22 void mdp_matrix::operator= (mdp_real *a*) [inline]

6.40.3.23 void mdp_matrix::operator= (mdp_complex *a*) [inline]

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_matrix.h

6.41 mdp_matrix_field Class Reference

a field of matrices

`#include <mdp_matrix_field.h>`Inheritance diagram for mdp_matrix_field::

Public Member Functions

- mdp_matrix_field ()
- mdp_matrix_field (mdp_matrix_field &field)
- mdp_matrix_field (mdp_lattice &a, int i, int j)
- void allocate_mdp_matrix_field (mdp_lattice &a, int i, int j)
- mdp_matrix operator() (mdp_site x)
- mdp_complex & operator() (mdp_site x, int i, int j)
- const mdp_complex & operator() (mdp_site x, int i, int j) const

Public Attributes

- int rows
- int columns
- int imax

6.41.1 Detailed Description

a field of matrices Example:

```
///    int box[]={10,10,10};
///    mdp_lattice lattice(3,box);
///    mdp_matrix_field h(lattice,5,5);
///    mdp_site x(lattice);
///    forallsites(x)
///       h(x)=lattice.random(x).SU(5);
///
```

6.41.2 Constructor & Destructor Documentation

6.41.2.1 mdp_matrix_field::mdp_matrix_field () [inline]

6.41.2.2 mdp_matrix_field::mdp_matrix_field (mdp_matrix_field & *field*) [inline]

6.41.2.3 mdp_matrix_field::mdp_matrix_field (mdp_lattice & *a*, int *i*, int *j*) [inline]

6.41.3 Member Function Documentation

6.41.3.1 void mdp_matrix_field::allocate_mdp_matrix_field (mdp_lattice & *a*, int *i*, int *j*) [inline]

6.41.3.2 const mdp_complex& mdp_matrix_field::operator() (mdp_site *x*, int *i*, int *j*) const [inline]

6.41.3.3 mdp_complex& mdp_matrix_field::operator() (mdp_site *x*, int *i*, int *j*) [inline]

6.41.3.4 mdp_matrix mdp_matrix_field::operator() (mdp_site *x*) [inline]

6.41.4 Member Data Documentation

6.41.4.1 int mdp_matrix_field::columns

6.41.4.2 int mdp_matrix_field::imax

6.41.4.3 int mdp_matrix_field::rows

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_matrix_field.h

6.42 mdp_measure Class Reference

implements error propagation

`#include <mdp_measure.h>`

Public Member Functions

- int getnum ()
- float getmean ()
- float getmerr ()
- mdp_measure ()
- mdp_measure (float mean_, float error_, int num_=1)
- void reset ()
- void set (float x, float dx, int i=1)
- void operator<< (float x)
- void operator>> (float &x)

Public Attributes

- int num
- float mean
- float error

Friends

- mdp_measure operator+ (mdp_measure a, mdp_measure b)
- mdp_measure operator- (mdp_measure a, mdp_measure b)
- mdp_measure operator* (mdp_measure a, mdp_measure b)
- mdp_measure operator/ (mdp_measure a, mdp_measure b)
- mdp_measure operator+ (float a, mdp_measure b)
- mdp_measure operator- (float a, mdp_measure b)
- mdp_measure operator* (float a, mdp_measure b)
- mdp_measure operator/ (float a, mdp_measure b)
- mdp_measure operator+ (mdp_measure a, float b)
- mdp_measure operator- (mdp_measure a, float b)
- mdp_measure operator* (mdp_measure a, float b)
- mdp_measure operator/ (mdp_measure a, float b)
- mdp_measure exp (mdp_measure a)
- mdp_measure log (mdp_measure a)
- mdp_measure pow (mdp_measure a, float b)
- mdp_measure sin (mdp_measure a)
- mdp_measure cos (mdp_measure a)
- void print (mdp_measure a)

6.42.1 Detailed Description

implements error propagation Example:

```
///    mdp_measure m;
///    // store 10 measurements
///    for(int i=0; i<10; i++)
///       m << 3.0+mdp_random.gaussian(2.0);
///    m=sin(exp(m)+m);
///    cout << m.getmean() << "+/-" << m.geterr() << endl;
///
```

Assumes gaussian error propagation

6.42.2 Constructor & Destructor Documentation

6.42.2.1 mdp_measure::mdp_measure () [inline]

6.42.2.2 mdp_measure::mdp_measure (float *mean_*, float *error_*, int *num_* = 1) [inline]

6.42.3 Member Function Documentation

6.42.3.1 float mdp_measure::getmean () [inline]

6.42.3.2 float mdp_measure::getmerr () [inline]

6.42.3.3 int mdp_measure::getnum () [inline]

6.42.3.4 void mdp_measure::operator<< (float *x*) [inline]

6.42.3.5 void mdp_measure::operator>> (float & *x*) [inline]

6.42.3.6 void mdp_measure::reset () [inline]

6.42.3.7 void mdp_measure::set (float *x*, float *dx*, int *i* = 1) [inline]

6.42.4 Friends And Related Function Documentation

6.42.4.1 mdp_measure cos (mdp_measure *a*) [friend]

6.42.4.2 mdp_measure exp (mdp_measure *a*) [friend]

6.42.4.3 mdp_measure log (mdp_measure *a*) [friend]

6.42.4.4 mdp_measure operator* (mdp_measure *a*, float *b*) [friend]

6.42.4.5 mdp_measure operator* (float *a*, mdp_measure *b*) [friend]

6.42.4.6 mdp_measure operator* (mdp_measure *a*, mdp_measure *b*) [friend]

6.42.4.7 mdp_measure operator+ (mdp_measure *a*, float *b*) [friend]

6.42.4.8 mdp_measure operator+ (float *a*, mdp_measure *b*) [friend]

6.42.4.9 mdp_measure operator+ (mdp_measure *a*, mdp_measure *b*) [friend]

6.42.4.10 mdp_measure operator- (mdp_measure *a*, float *b*) [friend]

6.42.4.11 mdp_measure operator- (float *a*, mdp_measure *b*) [friend]

6.42.4.12 mdp_measure operator- (mdp_measure *a*, mdp_measure *b*) [friend]

6.42.4.13 mdp_measure operator/ (mdp_measure *a*, float *b*) [friend]

6.42.4.14 mdp_measure operator/ (float *a*, mdp_measure *b*) [friend]

6.42.4.15 mdp_measure operator/ (mdp_measure *a*, mdp_measure *b*) [friend]

6.42.4.16 mdp_measure pow (mdp_measure *a*, float *b*) [friend]

6.42.4.17 void print (mdp_measure *a*) [friend]

6.42.4.18 mdp_measure sin (mdp_measure *a*) [friend]

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_measure.h

6.43 mdp_nmatrix_field Class Reference

field of vectors of matrices

`#include <mdp_nmatrix_field.h>`Inheritance diagram for mdp_nmatrix_field::

Public Member Functions

- mdp_nmatrix_field ()
- mdp_nmatrix_field (mdp_nmatrix_field &field)
- mdp_nmatrix_field (mdp_lattice &a, int n, int i, int j)

  declares a field object that at each site as vector of n ixj matrices

- void allocate_mdp_nmatrix_field (mdp_lattice &a, int n, int i, int j)

  dynamically allocates a field object that at each site as vector of n ixj matrices

- mdp_matrix operator() (mdp_site x, int n)

  returns the n-th matrix stored at site x

- mdp_complex & operator() (mdp_site x, int n, int i, int j)

  returns the (i,j) component of the n-th matrix stored at site x

- const mdp_complex & operator() (mdp_site x, int n, int i, int j) const

Public Attributes

- uint rows
- uint columns
- uint matrices
- uint imax
- uint imax2

6.43.1 Detailed Description

field of vectors of matrices Example:

```
///    int box[]={10,10,10};
///    mdp_lattice lattice(3,box);
///    mdp_nmatrix_field h(lattice,10,3,3);
///    mdp_site x(lattice);
///    forallsites(x)
///      for(int i=0; i<10; i++)
///        h(x,i)=lattice.random(x).SU(3);
///
```

6.43.2 Constructor & Destructor Documentation

6.43.2.1 mdp_nmatrix_field::mdp_nmatrix_field () [inline]

6.43.2.2 mdp_nmatrix_field::mdp_nmatrix_field (mdp_nmatrix_field & *field*) [inline]

6.43.2.3 mdp_nmatrix_field::mdp_nmatrix_field (mdp_lattice & *a*, int *n*, int *i*, int *j*) [inline]

declares a field object that at each site as vector of n ixj matrices

6.43.3 Member Function Documentation

6.43.3.1 void mdp_nmatrix_field::allocate_mdp_nmatrix_field (mdp_lattice & *a*, int *n*, int *i*, int *j*) [inline]

dynamically allocates a field object that at each site as vector of n ixj matrices

6.43.3.2 const mdp_complex& mdp_nmatrix_field::operator() (mdp_site *x*, int *n*, int *i*, int *j*) const [inline]

6.43.3.3 mdp_complex& mdp_nmatrix_field::operator() (mdp_site *x*, int *n*, int *i*, int *j*) [inline]

returns the (i,j) component of the n-th matrix stored at site x

6.43.3.4 mdp_matrix mdp_nmatrix_field::operator() (mdp_site *x*, int *n*) [inline]

returns the n-th matrix stored at site x

Reimplemented from mdp_field< mdp_complex >.

6.43.4 Member Data Documentation

6.43.4.1 uint mdp_nmatrix_field::columns

6.43.4.2 uint mdp_nmatrix_field::imax

6.43.4.3 uint mdp_nmatrix_field::imax2

6.43.4.4 uint mdp_nmatrix_field::matrices

6.43.4.5 uint mdp_nmatrix_field::rows

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_nmatrix_field.h

6.44 mdp_nvector_field Class Reference

field of vectors of vectors (DEPRECATED)

`#include <mdp_nvector_field.h>`Inheritance diagram for mdp_nvector_field::

Public Member Functions

- mdp_nvector_field ()
- mdp_nvector_field (mdp_nvector_field &field)
- mdp_nvector_field (mdp_lattice &a, int n, int i)
- void allocate_mdp_nvector_field (mdp_lattice &a, int n, int i)
- mdp_matrix operator() (mdp_site x, int n)

  returns component i of the vector of objects T stored at site x

- mdp_complex & operator() (mdp_site x, int n, int i)
- const mdp_complex & operator() (mdp_site x, int n, int i) const

Public Attributes

- uint rows
- uint columns
- uint imax
- uint imax2

6.44.1 Detailed Description

field of vectors of vectors (DEPRECATED)

6.44.2 Constructor & Destructor Documentation

6.44.2.1 mdp_nvector_field::mdp_nvector_field () [inline]

6.44.2.2 mdp_nvector_field::mdp_nvector_field (mdp_nvector_field & *field*) [inline]

6.44.2.3 mdp_nvector_field::mdp_nvector_field (mdp_lattice & *a*, int *n*, int *i*) [inline]

6.44.3 Member Function Documentation

6.44.3.1 void mdp_nvector_field::allocate_mdp_nvector_field (mdp_lattice & *a*, int *n*, int *i*) [inline]

6.44.3.2 const mdp_complex& mdp_nvector_field::operator() (mdp_site *x*, int *n*, int *i*) const [inline]

6.44.3.3 mdp_complex& mdp_nvector_field::operator() (mdp_site *x*, int *n*, int *i*) [inline]

6.44.3.4 mdp_matrix mdp_nvector_field::operator() (mdp_site *x*, int *i*) [inline]

returns component i of the vector of objects T stored at site x

Reimplemented from mdp_field< mdp_complex >.

6.44.4 Member Data Documentation

6.44.4.1 uint mdp_nvector_field::columns

6.44.4.2 uint mdp_nvector_field::imax

6.44.4.3 uint mdp_nvector_field::imax2

6.44.4.4 uint mdp_nvector_field::rows

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_nvector_field.h

6.45 mdp_postscript Class Reference

to output and draw in postscript

`#include <mdp_postscript.h>`

Public Types

- enum { BOLD = 10 }

Public Member Functions

- mdp_postscript ()
- mdp_postscript (char filename[ ])
- virtual ~mdp_postscript ()
- FILE * open (char filename[ ])
- void close ()
- void size (float x0, float y0, float x1, float y1)
- void line (float x0, float y0, float x1, float y1)
- void box (float x0, float y0, float x1, float y1, int fill=0)
- void arc (float x0, float y0, float r, float alpha, float beta)
- void circle (float x0, float y0, float r, int fill=0)
- void pen (float size)
- void color (float r, float g, float b)
- void font (const char *text, int size)
- void print (float x0, float y0, char text[ ])

Public Attributes

- FILE * fp
- float X0
- float Y0
- float Z0
- float X1
- float Y1
- float Z1
- float c0
- float c1
- float c2
- float scale
- float alpha

6.45.1 Detailed Description

to output and draw in postscript Example:

```
///    mdp_postscript ps("test.ps");
///    ps.color(0.2,0.2,0.7);
///    ps.line(0,0,  5,5);
///    ps.print(5,5,"a line from (0,0) to here");
///
```

6.45.2 Member Enumeration Documentation

6.45.2.1 anonymous enum

Enumerator:

BOLD

6.45.3 Constructor & Destructor Documentation

6.45.3.1 mdp_postscript::mdp_postscript () [inline]

6.45.3.2 mdp_postscript::mdp_postscript (char *filename*[ ]) [inline]

6.45.3.3 virtual mdp_postscript::~mdp_postscript () [inline, virtual]

6.45.4 Member Function Documentation

6.45.4.1 void mdp_postscript::arc (float *x0*, float *y0*, float *r*, float *alpha*, float *beta*) [inline]

6.45.4.2 void mdp_postscript::box (float *x0*, float *y0*, float *x1*, float *y1*, int *fill* = 0) [inline]

6.45.4.3 void mdp_postscript::circle (float *x0*, float *y0*, float *r*, int *fill* = 0) [inline]

6.45.4.4 void mdp_postscript::close () [inline]

6.45.4.5 void mdp_postscript::color (float *r*, float *g*, float *b*) [inline]

6.45.4.6 void mdp_postscript::font (const char * *text*, int *size*) [inline]

6.45.4.7 void mdp_postscript::line (float *x0*, float *y0*, float *x1*, float *y1*) [inline]

6.45.4.8 FILE* mdp_postscript::open (char *filename*[ ]) [inline]

6.45.4.9 void mdp_postscript::pen (float *size*) [inline]

6.45.4.10 void mdp_postscript::print (float *x0*, float *y0*, char *text*[ ]) [inline]

6.45.4.11 void mdp_postscript::size (float *x0*, float *y0*, float *x1*, float *y1*) [inline]

6.45.5 Member Data Documentation

6.45.5.1 float mdp_postscript::alpha

6.45.5.2 float mdp_postscript::c0

6.45.5.3 float mdp_postscript::c1

6.45.5.4 float mdp_postscript::c2

6.45.5.5 FILE* mdp_postscript::fp

6.45.5.6 float mdp_postscript::scale

6.45.5.7 float mdp_postscript::X0

6.45.5.8 float mdp_postscript::X1

6.45.5.9 float mdp_postscript::Y0

6.45.5.10 float mdp_postscript::Y1

6.45.5.11 float mdp_postscript::Z0

6.45.5.12 float mdp_postscript::Z1

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_postscript.h

6.46 mdp_prng Class Reference

Marsaglia's random number generator (same as UKQCD).

`#include <mdp_prng.h>`

Public Member Functions

- float plain ()

  return a uniform random number in (0,1)

- void initialize (mdp_int ijkl)
- mdp_prng (mdp_int k=0)
- float gaussian (float sigma=1)

  returns a gaussian random number

- double distribution (float(*fp)(float, void *), void *a=0)

  draws a random float in (0,1) from a distribution using accept-reject

- mdp_matrix SU (int n)

  returns a random SU(n) matrix using Cabibbo-Marinari

- void skip (int n)

  skip n numbers from the sequence

6.46.1 Detailed Description

Marsaglia's random number generator (same as UKQCD). You should not instantiate this class because:

- there is a global object mdp_random
- each field "lattice" has a parallel generator "lattice.random(x)" Example:

```
///    // print a uniform number in (0,1)
///    cout << mdp_random.plain() << endl;
///    // print a gaussian number
///    cout << mdp_random.gaussian() << endl;
///    // print a random SU(10) matrix
///    cout << mdp_random.SU(10) << endl;
///
```

6.46.2 Constructor & Destructor Documentation

6.46.2.1 mdp_prng::mdp_prng (mdp_int *k* = 0) [inline]

6.46.3 Member Function Documentation

6.46.3.1 double mdp_prng::distribution (float(*)(float, void *) *fp*, void * *a* = 0) [inline]

draws a random float in (0,1) from a distribution using accept-reject

6.46.3.2 float mdp_prng::gaussian (float *sigma* = 1) [inline]

returns a gaussian random number

6.46.3.3 void mdp_prng::initialize (mdp_int *ijkl*) [inline]

6.46.3.4 float mdp_prng::plain () [inline]

return a uniform random number in (0,1)

6.46.3.5 void mdp_prng::skip (int *n*) [inline]

skip n numbers from the sequence

6.46.3.6 mdp_matrix mdp_prng::SU (int *n*) [inline]

returns a random SU(n) matrix using Cabibbo-Marinari
The documentation for this class was generated from the following file:

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_prng.h

6.47 mdp_prng_sfmt Class Reference

`#include <mdp_prng_sfmt.h>`

Classes

- struct **W128_T**

Public Member Functions

- void initialize (unsigned int seed)
- float plain ()

6.47.1 Member Function Documentation

6.47.1.1 void mdp_prng_sfmt::initialize (unsigned int *seed*) [inline]

6.47.1.2 float mdp_prng_sfmt::plain () [inline]

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_prng_sfmt.h

6.48 mdp_psim Class Reference

Parallel SIMulator used by class mdp_communicator.

`#include <mdp_psim.h>`

Public Member Functions

- mdp_psim (int processCount, string logFileName=".psim.log", int verbatim=0)
- mdp_psim (int argc, char **argv)
- virtual ~mdp_psim ()
- void log (string message, int level=2)
- int id ()
- int nprocs ()
- void setCommTimeout (unsigned int commTimeout)
- template<class T >
  void send (int destProcessID, string dataTag, T &dataToSend)
- template<class T >
  void send (int destProcessID, string dataTag, T *pdataToSend, mdp_int dataSize)
- template<class T >
  void recv (int sourceProcessID, string dataTag, T &dataToReceive)
- template<class T >
  void recv (int sourceProcessID, string dataTag, T *pdataToReceive, mdp_int dataSize)
- template<class T >
  void broadcast (int sourceProcessID, T &data)
- template<class T >
  void broadcast (int sourceProcessID, T *data, int dataSize)
- template<class T >
  vector< T > collect (int dest, T &data)
- template<class T >
  vector< T > combine (T &data)
- void barrier ()
- template<class T >
  T add (T &item)

Static Public Member Functions

- static int parse_argv_nprocs (int argc, char **argv)
- static string parse_argv_logfile (int argc, char **argv)
- static int parse_argv_verbatim (int argc, char **argv)

6.48.1 Detailed Description

Parallel SIMulator used by class mdp_communicator. Attention: under MDP and/or FermiQCD this is already Instantiated inside class mdp_communicator.

Example:

```
/// int main(int argc, char** argv) {
///    mdp_psim node(argc,argv);
///    int a=3, b=0;
///    if(node.id()==0) node.send(1,a);
```

```
///     if(node.id()==1) { node.recv(0,b); cout << b << endl;
///     return 0;
/// }
///
```

Compile with

```
///     g++ [filename] -o a.out
///
```

and run with

```
///     ./a.out -PSIM_NPROCS=2
///
```

Output should be 3.

6.48.2 Constructor & Destructor Documentation

6.48.2.1 mdp_psim::mdp_psim (int *processCount*, string *logFileName* = `".psim.log"`, int *verbatim* = `0`) `[inline]`

6.48.2.2 mdp_psim::mdp_psim (int *argc*, char ** *argv*) `[inline]`

6.48.2.3 virtual mdp_psim::~mdp_psim () `[inline, virtual]`

6.48.3 Member Function Documentation

6.48.3.1 template<class T > T mdp_psim::add (T & *item*) `[inline]`

6.48.3.2 void mdp_psim::barrier () `[inline]`

6.48.3.3 template<class T > void mdp_psim::broadcast (int *sourceProcessID*, T * *data*, int *dataSize*) `[inline]`

6.48.3.4 template<class T > void mdp_psim::broadcast (int *sourceProcessID*, T & *data*) `[inline]`

6.48.3.5 template<class T > vector<T> mdp_psim::collect (int *dest*, T & *data*) `[inline]`

6.48.3.6 template<class T > vector<T> mdp_psim::combine (T & *data*) `[inline]`

6.48.3.7 int mdp_psim::id () `[inline]`

6.48.3.8 void mdp_psim::log (string *message*, int *level* = `2`) `[inline]`

6.48.3.9 int mdp_psim::nprocs () `[inline]`

6.48.3.10 static string mdp_psim::parse_argv_logfile (int *argc*, char ** *argv*) `[inline, static]`

6.48.3.11 static int mdp_psim::parse_argv_nprocs (int *argc*, char ** *argv*) `[inline, static]`

6.48.3.12 static int mdp_psim::parse_argv_verbatim (int *argc*, char ** *argv*) `[inline, static]`

6.48.3.13 template<class T > void mdp_psim::recv (int *sourceProcessID*, string *dataTag*, T * *pdataToReceive*, mdp_int *dataSize*) `[inline]`

6.48.3.14 template<class T > void mdp_psim::recv (int *sourceProcessID*, string *dataTag*, T & *dataToReceive*) `[inline]`

6.48.3.15 template<class T > void mdp_psim::send (int *destProcessID*, string *dataTag*, T * *pdataToSend*, mdp_int *dataSize*) `[inline]`

6.48.3.16 template<class T > void mdp_psim::send (int *destProcessID*, string *dataTag*, T & *dataToSend*) `[inline]`

6.48.3.17 void mdp_psim::setCommTimeout (unsigned int *commTimeout*) `[inline]`

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_psim.h

6.49 MDP_SFMT19937 Class Reference

Classes

- struct **W128_T**

Public Member Functions

- MDP_SFMT19937 (unsigned int seed)
- float uniform ()

6.49.1 Constructor & Destructor Documentation

6.49.1.1 MDP_SFMT19937::MDP_SFMT19937 (unsigned int *seed*) `[inline]`

6.49.2 Member Function Documentation

6.49.2.1 float MDP_SFMT19937::uniform () `[inline]`

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_sfmt.cpp

6.50 mdp_site Class Reference

site object to loop on a lattice

`#include <mdp_site.h>`

Public Member Functions

- mdp_site ()

  value of the mdp_site in local coordinate

- mdp_site (const mdp_lattice &a)
- void on (const mdp_lattice &a)
- mdp_lattice & lattice ()

  returns by reference the lattice the site lives on

- mdp_site (mdp_int i, mdp_lattice *ptr2)
- mdp_site (const mdp_site &x)
- mdp_site operator= (mdp_int i)
- mdp_site operator= (mdp_site x)
- int operator== (mdp_site x)
- int operator!= (mdp_site x)
- void start (int np=0)
- void next ()
- int is_in ()
- int is_here ()
- int parity ()

  returns the parity EVEN or ODD of the site

- int is_in_boundary ()
- mdp_int local_index ()
- mdp_int global_index ()

  returns the global (unique) index of the site

- void set_local (mdp_int idx2)

  sets the site by its local index (dangerous)

- void set_global (mdp_int idx_gl)

  sets the site by its global index

- mdp_site operator+ (int mu)

  returns the site shifted forward in direction mu=(0...ndim-1)

- mdp_site operator- (int mu)

  returns the site shifted backwards in direction mu=(0...ndim-1)

- mdp_site hop (int i, int mu)
- mdp_site operator= (mdp_vector v)

  sets the site to the coordinates stored in vector v

- mdp_site operator+ (mdp_vector v)
- mdp_site operator- (mdp_vector v)
- int operator() (int mu)

  returns mu coordinate of the site

- void operator= (int *x)
- void set (int x0, int x1=0, int x2=0, int x3=0, int x4=0, int x5=0, int x6=0, int x7=0, int x8=0, int x9=0)
- int operator== (int *x)
- int operator!= (int *x)
- int is_equal (int x0, int x1=0, int x2=0, int x3=0, int x4=0, int x5=0, int x6=0, int x7=0, int x8=0, int x9=0)

  checks the site coordinates vs the coordinates passed as args

Public Attributes

- mdp_int idx

  this points to the lattice for this field

Friends

- mdp_int site2binary (mdp_site x)
- int on_which_process (mdp_lattice &a, int x0, int x1, int x2, int x3, int x4, int x5, int x6, int x7, int x8, int x9)

6.50.1 Detailed Description

site object to loop on a lattice Example:

```
///   int box[]={10,10,10};
///   mdp_lattice lattice(3,box);
///   mdp_site x(lattice);
///   forallsites(x) cout << x << endl;
///   if(on_which_process(lattice,1,1,1)==ME) {
///      x.set(1,1,1);
///      cout << lattice.random(x).plain() << endl;
///   }
///
```

6.50.2 Constructor & Destructor Documentation

6.50.2.1 mdp_site::mdp_site () [inline]

value of the mdp_site in local coordinate

6.50.2.2 mdp_site::mdp_site (const mdp_lattice & *a*) [inline]

declares object of class mdp_site living on the lattice passed by reference

6.50.2.3 mdp_site::mdp_site (mdp_int *i*, mdp_lattice * *ptr2*) [inline]

6.50.2.4 mdp_site::mdp_site (const mdp_site & *x*) [inline]

6.50.3 Member Function Documentation

6.50.3.1 mdp_int mdp_site::global_index () [inline]

returns the global (unique) index of the site

6.50.3.2 mdp_site mdp_site::hop (int *i*, int *mu*) [inline]

returns a site shifted i position (backwards if i<0 or forward if i>0) in direction mu=(0...mdim-1)

6.50.3.3 int mdp_site::is_equal (int *x0*, int *x1* = 0, int *x2* = 0, int *x3* = 0, int *x4* = 0, int *x5* = 0, int *x6* = 0, int *x7* = 0, int *x8* = 0, int *x9* = 0) [inline]

checks the site coordinates vs the coordinates passed as args

6.50.3.4 int mdp_site::is_here () [inline]

checks if the site is inside the portion of the lattice stored by the current process or if the site is in a local copy of a remote site

6.50.3.5 int mdp_site::is_in () [inline]

checks if the site is inside the portion of the lattice stored by the current process

6.50.3.6 int mdp_site::is_in_boundary () [inline]

true if the site is stored locally as a copy of a site local in another process

6.50.3.7 mdp_lattice& mdp_site::lattice () [inline]

returns by reference the lattice the site lives on

6.50.3.8 mdp_int mdp_site::local_index () [inline]

returns the local index of the site local index is assigned by the process to the local sites and copies of remote sites. local index is not unique thoughout the lattice.

6.50.3.9 void mdp_site::next () [inline]

6.50.3.10 void mdp_site::on (const mdp_lattice & *a*) [inline]

6.50.3.11 int mdp_site::operator!= (int * *x*) [inline]

6.50.3.12 int mdp_site::operator!= (mdp_site *x*) [inline]

6.50.3.13 int mdp_site::operator() (int *mu*) [inline]

returns mu coordinate of the site

6.50.3.14 mdp_site mdp_site::operator+ (mdp_vector *v*) [inline]

retruns a site similar to the present but each coordinates mu of the site shifted according to v[mu]

6.50.3.15 mdp_site mdp_site::operator+ (int *mu*) [inline]

returns the site shifted forward in direction mu=(0...ndim-1)

6.50.3.16 mdp_site mdp_site::operator- (mdp_vector *v*) [inline]

retruns a site similar to the present but each coordinates mu of the site shifted according to -v[mu]

6.50.3.17 mdp_site mdp_site::operator- (int *mu*) [inline]

returns the site shifted backwards in direction mu=(0...ndim-1)

6.50.3.18 void mdp_site::operator= (int * *x*) [inline]

6.50.3.19 mdp_site mdp_site::operator= (mdp_vector *v*) [inline]

sets the site to the coordinates stored in vector v

6.50.3.20 mdp_site mdp_site::operator= (mdp_site *x*) [inline]

6.50.3.21 mdp_site mdp_site::operator= (mdp_int *i*) [inline]

6.50.3.22 int mdp_site::operator== (int * *x*) [inline]

6.50.3.23 int mdp_site::operator== (mdp_site *x*) [inline]

6.50.3.24 int mdp_site::parity () [inline]

returns the parity EVEN or ODD of the site

6.50.3.25 void mdp_site::set (int *x0*, int *x1* = 0, int *x2* = 0, int *x3* = 0, int *x4* = 0, int *x5* = 0, int *x6* = 0, int *x7* = 0, int *x8* = 0, int *x9* = 0) [inline]

sets the site to a the location spacified by the coordinates and assumes the site is local (or at least a copy)

See also:

on_which_process()

6.50.3.26 void mdp_site::set_global (mdp_int *idx_gl*) [inline]

sets the site by its global index

6.50.3.27 void mdp_site::set_local (mdp_int *idx2*) [inline]

sets the site by its local index (dangerous)

6.50.3.28 void mdp_site::start (int *np* = 0) [inline]

6.50.4 Friends And Related Function Documentation

6.50.4.1 int on_which_process (mdp_lattice & *a*, int *x0*, int *x1*, int *x2*, int *x3*, int *x4*, int *x5*, int *x6*, int *x7*, int *x8*, int *x9*) [friend]

checks which process of the lattice a stores locally the site of coordinates x0,x1,x2,...,x9 to be used before calling mdp_site::set()

checks which process of the lattice a stores locally the site of coordinates x0,x1,x2,...,x9 to be used before calling mdp_site::set() (note: prototyping of friend functions is required by some compilers)

6.50.4.2 mdp_int site2binary (mdp_site *x*) [friend]

converts a site into a binary number to be used only if the site is a vertex of an hypercube centered at the origin. this is used to make staggered mesons

6.50.5 Member Data Documentation

6.50.5.1 mdp_int mdp_site::idx

this points to the lattice for this field

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_site.h

6.51 mdp_vector Class Reference

discerete vectors to navigate on a lattice

`#include <mdp_vector.h>`

Public Member Functions

- mdp_vector ()
- mdp_vector (int x0, int x1=0, int x2=0, int x3=0, int x4=0, int x5=0, int x6=0, int x7=0, int x8=0, int x9=0)

Public Attributes

- int x [10]

6.51.1 Detailed Description

discerete vectors to navigate on a lattice

6.51.2 Constructor & Destructor Documentation

6.51.2.1 mdp_vector::mdp_vector () [inline]

6.51.2.2 mdp_vector::mdp_vector (int *x0*, int *x1* = 0, int *x2* = 0, int *x3* = 0, int *x4* = 0, int *x5* = 0, int *x6* = 0, int *x7* = 0, int *x8* = 0, int *x9* = 0) [inline]

6.51.3 Member Data Documentation

6.51.3.1 int mdp_vector::x[10]

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_vector.h

6.52 mdp_vector_field Class Reference

a field of vectors of complex numbers

`#include <mdp_vector_field.h>`Inheritance diagram for mdp_vector_field::

Public Member Functions

- mdp_vector_field ()
- mdp_vector_field (mdp_vector_field &field)
- mdp_vector_field (mdp_lattice &a, int i)
- void allocate_mdp_vector_field (mdp_lattice &a, int i)
- mdp_matrix operator() (mdp_site x)
- mdp_complex & operator() (mdp_site x, int i)

  returns component i of the vector of objects T stored at site x

- const mdp_complex & operator() (mdp_site x, int i) const

Public Attributes

- int rows
- int columns
- int imax

6.52.1 Detailed Description

a field of vectors of complex numbers Example:

```
///    int box[]={10,10,10};
///    mdp_lattice lattice(3,box);
///    mdp_vector_field h(lattice,10);
///    mdp_site x(lattice);
///    forallsites(x)
///      h(x)=0.0+0.0*I;
///
```

6.52.2 Constructor & Destructor Documentation

6.52.2.1 mdp_vector_field::mdp_vector_field () [inline]

6.52.2.2 mdp_vector_field::mdp_vector_field (mdp_vector_field & *field*) [inline]

6.52.2.3 mdp_vector_field::mdp_vector_field (mdp_lattice & *a*, int *i*) [inline]

6.52.3 Member Function Documentation

6.52.3.1 void mdp_vector_field::allocate_mdp_vector_field (mdp_lattice & *a*, int *i*) [inline]

6.52.3.2 const mdp_complex& mdp_vector_field::operator() (mdp_site *x*, int *i*) const [inline]

6.52.3.3 mdp_complex& mdp_vector_field::operator() (mdp_site *x*, int *i*) [inline]

returns component i of the vector of objects T stored at site x

Reimplemented from mdp_field< mdp_complex >.

6.52.3.4 mdp_matrix mdp_vector_field::operator() (mdp_site *x*) [inline]

6.52.4 Member Data Documentation

6.52.4.1 int mdp_vector_field::columns

6.52.4.2 int mdp_vector_field::imax

6.52.4.3 int mdp_vector_field::rows

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

- /Users/mdipierro/fermiqcd/development/Libraries/mdp_vector_field.h

6.53 MinRes Class Reference

the minimum residure inverter

`#include <fermiqcd_minres_inverter.h>`

Static Public Member Functions

- template<class fieldT , class fieldG >
  static inversion_stats inverter (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_precision=0, int max_steps=2000)

6.53.1 Detailed Description

the minimum residure inverter It inverts mul_Q(psi_out,psi_in,U,coeff) iteratively

Parameters:

psi_out the output field passed by reference

psi_in the input field passed by reference

U the gauge field to be passed to mul_Q

coeff the gauge parameters to be passed to mul_Q

absolute_precision the target absolute precision

relative_precision the target relative precision

max_steps the maximum number of steps

Example:

```
/// gauge_field U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// fermi_field chi(lattice,nc);
/// coefficinets coeff;
/// coeff["kappa"]=1.12;
/// U.load("myfield");
/// psi.load("myfield_psi");
/// default_fermi_inverter=MinRes::inverter<fermi_field,gauge_field>;
/// default_fermi_action=FermiCloverActionSlow::mul_Q;
/// mul_invQ(chi,psi,U,coeff);
/// chi.save("myfield_chi");
///
```

Note that mul_invQ(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)^{-1}\psi$

6.53.2 Member Function Documentation

6.53.2.1 template<class fieldT , class fieldG > static inversion_stats MinRes::inverter (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_minres_inverter.h

6.54 MinResVtk Class Reference

the minimum residure inverter

`#include <fermiqcd_minres_inverter_vtk.h>`

Static Public Member Functions

- template<class fieldT , class fieldG >
  static inversion_stats inverter (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_precision=0, int max_steps=2000)

6.54.1 Detailed Description

the minimum residure inverter It inverts mul_Q(psi_out,psi_in,U,coeff) iteratively

Parameters:

psi_out the output field passed by reference

psi_in the input field passed by reference

U the gauge field to be passed to mul_Q

coeff the gauge parameters to be passed to mul_Q

absolute_precision the target absolute precision

relative_precision the target relative precision

max_steps the maximum number of steps

Example:

```
/// gauge_field U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// fermi_field chi(lattice,nc);
/// coefficinets coeff;
/// coeff["kappa"]=1.12;
/// U.load("myfield");
/// psi.load("myfield_psi");
/// default_fermi_inverter=MinResVtk::inverter<fermi_field,gauge_field>;
/// default_fermi_action=FermiCloverActionSlow::mul_Q;
/// mul_invQ(chi,psi,U,coeff);
/// chi.save("myfield_chi");
///
```

Note that mul_invQ(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)^{-1}\psi$

6.54.2 Member Function Documentation

6.54.2.1 template<class fieldT , class fieldG > static inversion_stats MinResVtk::inverter (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_minres_inverter_vtk.h

6.55 phase_field Class Reference

`#include <fermiqcd_staggered_mesons.h>`Inheritance diagram for phase_field::

Public Member Functions

- phase_field (mdp_lattice &a)
- int component (site x, site y)
- void compute (mdp_matrix GAMMA, mdp_matrix ZETA)

6.55.1 Constructor & Destructor Documentation

6.55.1.1 phase_field::phase_field (mdp_lattice & *a*) [inline]

6.55.2 Member Function Documentation

6.55.2.1 int phase_field::component (site *x*, site *y*) [inline]

6.55.2.2 void phase_field::compute (mdp_matrix *GAMMA*, mdp_matrix *ZETA*) [inline]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_mesons.h

6.56 sdwf_field Class Reference

field for domain wall staggered fermions

`#include <fermiqcd_sdwf_field.h>`Inheritance diagram for sdwf_field::

Public Member Functions

- sdwf_field (mdp_lattice &a, int L5_, int nc_, int nspin_=4)
- sdwf_field (sdwf_field &chi)
- mdp_matrix operator() (site x, int x5)
- mdp_complex & operator() (site x, int x5, int i)
- const mdp_complex & operator() (site x, int x5, int i) const
- void operator= (mdp_complex a)
- mdp_real component (site x, int mu)
- mdp_real eta (site x, int mu)
- mdp_real eps (site x)
- mdp_real type (site x)
- site chiral_shift (site x)
- mdp_real chiral_phase (site x)
- mdp_real chiral_phase2 (site x)

Public Attributes

- int nc
- int ndim
- int nspin
- int L5

6.56.1 Detailed Description

field for domain wall staggered fermions

6.56.2 Constructor & Destructor Documentation

6.56.2.1 sdwf_field::sdwf_field (mdp_lattice & *a*, int *L5_*, int *nc_*, int *nspin_* = 4) [inline]

6.56.2.2 sdwf_field::sdwf_field (sdwf_field & *chi*) [inline]

6.56.3 Member Function Documentation

6.56.3.1 mdp_real sdwf_field::chiral_phase (site *x*) [inline]

6.56.3.2 mdp_real sdwf_field::chiral_phase2 (site *x*) [inline]

6.56.3.3 site sdwf_field::chiral_shift (site *x*) [inline]

6.56.3.4 mdp_real sdwf_field::component (site *x*, int *mu*) [inline]

6.56.3.5 mdp_real sdwf_field::eps (site *x*) [inline]

6.56.3.6 mdp_real sdwf_field::eta (site *x*, int *mu*) [inline]

6.56.3.7 const mdp_complex& sdwf_field::operator() (site *x*, int *x5*, int *i*) const [inline]

6.56.3.8 mdp_complex& sdwf_field::operator() (site *x*, int *x5*, int *i*) [inline]

6.56.3.9 mdp_matrix sdwf_field::operator() (site *x*, int *x5*) [inline]

6.56.3.10 void sdwf_field::operator= (mdp_complex *a*) [inline]

Reimplemented from mdp_field< mdp_complex >.

6.56.3.11 mdp_real sdwf_field::type (site *x*) [inline]

6.56.4 Member Data Documentation

6.56.4.1 int sdwf_field::L5

6.56.4.2 int sdwf_field::nc

6.56.4.3 int sdwf_field::ndim

6.56.4.4 int sdwf_field::nspin

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sdwf_field.h

6.57 SDWFActionSlow Class Reference

domain wall staggered (WORK IN PROGRESS)

`#include <fermiqcd_sdwf_actions.h>`

Static Public Member Functions

- static void mul_Q (sdwf_field &chi_out, sdwf_field &chi_in, gauge_field &U, coefficients &coeff, int parity=EVENODD)

6.57.1 Detailed Description

domain wall staggered (WORK IN PROGRESS)

6.57.2 Member Function Documentation

6.57.2.1 static void SDWFActionSlow::mul_Q (sdwf_field & *chi_out*, sdwf_field & *chi_in*, gauge_field & *U*, coefficients & *coeff*, int *parity* = EVENODD) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sdwf_actions.h

6.58 staggered_field Class Reference

staggered fermionic field

`#include <fermiqcd_staggered_field.h>`Inheritance diagram for staggered_field::

Public Member Functions

- staggered_field (mdp_lattice &a, int nc_, int nspin_=4)
- staggered_field (const staggered_field &chi)
- void operator= (const staggered_field &chi)
- mdp_matrix operator() (site x)
- mdp_complex & operator() (site x, int i)
- const mdp_complex & operator() (site x, int i) const
- void operator= (mdp_complex a)
- mdp_real component (site x, int mu)
- mdp_real eta (site x, int mu)
- mdp_real eps (site x)
- mdp_real type (site x)

Public Attributes

- int nc
- int ndim
- int nspin

6.58.1 Detailed Description

staggered fermionic field Example:

```
/// staggered_field psi(lattice,nc);
/// mdp_site x(lattice);
/// forallsites(x)
///   for(int i=0; i<nc; i++)
///     psi(x,i)=0.0+0.0*I;
///
```

6.58.2 Constructor & Destructor Documentation

6.58.2.1 staggered_field::staggered_field (mdp_lattice & *a*, int *nc_*, int *nspin_* = 4) [inline]

6.58.2.2 staggered_field::staggered_field (const staggered_field & *chi*) [inline]

6.58.3 Member Function Documentation

6.58.3.1 mdp_real staggered_field::component (site *x*, int *mu*) [inline]

6.58.3.2 mdp_real staggered_field::eps (site *x*) [inline]

6.58.3.3 mdp_real staggered_field::eta (site *x*, int *mu*) [inline]

6.58.3.4 const mdp_complex& staggered_field::operator() (site *x*, int *i*) const [inline]

6.58.3.5 mdp_complex& staggered_field::operator() (site *x*, int *i*) [inline]

6.58.3.6 mdp_matrix staggered_field::operator() (site *x*) [inline]

6.58.3.7 void staggered_field::operator= (mdp_complex *a*) [inline]

Reimplemented from mdp_field< mdp_complex >.

6.58.3.8 void staggered_field::operator= (const staggered_field & *chi*) [inline]

Reimplemented from mdp_complex_field.

6.58.3.9 mdp_real staggered_field::type (site *x*) [inline]

6.58.4 Member Data Documentation

6.58.4.1 int staggered_field::nc

6.58.4.2 int staggered_field::ndim

6.58.4.3 int staggered_field::nspin

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_field.h

6.59 staggered_propagator Class Reference

staggared quark propagator

`#include <fermiqcd_staggered_propagator.h>`Inheritance diagram for staggered_-propagator::

Public Member Functions

- staggered_propagator (mdp_lattice &mylattice, int nc_)
- mdp_matrix operator() (site x, int a)
- mdp_complex & operator() (site x, int a, int i, int j)

Public Attributes

- int nc

Friends

- void generate (staggered_propagator &S, gauge_field &U, coefficients &coeff, mdp_real absolute_precision=fermi_inversion_precision, mdp_real relative_precision=0, int max_steps=2000, void(∗smf)(staggered_field &, gauge_field &)=0, int comp=0)

6.59.1 Detailed Description

staggared quark propagator On a (2n) dimensional lattice this makes 3∗(2^n) sources at the vertices of the hypercube at the origin of the lattice and inverts the Staggered/Asqtad action on them.

Example:

```
/// mdp_gauge U(lattice,nc);
/// staggered_propagator S(lattice,nc);
/// mdp_site x(lattice);
/// mdp_site y(lattice);
/// coefficients coeff;
/// coeff["mass"]=1.0;
/// generate(S,U,coeff);
/// for(int i=0; i<(int) pow(2,lattice.ndim); i++) {
///    x=binary2versor(a);
///    cout << "source at:" << x << "\nprop:\n";
///    forallsites(y) cout << S(x,a) << endl;
/// }
///
```

6.59.2 Constructor & Destructor Documentation

6.59.2.1 staggered_propagator::staggered_propagator (mdp_lattice & *mylattice*, int *nc_*) [inline]

6.59.3 Member Function Documentation

6.59.3.1 mdp_complex& staggered_propagator::operator() (site *x*, int *a*, int *i*, int *j*) [inline]

6.59.3.2 mdp_matrix staggered_propagator::operator() (site *x*, int *a*) [inline]

6.59.4 Friends And Related Function Documentation

6.59.4.1 void generate (staggered_propagator & *S*, gauge_field & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = fermi_inversion_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000, void(∗)(staggered_field &, gauge_field &) *smf* = 0, int *comp* = 0) [friend]

6.59.5 Member Data Documentation

6.59.5.1 int staggered_propagator::nc

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_propagator.h

6.60 StaggeredAsqtadActionFast Class Reference

Staggered/Asqtad action.

`#include <fermiqcd_staggered_actions.h>`

Static Public Member Functions

- static void mul_Q (staggered_field &chi_out, staggered_field &chi_in, gauge_field &U, coefficients &coeff, int parity=EVENODD)

6.60.1 Detailed Description

Staggered/Asqtad action. Example:

```
/// gauge_field U(lattice,nc);
/// staggered_field psi(lattice,nc);
/// staggered_field chi(lattice,nc);
/// coefficients coeff;
/// coeff["mass"]=2.0;
/// default_staggered_action=StaggeredAsqtadActionFast::mul_Q;
/// mul_Q(chi,psi,U,coeff);
///
```

6.60.2 Member Function Documentation

6.60.2.1 static void StaggeredAsqtadActionFast::mul_Q (staggered_field & *chi_out*, staggered_field & *chi_in*, gauge_field & *U*, coefficients & *coeff*, int *parity* = EVENODD) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_actions.h

6.61 StaggeredAsqtadActionSlow Class Reference

Staggered/Asqtad action (SLOW: DO NOT USE IN PRODUCTION).

`#include <fermiqcd_staggered_actions.h>`

Static Public Member Functions

- static void mul_Q (staggered_field &chi_out, staggered_field &chi_in, gauge_field &U, coefficients &coeff, int parity=EVENODD)

6.61.1 Detailed Description

Staggered/Asqtad action (SLOW: DO NOT USE IN PRODUCTION). Example:

```
/// gauge_field U(lattice,nc);
/// staggered_field psi(lattice,nc);
/// staggered_field chi(lattice,nc);
/// coefficients coeff;
/// coeff["mass"]=2.0;
/// default_staggered_action=StaggeredAsqtadActionSlow::mul_Q;
/// mul_Q(chi,psi,U,coeff);
///
```

Note that mul_Q(chi,psi,U,coeff) reads $\chi = (\not{D}[U] + m)\psi$

6.61.2 Member Function Documentation

6.61.2.1 static void StaggeredAsqtadActionSlow::mul_Q (staggered_field & *chi_out*, staggered_field & *chi_in*, gauge_field & *U*, coefficients & *coeff*, int *parity* = EVENODD) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_actions.h

6.62 StaggeredBiCGUML Class Reference

MILC staggered UML inverter (optimized bicgstab).

`#include <fermiqcd_staggered_uml_inverter.h>`

Static Public Member Functions

- static inversion_stats inverter (staggered_field &psi_out, staggered_field &psi_in, gauge_field &U, coefficients &coeff, mdp_real absolute_precision=staggered_inversion_precision, mdp_real relative_precision=0, int max_steps=2000)

6.62.1 Detailed Description

MILC staggered UML inverter (optimized bicgstab). The algorithm is taken form hep-lat/9212007 This is best algorithm for staggered fermions

It inverts mul_Q(psi_out,psi_in,U,coeff) iteratively

Parameters:

psi_out the output field passed by reference

psi_in the input field passed by reference

U the gauge field to be passed to mul_Q

coeff the gauge parameters to be passed to mul_Q

absolute_precision the target absolute precision

relative_precision the target relative precision

max_steps the maximum number of steps

Example:

```
/// gauge_field U(lattice,nc);
/// staggered_field psi(lattice,nc);
/// staggered_field chi(lattice,nc);
/// coefficinets coeff;
/// coeff["kappa"]=1.12;
/// U.load("myfield");
/// psi.load("myfield_psi");
/// default_staggered_inverter=StaggeredBiCGUML::inverter;
/// default_staggered_action=StaggeredAsqtadActionFast::mul_Q;
/// mul_invQ(chi,psi,U,coeff);
/// chi.save("myfield_chi");
///
```

6.62.2 Member Function Documentation

6.62.2.1 static inversion_stats StaggeredBiCGUML::inverter (staggered_field & *psi_out*, staggered_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = staggered_inversion_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_uml_inverter.h

6.63 SU_Generators Class Reference

`#include <fermiqcd_su_generators.h>`

Public Member Functions

- SU_Generators (int n)
- mdp_matrix build_matrix (int a)

Public Attributes

- vector< mdp_matrix > lambda
- int n
- int ngenerators

6.63.1 Constructor & Destructor Documentation

6.63.1.1 SU_Generators::SU_Generators (int *n*) [inline]

6.63.2 Member Function Documentation

6.63.2.1 mdp_matrix SU_Generators::build_matrix (int *a*) [inline]

6.63.3 Member Data Documentation

6.63.3.1 vector<mdp_matrix> SU_Generators::lambda

6.63.3.2 int SU_Generators::n

6.63.3.3 int SU_Generators::ngenerators

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_su_generators.h

6.64 WilsonGaugeAction Class Reference

the Wilson Gauge Action

`#include <fermiqcd_gauge_actions.h>`Inheritance diagram for WilsonGaugeAction::

Static Public Member Functions

- static void heatbath_SU2 (mdp_prng &random, mdp_real beta_eff, mdp_complex *a)
- static gauge_stats heatbath (gauge_field &U, coefficients &coeff, int n_iter=1)

6.64.1 Detailed Description

the Wilson Gauge Action Example:

```
///    int ns=2, steps=10;
///    gauge_field U(lattice,nc);
///    coefficients gauge;
///    U.load("myfield.0000");
///    gauge["beta"]=6.0;
///    ImprovedGaugeAction::heatbath(U,gauge,steps);
///    U.save("myfield.0001");
///
```

6.64.2 Member Function Documentation

6.64.2.1 static gauge_stats WilsonGaugeAction::heatbath (gauge_field & *U*, coefficients & *coeff*, int *n_iter* = 1) [inline, static]

6.64.2.2 static void WilsonGaugeAction::heatbath_SU2 (mdp_prng & *random*, mdp_real *beta_eff*, mdp_complex * *a*) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_actions.h

6.65 WupperthalSmearing Class Reference

wupperthal smearing algotihm

`#include <fermiqcd_fermi_smearing.h>`

Static Public Member Functions

- static void smear (fermi_field &psi, gauge_field &U, coefficients &coeff)

6.65.1 Detailed Description

wupperthal smearing algotihm Example:

```
/// gauge_field U(lattice,nc);
/// fermi_field psi(lattice,nc);
/// coefficient smear;
/// smear["factor"]=2;
/// smear["steps"]=10;
/// WupperthalSmearing::smear(psi,U,spear);
///
```

6.65.2 Member Function Documentation

6.65.2.1 static void WupperthalSmearing::smear (fermi_field & *psi*, gauge_field & *U*, coefficients & *coeff*) [inline, static]

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

- /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_fermi_smearing.h

Chapter 7

File Documentation

7.1 /Users/mdipierro/fermiqcd/development/Libraries/average_plaquette.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.1.1 Function Documentation

7.1.1.1 int main (int *argc*, char ** *argv*)

7.2 /Users/mdipierro/fermiqcd/development/Libraries/check_cold.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.2.1 Function Documentation

7.2.1.1 int main (int *argc*, char ** *argv*)

7.3 /Users/mdipierro/fermiqcd/development/Libraries/cleanup_cpp.py File Reference

Namespaces

- namespace cleanup_cpp

Functions

- def cleanup_cpp::cleanup_cpp

7.4 /Users/mdipierro/fermiqcd/development/Libraries/cool_and_topological.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- void test_gauge (int nt, int nx, char *filename)
- int main (int argc, char **argv)

7.4.1 Function Documentation

7.4.1.1 int main (int *argc*, char ** *argv*)

7.4.1.2 void test_gauge (int *nt*, int *nx*, char * *filename*)

7.5 /Users/mdipierro/fermiqcd/development/Libraries/cool_and_-topological_step_by_step.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- void test_gauge (int nt, int nx, char *filename)
- int main (int argc, char **argv)

7.5.1 Function Documentation

7.5.1.1 int main (int *argc*, char ** *argv*)

7.5.1.2 void test_gauge (int *nt*, int *nx*, char * *filename*)

7.6 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd.h File Reference

```
#include "mdp.h"
#include "fermiqcd_su_generators.h"
#include "fermiqcd_global_vars.h"
#include "fermiqcd_gamma_matrices.h"
#include "fermiqcd_default_parameters.h"
#include "fermiqcd_check_differences.h"
#include "fermiqcd_set_random.h"
#include "fermiqcd_coefficients.h"
#include "fermiqcd_sse_su3.h"
#include "fermiqcd_gauge_field.h"
#include "fermiqcd_gauge_routines.h"
#include "fermiqcd_gauge_actions.h"
#include "fermiqcd_gauge_algorithms.h"
#include "fermiqcd_gauge_fixing.h"
#include "fermiqcd_topological_charge.h"
#include "fermiqcd_minres_inverter.h"
#include "fermiqcd_cg_inverter.h"
#include "fermiqcd_bicgstab_inverter.h"
#include "fermiqcd_fermi_field.h"
#include "fermiqcd_fermi_actions.h"
#include "fermiqcd_fermi_actions_sse2.h"
#include "fermiqcd_fermi_algorithms.h"
#include "fermiqcd_fermi_propagator.h"
#include "fermiqcd_fermi_rotation.h"
#include "fermiqcd_fermi_smearing.h"
#include "fermiqcd_lanczos.h"
#include "fermiqcd_staggered_field.h"
#include "fermiqcd_staggered_actions.h"
#include "fermiqcd_staggered_actions_sse2.h"
#include "fermiqcd_staggered_algorithms.h"
#include "fermiqcd_staggered_uml_inverter.h"
#include "fermiqcd_staggered_propagator.h"
#include "fermiqcd_staggered_mesons.h"
#include "fermiqcd_dwfermi_field.h"
```

```
#include "fermiqcd_dwfermi_actions.h"
#include "fermiqcd_dwfermi_algorithms.h"
#include "fermiqcd_hmc.h"
#include "fermiqcd_instanton4d.h"
```

7.6.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Main header file for FermiQCD libraries

Created with support from the US Department of Energy

7.7 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-bicgstab_inverter.h File Reference

Classes

- class BiCGStab

  the stabilized biconjugate inverter

Functions

- template<class fieldT , class fieldG >
  inversion_stats BiConjugateGradientStabilizedInverter (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_precision=0, int max_steps=2000)

7.7.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains the stabilized biconjugate inverter From hep-lat/9404013 (by A. Frommer et al.)

Distributed under GPL2 License

Created with support from the US Department of Energy

7.7.2 Function Documentation

7.7.2.1 template<class fieldT , class fieldG > inversion_stats BiConjugateGradientStabilizedInverter (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline]

7.8 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-bicgstab_inverter_vtk.h File Reference

Classes

- class BiCGStabVtk

  the stabilized biconjugate inverter

Functions

- template<class fieldT , class fieldG >
  inversion_stats BiConjugateGradientStabilizedInverterVtk (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_-precision=0, int max_steps=2000)

7.8.1 Function Documentation

7.8.1.1 template<class fieldT , class fieldG > inversion_stats BiConjugateGradientStabilizedInverterVtk (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline]

7.9 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-cg_inverter.h File Reference

Classes

- class CG2

  the conjugate gradient inverter

7.9.1 Detailed Description

Version:

5-8-2007

Author:

Joseph Schneible <>

7.10 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_check_differences.h File Reference

Functions

- float check_differences (mdp_field< mdp_complex > &chi, mdp_field< mdp_complex > &psi)

7.10.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Constants and parameters used by FermiQCD

Distributed under GPL2 License

Created with support from the US Department of Energy

7.10.2 Function Documentation

7.10.2.1 float check_differences (mdp_field< mdp_complex > & *chi*, mdp_field< mdp_complex > & *psi*)

7.11 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-coefficients.h File Reference

Classes

- class coefficients

  container for action parameters

Functions

- void dagger (coefficients &coeff)
- ostream & operator<< (ostream &os, const coefficients &coeff)

7.11.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Container for action parameters

Distributed under GPL2 License

Created with support from the US Department of Energy

7.11.2 Function Documentation

7.11.2.1 void dagger (coefficients & *coeff*)

7.11.2.2 ostream& operator<< (ostream & *os*, const coefficients & *coeff*)

7.12 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_default_parameters.h File Reference

Defines

- #define TIME 0
- #define SPACE_X 1
- #define SPACE_Y 2
- #define SPACE_Z 3
- #define DAGGER -1

Variables

- mdp_real fermi_inversion_precision = 1e-6
- mdp_real staggered_inversion_precision = 1e-6
- mdp_real dwfermi_inversion_precision = 1e-6
- mdp_real sdwf_inversion_precision = 1e-6
- bool BiCGStabRestart = false

  Set this to true to run BuCGStab with restart.

7.12.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Constants and parameters used by FermiQCD

Created with support from the US Department of Energy

7.12.2 Define Documentation

7.12.2.1 #define DAGGER -1

7.12.2.2 #define SPACE_X 1

7.12.2.3 #define SPACE_Y 2

7.12.2.4 #define SPACE_Z 3

7.12.2.5 #define TIME 0

7.12.3 Variable Documentation

7.12.3.1 bool BiCGStabRestart = false

Set this to true to run BuCGStab with restart.

7.12.3.2 mdp_real dwfermi_inversion_precision = 1e-6

7.12.3.3 mdp_real fermi_inversion_precision = 1e-6

7.12.3.4 mdp_real sdwf_inversion_precision = 1e-6

7.12.3.5 mdp_real staggered_inversion_precision = 1e-6

7.13 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_dwfermi_actions.h File Reference

Classes

- class DWFermiActionSlow
  domain wall action (SORRY THIS IS SLOW)
- class DWFermiActionFast
  domain wall action fast

7.13.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Actions for Domain Wall fermions

Created with support from the US Department of Energy

7.14 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-dwfermi_actions_sse2.h File Reference

7.15 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_dwfermi_algorithms.h File Reference

Functions

- void project (fermi_field &psi, dwfermi_field &chi)

  Projects a domain wall fermion (chi) into a wilson fermion (psi).

- void project (dwfermi_field &chi, fermi_field &psi)

  Projects a will fermion (psi) into a domain wall fermion (chi).

- void mul_Q (dwfermi_field &psi_out, dwfermi_field &psi_in, gauge_field &U, coefficients &coeff)

  Executes the current dwfermi action.

- inversion_stats mul_invQ (dwfermi_field &psi_out, dwfermi_field &psi_in, gauge_field &U, coefficients &coeff, mdp_real absolute_precision=dwfermi_inversion_precision, mdp_real relative_precision=0, int max_steps=2000)

  Execute the default dwfermi inverter.

Variables

- void(∗ default_dwfermi_action)(dwfermi_field &, dwfermi_field &, gauge_field &, coefficients &) = DWFermiActionFast::mul_Q

  Pointer to the current dwfermi action.

- inversion_stats(∗ default_dwfermi_inverter)(dwfermi_field &, dwfermi_field &, gauge_field &, coefficients &, mdp_real, mdp_real, int)

  Pointer to the current dwfermi inverter.

7.15.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

More stuff for domain wall fermions

Created with support from the US Department of Energy

7.15.2 Function Documentation

7.15.2.1 inversion_stats mul_invQ (dwfermi_field & *psi_out*, dwfermi_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = dwfermi_inversion_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000)

Execute the default dwfermi inverter.

7.15.2.2 void mul_Q (dwfermi_field & *psi_out*, dwfermi_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*)

Executes the current dwfermi action.

7.15.2.3 void project (dwfermi_field & *chi*, fermi_field & *psi*)

Projects a will fermion (psi) into a domain wall fermion (chi).

7.15.2.4 void project (fermi_field & *psi*, dwfermi_field & *chi*)

Projects a domain wall fermion (chi) into a wilson fermion (psi).

7.15.3 Variable Documentation

7.15.3.1 void(∗ default_dwfermi_action)(dwfermi_field &, dwfermi_field &, gauge_field &, coefficients &) = DWFermiActionFast::mul_Q

Pointer to the current dwfermi action.

7.15.3.2 inversion_stats(∗ default_dwfermi_inverter)(dwfermi_field &, dwfermi_field &, gauge_field &, coefficients &, mdp_real, mdp_real, int)

Initial value:

```
&(MinRes::inverter<dwfermi_field,gauge_field>)
```

Pointer to the current dwfermi inverter.

7.16 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-dwfermi_field.h File Reference

Classes

- class dwfermi_field
  domain wall fermionic field

7.16.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains the class dwfermi_field fro domain wall fermions

Created with support from the US Department of Energy

7.17 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermi_actions.h File Reference

Classes

- class FermiCloverActionSlow

  Wilson/Clover action (SLOW: DO NOT USE IN PRODUCTION).

- class FermiCloverActionFast

  Wilson/Clover action.

7.17.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Basic actions for Wilson Fermions

Created with support from the US Department of Energy

7.18 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermi_actions_sse2.h File Reference

7.18.1 Detailed Description

Version:

2009-12-21

Author:

Martin Luescher and Massimo Di Pierro <mdipierro@cs.depaul.edu>

Basic actions for Wilson Fermions optimized in assembler

7.19 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermi_algorithms.h File Reference

Functions

- void multiply_by_gamma5 (fermi_field &r, fermi_field &s)

  *r(x,alpha,i) = Gamma5(alpha,beta) * s(x,beta,i)*

- void mul_Q (fermi_field &psi_out, fermi_field &psi_in, gauge_field &U, coefficients &coeff, int parity=EVENODD)

  Calls the current Wilson/Clover action.

- inversion_stats mul_invQ (fermi_field &psi_out, fermi_field &psi_in, gauge_field &U, coefficients &coeff, mdp_real absolute_precision=fermi_inversion_precision, mdp_real relative_precision=0, int max_steps=2000)

  Executes the current Wilson/Clover inverter.

- mdp_real check_inversion (fermi_field &phi, gauge_field &U, coefficients &coeff)

  Checks that inversion is working.

Variables

- void(* default_fermi_action)(fermi_field &, fermi_field &, gauge_field &, coefficients &, int) = FermiCloverActionFast::mul_Q

  Pointer to the current Wilson/Clover action.

- inversion_stats(* default_fermi_inverter)(fermi_field &, fermi_field &, gauge_field &, coefficients &, mdp_real, mdp_real, int) = &(MinRes::inverter<fermi_field,gauge_field>)

  Pointer to the current Wilson/Clover inverter.

7.19.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Various algorithms for Wilson fermions

Distributed under GPL2 License

Created with support from the US Department of Energy

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

class for building a single instanton gauge configuration

Created with support from the US Department of Energy

7.19.2 Function Documentation

7.19.2.1 mdp_real check_inversion (fermi_field & *phi*, gauge_field & *U*, coefficients & *coeff*)

Checks that inversion is working.

7.19.2.2 inversion_stats mul_invQ (fermi_field & *psi_out*, fermi_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = fermi_inversion_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000)

Executes the current Wilson/Clover inverter.

7.19.2.3 void mul_Q (fermi_field & *psi_out*, fermi_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, int *parity* = EVENODD)

Calls the current Wilson/Clover action.

7.19.2.4 void multiply_by_gamma5 (fermi_field & *r*, fermi_field & *s*)

r(x,alpha,i) = Gamma5(alpha,beta) * s(x,beta,i)

7.19.3 Variable Documentation

7.19.3.1 void(* default_fermi_action)(fermi_field &, fermi_field &, gauge_field &, coefficients &, int) = FermiCloverActionFast::mul_Q

Pointer to the current Wilson/Clover action.

7.19.3.2 inversion_stats(* default_fermi_inverter)(fermi_field &, fermi_field &, gauge_field &, coefficients &, mdp_real, mdp_real, int) = &(MinRes::inverter<fermi_field,gauge_field>)

Pointer to the current Wilson/Clover inverter.

7.20 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermi_field.h File Reference

Classes

- class fermi_field

  wilson fermionic field

Functions

- void print_fermi_field (fermi_field &psi)

7.20.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains the class fermi_field

Distributed under GPL2 License

Created with support from the US Department of Energy

7.20.2 Function Documentation

7.20.2.1 void print_fermi_field (fermi_field & *psi*)

7.21 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermi_propagator.h File Reference

Classes

- class fermi_propagator

  a Wilson/Clover quark propagator (all 12 components)

Functions

- void print_propagator (fermi_propagator &S)

7.21.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class fermi_propagator

Distributed under GPL2 License

Created with support from the US Department of Energy

7.21.2 Function Documentation

7.21.2.1 void print_propagator (fermi_propagator & *S*)

7.22 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermi_rotation.h File Reference

Functions

- void rotate_field (fermi_field &psi, gauge_field &U, coefficients &coeff)

7.22.1 Function Documentation

7.22.1.1 void rotate_field (fermi_field & *psi*, gauge_field & *U*, coefficients & *coeff*)

7.23 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermi_smearing.h File Reference

Classes

- class WupperthalSmearing

  wupperthal smearing algotihm

Functions

- void smearSink (fermi_propagator &S, gauge_field &U, void(∗smf)(fermi_field &, gauge_field &, coefficients &), coefficients &coeff)

  smears a propagator

7.23.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Smearing algorithms

Distributed under GPL2 License

Created with support from the US Department of Energy

7.23.2 Function Documentation

7.23.2.1 void smearSink (fermi_propagator & *S*, gauge_field & *U*, void(∗)(fermi_field &, gauge_field &, coefficients &) *smf*, coefficients & *coeff*)

smears a propagator

7.24 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermilab_action.h File Reference

7.24.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Fermilab action with all dimension 5 and 6 terms (requires SSE)

Created with support from the US Department of Energy

7.25 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-fermilab_coefficients.h File Reference

Functions

- mdp_matrix M (4, 4)
- for (a=0;a< 4;a++) for(b=0

Variables

- mdp_matrix SE [4]
- mdp_matrix SB [4]
- M = 1.0+(2.0∗42∗c3+2.0∗18∗c4)∗kappat+Gamma[0]∗2.0∗kappat∗(mac1-2.0∗mac2+6.0∗mac4)+Gamma1∗2.0∗kappat∗mac5

7.25.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Included by fermiqcd_fermilab_action.h

Distributed under GPL2 License

Created with support from the US Department of Energy

7.25.2 Function Documentation

7.25.2.1 for () [pure virtual]

7.25.2.2 mdp_matrix M (4, 4)

7.25.3 Variable Documentation

7.25.3.1 M = 1.0+(2.0∗42∗c3+2.0∗18∗c4)∗kappat+Gamma[0]∗2.0∗kappat∗(mac1-2.0∗mac2+6.0∗mac4)+Gamma1∗2.0∗kappat∗mac5

7.25.3.2 mdp_matrix SB[4]

7.25.3.3 mdp_matrix SE[4]

7.26 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-ffts.h File Reference

Functions

- mdp_int i2pow (mdp_int n)
- void dft (mdp_complex *fft_f, mdp_complex *f, mdp_int n, double sign, mdp_int offset=0, mdp_int coeff=1)
- void fermi_field_fft (int t, fermi_field &psi_out, fermi_field &psi_in, int sign)
- void fermi_field_fft (fermi_field &psi_out, fermi_field &psi_in, int sign)

7.26.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Discrete Fourier stransform (not FFT quote yet)

Distributed under GPL2 License

Created with support from the US Department of Energy

7.26.2 Function Documentation

7.26.2.1 void dft (mdp_complex * *fft_f*, mdp_complex * *f*, mdp_int *n*, double *sign*, mdp_int *offset* = 0, mdp_int *coeff* = 1)

7.26.2.2 void fermi_field_fft (fermi_field & *psi_out*, fermi_field & *psi_in*, int *sign*)

7.26.2.3 void fermi_field_fft (int *t*, fermi_field & *psi_out*, fermi_field & *psi_in*, int *sign*)

7.26.2.4 mdp_int i2pow (mdp_int *n*) [inline]

7.27 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-gamma_matrices.h File Reference

Defines

- #define GAMMA_MATRICES

Functions

- void define_base_matrices (string convention="FERMILAB")

  define convetion for Gamma matrices and bases of SU(2) and SU(3)

Variables

- mdp_complex Gamma_val [4][4]
- mdp_complex Sigma_val [4][4][4]
- int Gamma_idx [4][4]
- int Sigma_idx [4][4][4]
- mdp_complex Gamma5_val [4]
- mdp_complex GammaxGamma5_val [4][4]
- int Gamma5_idx [4]
- int GammaxGamma5_idx [4][4]
- mdp_complex Gamma_valr [4][4]
- mdp_complex Sigma_valr [4][4][4]
- int Gamma_idxr [4][4]
- int Sigma_idxr [4][4][4]
- mdp_complex Gamma5_valr [4]
- mdp_complex GammaxGamma5_valr [4][4]
- int Gamma5_idxr [4]
- int GammaxGamma5_idxr [4][4]
- int G16_idx [16][4]
- mdp_complex G16_val [16][4]
- mdp_matrix Gamma [4]
- mdp_matrix Gamma1
- mdp_matrix Gamma5
- mdp_matrix Pleft
- mdp_matrix Pright
- mdp_matrix Lambda [9]
- mdp_matrix Sigma [4][4]
- mdp_matrix sigma [4]

7.27.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Declares:

- GammaI = 1
- Gamma[mu] = γ^μ
- Gamma5 = γ^5
- Sigma[mu][nu] = $\sigma^{\mu\nu}$
- sigma[i] = σ^i (generators SU(2))
- Lambda[i] = λ^i (generators SU(3))

Distributed under GPL2 License

Created with support from the US Department of Energy

7.27.2 Define Documentation

7.27.2.1 #define GAMMA_MATRICES

7.27.3 Function Documentation

7.27.3.1 void define_base_matrices (string *convention* = "FERMILAB")

define convetion for Gamma matrices and bases of SU(2) and SU(3) At the beginning of any FermiQCD program you MUST call

```
/// define_base_matrices("FERMILAB");
///
```

Possible convections are:

- "FERMILAB" (ok for lattice qcd)
- "MILC" (ok for lattice qcd)
- "UKQCD" (ok for lattice qcd)
- "Minkowsy-Dirac" (not ok for lattice qcd)
- "Minkowsy-Chiral" (not ok for lattice qcd) Convention can be changed within the program.

7.27.4 Variable Documentation

7.27.4.1 int G16_idx[16][4]

7.27.4.2 mdp_complex G16_val[16][4]

7.27.4.3 mdp_matrix Gamma[4]

7.27.4.4 mdp_matrix Gamma1

7.27.4.5 mdp_matrix Gamma5

7.27.4.6 int Gamma5_idx[4]

7.27.4.7 int Gamma5_idxr[4]

7.27.4.8 mdp_complex Gamma5_val[4]

7.27.4.9 mdp_complex Gamma5_valr[4]

7.27.4.10 int Gamma_idx[4][4]

7.27.4.11 int Gamma_idxr[4][4]

7.27.4.12 mdp_complex Gamma_val[4][4]

7.27.4.13 mdp_complex Gamma_valr[4][4]

7.27.4.14 int GammaxGamma5_idx[4][4]

7.27.4.15 int GammaxGamma5_idxr[4][4]

7.27.4.16 mdp_complex GammaxGamma5_val[4][4]

7.27.4.17 mdp_complex GammaxGamma5_valr[4][4]

7.27.4.18 mdp_matrix Lambda[9]

7.27.4.19 mdp_matrix Pleft

7.27.4.20 mdp_matrix Pright

7.27.4.21 mdp_matrix sigma[4]

7.27.4.22 mdp_matrix Sigma[4][4]

7.27.4.23 int Sigma_idx[4][4][4]

7.27.4.24 int Sigma_idxr[4][4][4]

7.27.4.25 mdp_complex Sigma_val[4][4][4]

7.27.4.26 mdp_complex Sigma_valr[4][4][4]

7.28 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_actions.h File Reference

Classes

- class gauge_stats
  (unused)
- class WilsonGaugeAction
  the Wilson Gauge Action
- class ImprovedGaugeAction
  the $O(a^2)$ Improved Gauge Action

7.28.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

All simple gauge actions

Created with support from the US Department of Energy

7.29 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-gauge_actions_sse2.h File Reference

Classes

- class ImprovedGaugeActionSSE2
  the $O(a^2)$ Improved Gauge Action for SU3 with SSE2 and double precision (UNTESTED)

7.29.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

All simple gauge actions

Distributed under GPL2 License

Created with support from the US Department of Energy

7.30 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-gauge_algorithms.h File Reference

Functions

- void set_cold (gauge_field &U)

  make a cold gauge configuration

- void set_hot (gauge_field &U)

  Make a hot gauge configuration.

- void check_unitarity (gauge_field &U, double precision=PRECISION)

  Check that gauge field is unitary within precision.

- mdp_real average_plaquette (gauge_field &U, int mu, int nu)

  Compute average plaquette on plane mu-nu.

- mdp_real average_plaquette (gauge_field &U)

  Compute average plaquette (all planes).

- void compute_em_field (gauge_field &U)

  Given a field U compute the chromo-eletro-magntic field U.em.

- void compute_long_links (gauge_field &U, gauge_field &V, int length=2)
- void set_antiperiodic_phases (gauge_field &U, int mu=0, int check=true)
- mdp_matrix project_SU (mdp_matrix M, int nstep=1)
- mdp_complex average_path (gauge_field &U, int length, int d[ ][2])
- mdp_matrix build_path (gauge_field &U, site x, int length, int d[ ][2])
- void copy_path (int length, int d[ ][2], int c[ ][2])
- void invert_path (int mu, int length, int d[ ][2])
- void rotate_path (int angle, int mu, int nu, int length, int d[ ][2])

7.30.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Various stuff for gauge field

Created with support from the US Department of Energy

7.30.2 Function Documentation

7.30.2.1 mdp_complex average_path (gauge_field & *U*, int *length*, int *d*[ ][2])

Takes a field U and path d of length and compute the average of the path on the entire lattice. Assumes computation can be done locally for each site

Example:

```
///   int mu=0, nu=1;
///   gauge_field U(lattice,nc);
///   int d[][2]={{+1,mu},{+1,nu},{-1,mu},{-1,nu}}
///   mdp << "plaquette=" << average_path(U,4,d) << endl;
///
```

7.30.2.2 mdp_real average_plaquette (gauge_field & *U*)

Compute average plaquette (all planes).

7.30.2.3 mdp_real average_plaquette (gauge_field & *U*, int *mu*, int *nu*)

Compute average plaquette on plane mu-nu.

7.30.2.4 mdp_matrix build_path (gauge_field & *U*, site *x*, int *length*, int *d*[ ][2])

Takes a field U, a site x, a path d of length and compute the product of links amdp_int the path starting at x. Assumes computation can be done locally for each site

Example:

```
///   int mu=0, nu=1;
///   gauge_field U(lattice,nc);
///   int d[][2]={{+1,mu},{+1,nu},{-1,mu},{-1,nu}}
///   forallsites(x)
///      cout << "plaquette(x)=" << average_path(U,x,4,d) << endl;
///
```

7.30.2.5 void check_unitarity (gauge_field & *U*, double *precision* = PRECISION)

Check that gauge field is unitary within precision.

7.30.2.6 void compute_em_field (gauge_field & *U*)

Given a field U compute the chromo-eletro-magntic field U.em.

7.30.2.7 void compute_long_links (gauge_field & *U*, gauge_field & *V*, int *length* = 2)

For use with asqtad staggered action Given field V makes a field U.long_links where (if length==2)

```
/// U.long_links(x,mu)=V(x,mu)*V(x+mu,mu);
///
```

or (if length==3)

```
/// U.long_links(x,mu)=V(x,mu)*V(x+mu,mu)*V((x+mu)+mu,mu);
///
```

7.30.2.8 void copy_path (int *length*, int *d*[ ][2], int *c*[ ][2])

7.30.2.9 void invert_path (int *mu*, int *length*, int *d*[ ][2])

7.30.2.10 mdp_matrix project_SU (mdp_matrix *M*, int *nstep* = 1)

takes a matrix M, performs a Cabibbo-Marinari cooling and returns the projected matrix

7.30.2.11 void rotate_path (int *angle*, int *mu*, int *nu*, int *length*, int *d*[ ][2])

7.30.2.12 void set_antiperiodic_phases (gauge_field & *U*, int *mu* = 0, int *check* = true)

To set antiperiodic boundary conditions on in direction mu

```
///    gauge_field U(lattice,nc);
///    // do heatbath on U
///    set_antiperiodic_phases(U,mu,true);
///    // use quarks (will have antiperiodic boundary conditions)
///    set_antiperiodic_phases(U,mu,false);
///
```

7.30.2.13 void set_cold (gauge_field & *U*)

make a cold gauge configuration

7.30.2.14 void set_hot (gauge_field & *U*)

Make a hot gauge configuration.

7.31 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-gauge_field.h File Reference

Classes

- class em_field

  the chromo-electr-magnetic field for any SU(n)

- class gauge_field

  the gauge field for any SU(n)

7.31.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class gauge_field

Created with support from the US Department of Energy

7.32 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_gauge_fixing.h File Reference

Classes

- class gaugefixing_stats

  Structure for gaugefixing stats.

- class GaugeFixing

  the main gaugefixing algorithm

7.32.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Gauge fixing stuff

Created with support from the US Department of Energy

7.33 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-gauge_routines.h File Reference

Functions

- mdp_matrix staple (gauge_field &U, register site x, int mu, int s1, int nu)
- mdp_matrix staple (gauge_field &U, register site x, int mu)
- mdp_matrix staple_H (gauge_field &U, register site x, int mu, int s1, int nu)
- mdp_matrix staple_H (gauge_field &U, register site x, int mu)
- mdp_matrix staple_0i_H (gauge_field &U, register site x, int mu)
- mdp_matrix staple_ij_H (gauge_field &U, register site x, int mu)
- mdp_matrix plaquette (gauge_field &U, site x, int mu, int nu)

7.33.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Various gauge multiplication routines (some call SSE/SSE2 macors)

Distributed under GPL2 License

Created with support from the US Department of Energy

7.33.2 Function Documentation

7.33.2.1 mdp_matrix plaquette (gauge_field & *U*, site *x*, int *mu*, int *nu*) [inline]

7.33.2.2 mdp_matrix staple (gauge_field & *U*, register site *x*, int *mu*) [inline]

7.33.2.3 mdp_matrix staple (gauge_field & *U*, register site *x*, int *mu*, int *s1*, int *nu*) [inline]

7.33.2.4 mdp_matrix staple_0i_H (gauge_field & *U*, register site *x*, int *mu*) [inline]

7.33.2.5 mdp_matrix staple_H (gauge_field & *U*, register site *x*, int *mu*) [inline]

7.33.2.6 mdp_matrix staple_H (gauge_field & *U*, register site *x*, int *mu*, int *s1*, int *nu*) [inline]

7.33.2.7 mdp_matrix staple_ij_H (gauge_field & *U*, register site *x*, int *mu*) [inline]

7.34 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-global_vars.h File Reference

Variables

- bool shutup = false

  Do or do not dump output to stdout.

- bool shutup_loops = false

  Do or do not dump output to stdout in loops.

7.34.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Main header file for FermiQCD libraries

Created with support from the US Department of Energy

7.34.2 Variable Documentation

7.34.2.1 bool shutup = false

Do or do not dump output to stdout.

7.34.2.2 bool shutup_loops = false

Do or do not dump output to stdout in loops.

7.35 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-hmc.h File Reference

Classes

- class HMC< GaugeClass, FermiClass >

7.36 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_ildg_gauge_reader.h File Reference

Functions

- void ildg_gauge_reader (gauge_field &U, filename, header_bytes=0, precision=16)

7.36.1 Function Documentation

7.36.1.1 void ildg_gauge_reader (gauge_field & *U*, filename, header_bytes = 0, precision = 16)

7.37 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-instanton4d.h File Reference

Classes

- class Instanton4D

7.38 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_lanczos.h File Reference

Classes

- class Lanczos< fieldT >

  Lanczos algorithms.

7.38.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Lanczos routine

Distributed under GPL2 License

Created with support from the US Department of Energy

7.39 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-MILC_IO.h File Reference

Functions

- bool milc_read_as_float_noswitch (FILE *fp, void *data, mdp_int psize, mdp_int header_size, mdp_int position, const mdp_lattice &lattice)
- bool milc_read_as_float_switch (FILE *fp, void *data, mdp_int psize, mdp_int header_size, mdp_int position, const mdp_lattice &lattice)
- bool load_milc (gauge_field &U, string filename, mdp_int max_buffer_size=128, int processIO=0)

7.39.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Functions to read a MILC gauge configuration without conversion

Distributed under GPL2 License

Created with support from the US Department of Energy

7.39.2 Function Documentation

7.39.2.1 bool load_milc (gauge_field & *U*, string *filename*, mdp_int *max_buffer_size* = 128, int *processIO* = 0)

7.39.2.2 bool milc_read_as_float_noswitch (FILE * *fp*, void * *data*, mdp_int *psize*, mdp_int *header_size*, mdp_int *position*, const mdp_lattice & *lattice*)

7.39.2.3 bool milc_read_as_float_switch (FILE * *fp*, void * *data*, mdp_int *psize*, mdp_int *header_size*, mdp_int *position*, const mdp_lattice & *lattice*)

7.40 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_minres_inverter.h File Reference

Classes

- class inversion_stats

  structure for inverstion stats

- class MinRes

  the minimum residure inverter

Functions

- template<class fieldT , class fieldG >
  inversion_stats MinimumResidueInverter (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_precision=0, int max_steps=2000)

7.40.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains the minimum residue inverter

Distributed under GPL2 License

Created with support from the US Department of Energy

7.40.2 Function Documentation

7.40.2.1 template<class fieldT , class fieldG > inversion_stats MinimumResidueInverter (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline]

7.41 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_minres_inverter_vtk.h File Reference

Classes

- class MinResVtk

  the minimum residure inverter

Functions

- template<class fieldT , class fieldG >
  inversion_stats MinimumResidueInverterVtk (fieldT &psi_out, fieldT &psi_in, fieldG &U, coefficients &coeff, mdp_real absolute_precision=mdp_precision, mdp_real relative_precision=0, int max_steps=2000)

7.41.1 Function Documentation

7.41.1.1 template<class fieldT , class fieldG > inversion_stats MinimumResidueInverterVtk (fieldT & *psi_out*, fieldT & *psi_in*, fieldG & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = mdp_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000) [inline]

7.42 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-sdwf_actions.h File Reference

Classes

- class SDWFActionSlow

  domain wall staggered (WORK IN PROGRESS)

7.42.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

WORK IN PROGRESS

Created with support from the US Department of Energy

7.43 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-sdwf_algorithms.h File Reference

Functions

- void project (staggered_field &psi, sdwf_field &chi, gauge_field &U)
- void project (staggered_field &psi, sdwf_field &chi, gauge_field &U, int sign, int L)
- void project (sdwf_field &chi, staggered_field &psi, gauge_field &U)
- void mul_Q (sdwf_field &psi_out, sdwf_field &psi_in, gauge_field &U, coefficients &coeff, int parity=EVENODD)
- inversion_stats mul_invQ (sdwf_field &psi_out, sdwf_field &psi_in, gauge_field &U, coefficients &coeff, mdp_real absolute_precision=sdwf_inversion_precision, mdp_real relative_precision=0, int max_steps=2000)
- void compute_swirls_field (gauge_field &U)

Variables

- const double MDP_SDWF_SGN = 1.0
- void(* default_sdwf_action)(sdwf_field &, sdwf_field &, gauge_field &, coefficients &, int) = SDWFActionSlow::mul_Q
- inversion_stats(* default_sdwf_inverter)(sdwf_field &, sdwf_field &, gauge_field &, coefficients &, mdp_real, mdp_real, int) = BiConjugateGradientStabilizedInverter<sdwf_field,gauge_field>

7.43.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

WORK IN PROGRESS

Created with support from the US Department of Energy

7.43.2 Function Documentation

7.43.2.1 void compute_swirls_field (gauge_field & *U*)

7.43.2.2 inversion_stats mul_invQ (sdwf_field & *psi_out*, sdwf_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = sdwf_inversion_precision, mdp_real *relative_precision* = 0, int *max_steps* = 2000)

7.43.2.3 void mul_Q (sdwf_field & *psi_out*, sdwf_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, int *parity* = EVENODD)

7.43.2.4 void project (sdwf_field & *chi*, staggered_field & *psi*, gauge_field & *U*)

7.43.2.5 void project (staggered_field & *psi*, sdwf_field & *chi*, gauge_field & *U*, int *sign*, int *L*)

7.43.2.6 void project (staggered_field & *psi*, sdwf_field & *chi*, gauge_field & *U*)

7.43.3 Variable Documentation

7.43.3.1 void(∗ default_sdwf_action)(sdwf_field &, sdwf_field &, gauge_field &, coefficients &, int) = SDWFActionSlow::mul_Q

7.43.3.2 inversion_stats(∗ default_sdwf_inverter)(sdwf_field &, sdwf_field &, gauge_field &, coefficients &, mdp_real, mdp_real, int) = BiConjugateGradientStabilizedInverter<sdwf_field,gauge_field>

7.43.3.3 const double MDP_SDWF_SGN = 1.0

7.44 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-sdwf_field.h File Reference

Classes

- class sdwf_field

  field for domain wall staggered fermions

7.44.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

WORK IN PROGRESS

Created with support from the US Department of Energy

7.45 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-set_random.h File Reference

Functions

- void set_random (generic_field< mdp_complex > &chi, int parity=EVENODD)
- void set_wall_random (generic_field< mdp_complex > &chi, int t=0, int parity=EVENODD)
- void set_zero (generic_field< mdp_complex > &chi, int parity=EVENODD)

7.45.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Function to initialize fields

Distributed under GPL2 License

Created with support from the US Department of Energy

7.45.2 Function Documentation

7.45.2.1 void set_random (generic_field< mdp_complex > & *chi*, int *parity* = EVENODD)

Set the complex field components of chi to be gaussian random numbers with mean=0 and sigma=1 (useful for stochastic propagators). can choose parity=EVEN, ODD or EVENODD

7.45.2.2 void set_wall_random (generic_field< mdp_complex > & *chi*, int *t* = 0, int *parity* = EVENODD)

Set the complex field components of chi to be gaussian random numbers on the wall identified by t with mean=0 and sigma=1 (useful for stochastic propagators). can choose parity=EVEN, ODD or EVENODD attention! does not set to zero other timeslices!!!

7.45.2.3 void set_zero (generic_field< mdp_complex > & *chi*, int *parity* = EVENODD)

Set the complex field components of chi tozero. can choose parity=EVEN, ODD or EVENODD

7.46 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse.h File Reference

Classes

- struct _sse_float
- struct _sse_vector
- struct _sse_int
- struct _sse_double
- struct _sse_su3
- struct _sse_su3_vector
- struct _sse_spinor

Defines

- #define ALIGN16 __attribute__ ((aligned (16)))
- #define ALIGN64 __attribute__ ((aligned (64)))
- #define _ASM __asm__ __volatile__
- #define _sse_float_prefetch_spinor(addr)
- #define _sse_float_prefetch_su3(addr)
- #define _sse_float_pair_load(sl, sh)
- #define _sse_float_pair_load_up(sl, sh)
- #define _sse_float_pair_store(rl, rh)
- #define _sse_float_pair_store_up(rl, rh)
- #define _sse_float_vector_load(s)
- #define _sse_float_vector_load_up(s)
- #define _sse_float_vector_store(r)
- #define _sse_float_vector_mul(c)
- #define _sse_float_vector_add()
- #define _sse_float_vector_sub()
- #define _sse_float_vector_addsub()
- #define _sse_float_su3_multiply(u)
- #define _sse_float_su3_inverse_multiply(u)
- #define _sse_float_vector_subadd()
- #define _sse_float_vector_i_add()
- #define _sse_float_vector_i_sub()
- #define _sse_float_vector_xch_i_add()
- #define _sse_float_vector_xch_i_sub()
- #define _sse_float_vector_i_addsub()
- #define _sse_float_vector_i_subadd()
- #define _sse_float_vector_xch()
- #define _sse_double_prefetch_16(addr)
- #define _sse_double_prefetch_spinor(addr)
- #define _sse_double_prefetch_nta_spinor(addr)
- #define _sse_double_prefetch_su3(addr)
- #define _sse_double_load(s)
- #define _sse_double_load_123(c1, c2, c3)
- #define _sse_double_load_up(s)
- #define _sse_double_load_up_123(c1, c2, c3)

- #define _sse_double_store(r)
- #define _sse_double_store_123(c1, c2, c3)
- #define _sse_double_store_up(r)
- #define _sse_double_store_up_123(c1, c2, c3)
- #define _sse_double_vector_mul(c)
- #define _sse_double_vector_mul_complex(x, y)
- #define _sse_double_vector_add()
- #define _sse_double_vector_sub()
- #define _sse_double_su3_multiply(u)
- #define _sse_double_su3_inverse_multiply(u)
- #define _sse_double_vector_i_mul()
- #define _sse_double_vector_minus_i_mul()
- #define _sse_double_add_norm_square_16(r, c)
- #define _sse_double_add_real_scalar_product_16(r, s, c)
- #define _sse_double_add_imag_scalar_product_16(r, s, c)
- #define _sse_double_hermitian_su3(r, s)
- #define _sse_double_copy_16(r, s)
- #define _sse_double_add_16(r, s)
- #define _sse_double_sub_16(r, s)
- #define _sse_double_add_multiply_16(r, c, s)
- #define _sse_double_multiply_16(r, c, s)

7.46.1 Detailed Description

Version:

2009-12-21

Author:

Martin Luesher and Massimo Di Pierro <mdipierro@cs.depaul.edu>

Basic actions for Wilson Fermions optimized in assembler

7.46.2 Define Documentation

7.46.2.1 #define _ASM __asm__ __volatile__

7.46.2.2 #define _sse_double_add_16(r, s)

7.46.2.3 #define _sse_double_add_imag_scalar_product_16(r, s, c)

7.46.2.4 #define _sse_double_add_multiply_16(r, c, s)

7.46.2.5 #define _sse_double_add_norm_square_16(r, c)

7.46.2.6 #define _sse_double_add_real_scalar_product_16(r, s, c)

7.46.2.7 #define _sse_double_copy_16(r, s)

7.46.2.8 #define _sse_double_hermitian_su3(r, s)

7.46.2.9 #define _sse_double_load(s)

Value:

```
_ASM ("movapd %0, %%xmm0 \n\t" \
      "movapd %1, %%xmm1 \n\t" \
      "movapd %2, %%xmm2" \
      : \
      : \
      "m" ((s).c1), \
      "m" ((s).c2), \
      "m" ((s).c3))
```

7.46.2.10 #define _sse_double_load_123(c1, c2, c3)

Value:

```
_ASM ("movapd %0, %%xmm0 \n\t" \
      "movapd %1, %%xmm1 \n\t" \
      "movapd %2, %%xmm2" \
      : \
      : \
      "m" (c1), \
      "m" (c2), \
      "m" (c3))
```

7.46.2.11 #define _sse_double_load_up(s)

Value:

```
_ASM ("movapd %0, %%xmm3 \n\t" \
      "movapd %1, %%xmm4 \n\t" \
      "movapd %2, %%xmm5" \
      : \
      : \
      "m" ((s).c1), \
      "m" ((s).c2), \
      "m" ((s).c3))
```

7.46.2.12 #define _sse_double_load_up_123(c1, c2, c3)

Value:

```
_ASM ("movapd %0, %%xmm3 \n\t" \
      "movapd %1, %%xmm4 \n\t" \
      "movapd %2, %%xmm5" \
      : \
      : \
      "m" (c1), \
      "m" (c2), \
      "m" (c3))
```

7.46.2.13 #define _sse_double_multiply_16(r, c, s)

7.46.2.14 #define _sse_double_prefetch_16(addr)

Value:

```
_ASM ("prefetcht0 %0" \
      : \
      : "m" (*(addr)))
```

7.46.2.15 #define _sse_double_prefetch_nta_spinor(addr)

Value:

```
_ASM ("prefetchnta %0 \n\t" \
      "prefetchnta %1" \
      : \
      : \
      "m" (*(((char*)((unsigned int)(addr))&~0x7f))), \
      "m" (*(((char*)((unsigned int)(addr))&~0x7f))+128)))
```

7.46.2.16 #define _sse_double_prefetch_spinor(addr)

Value:

```
_ASM ("prefetcht0 %0 \n\t" \
      "prefetcht0 %1" \
      : \
      : \
      "m" (*(((char*)((unsigned int)(addr))&~0x7f))), \
      "m" (*(((char*)((unsigned int)(addr))&~0x7f))+128)))
```

7.46.2.17 #define _sse_double_prefetch_su3(addr)

Value:

```
_ASM ("prefetcht0 %0 \n\t" \
      "prefetcht0 %1" \
      : \
      : \
      "m" (*(((char*)((unsigned int)(addr))&~0x7f))), \
      "m" (*(((char*)((unsigned int)(addr))&~0x7f))+128)))
```

7.46.2.18 #define _sse_double_store(r)

Value:

```
_ASM ("movapd %%xmm0, %0 \n\t" \
      "movapd %%xmm1, %1 \n\t" \
      "movapd %%xmm2, %2" \
      : \
      "=m" ((r).c1), \
      "=m" ((r).c2), \
      "=m" ((r).c3))
```

7.46.2.19 #define _sse_double_store_123(c1, c2, c3)

Value:

```
_ASM ("movapd %%xmm0, %0 \n\t" \
      "movapd %%xmm1, %1 \n\t" \
      "movapd %%xmm2, %2" \
      : \
      "=m" (c1), \
      "=m" (c2), \
      "=m" (c3))
```

7.46.2.20 #define _sse_double_store_up(r)

Value:

```
_ASM ("movapd %%xmm3, %0 \n\t" \
      "movapd %%xmm4, %1 \n\t" \
      "movapd %%xmm5, %2" \
      : \
      "=m" ((r).c1), \
      "=m" ((r).c2), \
      "=m" ((r).c3))
```

7.46.2.21 #define _sse_double_store_up_123(c1, c2, c3)

Value:

```
_ASM ("movapd %%xmm3, %0 \n\t" \
      "movapd %%xmm4, %1 \n\t" \
      "movapd %%xmm5, %2" \
      : \
      "=m" (c1), \
      "=m" (c2), \
      "=m" (c3))
```

7.46.2.22 #define _sse_double_su3_inverse_multiply(u)

7.46.2.23 #define _sse_double_su3_multiply(u)

7.46.2.24 #define _sse_double_sub_16(r, s)

7.46.2.25 #define _sse_double_vector_add()

Value:

```
_ASM ("addpd %%xmm3, %%xmm0 \n\t" \
      "addpd %%xmm4, %%xmm1 \n\t" \
      "addpd %%xmm5, %%xmm2" \
      : \
      :)
```

7.46.2.26 #define _sse_double_vector_i_mul()

Value:

```
_ASM ("shufpd $0x1, %%xmm3, %%xmm3 \n\t" \
      "shufpd $0x1, %%xmm4, %%xmm4 \n\t" \
      "shufpd $0x1, %%xmm5, %%xmm5 \n\t" \
      "xorpd %0, %%xmm3 \n\t" \
      "xorpd %0, %%xmm4 \n\t" \
      "xorpd %0, %%xmm5" \
      : \
      : \
      "m" (_sse_double_sgn))
```

7.46.2.27 #define _sse_double_vector_minus_i_mul()

Value:

```
_ASM ("xorpd %0, %%xmm3 \n\t" \
      "xorpd %0, %%xmm4 \n\t" \
      "xorpd %0, %%xmm5 \n\t" \
      "shufpd $0x1, %%xmm3, %%xmm3 \n\t" \
      "shufpd $0x1, %%xmm4, %%xmm4 \n\t" \
      "shufpd $0x1, %%xmm5, %%xmm5" \
      : \
      : \
      "m" (_sse_double_sgn))
```

7.46.2.28 #define _sse_double_vector_mul(c)

Value:

```
_ASM ("mulpd %0, %%xmm0 \n\t" \
      "mulpd %0, %%xmm1 \n\t" \
      "mulpd %0, %%xmm2" \
      : \
      : \
      "m" (c))
```

7.46.2.29 #define _sse_double_vector_mul_complex(x, y)

Value:

```
_ASM ("movapd %%xmm0, %%xmm3 \n\t" \
      "movapd %%xmm1, %%xmm4 \n\t" \
      "movapd %%xmm2, %%xmm5 \n\t" \
      "mulpd %1, %%xmm3 \n\t" \
      "mulpd %1, %%xmm4 \n\t" \
      "mulpd %1, %%xmm5 \n\t" \
      "shufpd $0x1, %%xmm3, %%xmm3 \n\t" \
```

```
      "shufpd $0x1, %%xmm4, %%xmm4 \n\t" \
      "shufpd $0x1, %%xmm5, %%xmm5 \n\t" \
      "xorpd %2, %%xmm3 \n\t" \
      "xorpd %2, %%xmm4 \n\t" \
      "xorpd %2, %%xmm5 \n\t" \
      "mulpd %0, %%xmm0 \n\t" \
      "mulpd %0, %%xmm1 \n\t" \
      "mulpd %0, %%xmm2 \n\t" \
      "addpd %%xmm0, %%xmm3 \n\t" \
      "addpd %%xmm1, %%xmm4 \n\t" \
      "addpd %%xmm2, %%xmm5" \
      : \
      : \
      "m" (x), \
      "m" (y), \
      "m" (_sse_double_sgn))
```

7.46.2.30 #define _sse_double_vector_sub()

Value:

```
_ASM ("subpd %%xmm3, %%xmm0 \n\t" \
      "subpd %%xmm4, %%xmm1 \n\t" \
      "subpd %%xmm5, %%xmm2" \
      : \
      :)
```

7.46.2.31 #define _sse_float_pair_load(sl, sh)

Value:

```
_ASM ("movlps %0, %%xmm0 \n\t" \
      "movlps %1, %%xmm1 \n\t" \
      "movlps %2, %%xmm2 \n\t" \
      "movhps %3, %%xmm0 \n\t" \
      "movhps %4, %%xmm1 \n\t" \
      "movhps %5, %%xmm2 " \
       : \
       : \
       "m" ((sl).c1), \
       "m" ((sl).c2), \
       "m" ((sl).c3), \
       "m" ((sh).c1), \
       "m" ((sh).c2), \
       "m" ((sh).c3))
```

7.46.2.32 #define _sse_float_pair_load_up(sl, sh)

Value:

```
_ASM ("movlps %0, %%xmm3 \n\t" \
      "movlps %1, %%xmm4 \n\t" \
      "movlps %2, %%xmm5 \n\t" \
      "movhps %3, %%xmm3 \n\t" \
      "movhps %4, %%xmm4 \n\t" \
      "movhps %5, %%xmm5" \
      : \
      : \
      "m" ((sl).c1), \
```

```
      "m" ((sl).c2), \
      "m" ((sl).c3), \
      "m" ((sh).c1), \
      "m" ((sh).c2), \
      "m" ((sh).c3))
```

7.46.2.33 #define _sse_float_pair_store(rl, rh)

Value:

```
_ASM ("movlps %%xmm0, %0 \n\t" \
      "movlps %%xmm1, %1 \n\t" \
      "movlps %%xmm2, %2 \n\t" \
      "movhps %%xmm0, %3 \n\t" \
      "movhps %%xmm1, %4 \n\t" \
      "movhps %%xmm2, %5" \
      : \
      "=m" ((rl).c1), \
      "=m" ((rl).c2), \
      "=m" ((rl).c3), \
      "=m" ((rh).c1), \
      "=m" ((rh).c2), \
      "=m" ((rh).c3))
```

7.46.2.34 #define _sse_float_pair_store_up(rl, rh)

Value:

```
_ASM ("movlps %%xmm3, %0 \n\t" \
      "movlps %%xmm4, %1 \n\t" \
      "movlps %%xmm5, %2 \n\t" \
      "movhps %%xmm3, %3 \n\t" \
      "movhps %%xmm4, %4 \n\t" \
      "movhps %%xmm5, %5" \
      : \
      "=m" ((rl).c1), \
      "=m" ((rl).c2), \
      "=m" ((rl).c3), \
      "=m" ((rh).c1), \
      "=m" ((rh).c2), \
      "=m" ((rh).c3))
```

7.46.2.35 #define _sse_float_prefetch_spinor(addr)

Value:

```
_ASM ("prefetcht0 %0 \n\t" \
      "prefetcht0 %1" \
      : \
      : \
      "m" (*(((char*)((unsigned int)(addr))&~0x7f))), \
      "m" (*(((char*)((unsigned int)(addr))&~0x7f))+128)))
```

7.46.2.36 #define _sse_float_prefetch_su3(addr)

Value:

```
_ASM ("prefetcht0 %0 \n\t" \
      "prefetcht0 %1" \
      : \
      : \
      "m" (*(((char*)(((unsigned int)(addr))&~0x7f)))), \
      "m" (*(((char*)(((unsigned int)(addr))&~0x7f))+128)))
```

7.46.2.37 #define _sse_float_su3_inverse_multiply(u)

7.46.2.38 #define _sse_float_su3_multiply(u)

7.46.2.39 #define _sse_float_vector_add()

Value:

```
_ASM ("addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
      "addps %%xmm5, %%xmm2 \n\t" \
      : \
      : )
```

7.46.2.40 #define _sse_float_vector_addsub()

Value:

```
_ASM ("mulps %0, %%xmm3 \n\t" \
      "mulps %0, %%xmm4 \n\t" \
      "mulps %0, %%xmm5 \n\t" \
      "addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
      "addps %%xmm5, %%xmm2" \
      : \
      : \
      "m" (_sse_float_sgn34))
```

7.46.2.41 #define _sse_float_vector_i_add()

Value:

```
_ASM ("shufps $0xb1, %%xmm3, %%xmm3 \n\t" \
      "shufps $0xb1, %%xmm4, %%xmm4 \n\t" \
      "shufps $0xb1, %%xmm5, %%xmm5 \n\t" \
      "mulps %0, %%xmm3 \n\t" \
      "mulps %0, %%xmm4 \n\t" \
      "mulps %0, %%xmm5 \n\t" \
      "addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
      "addps %%xmm5, %%xmm2" \
      : \
      : \
      "m" (_sse_float_sgn13))
```

7.46.2.42 #define _sse_float_vector_i_addsub()

Value:

```
_ASM ("shufps $0xb1, %%xmm3, %%xmm3 \n\t" \
      "shufps $0xb1, %%xmm4, %%xmm4 \n\t" \
      "shufps $0xb1, %%xmm5, %%xmm5 \n\t" \
      "mulps %0, %%xmm3 \n\t" \
      "mulps %0, %%xmm4 \n\t" \
      "mulps %0, %%xmm5 \n\t" \
      "addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
      "addps %%xmm5, %%xmm2" \
      : \
      : \
      "m" (_sse_float_sgn14))
```

7.46.2.43 #define _sse_float_vector_i_sub()

Value:

```
_ASM ("shufps $0xb1, %%xmm3, %%xmm3 \n\t" \
      "shufps $0xb1, %%xmm4, %%xmm4 \n\t" \
      "shufps $0xb1, %%xmm5, %%xmm5 \n\t" \
      "mulps %0, %%xmm3 \n\t" \
      "mulps %0, %%xmm4 \n\t" \
      "mulps %0, %%xmm5 \n\t" \
      "addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
      "addps %%xmm5, %%xmm2" \
      : \
      : \
      "m" (_sse_float_sgn24))
```

7.46.2.44 #define _sse_float_vector_i_subadd()

Value:

```
_ASM ("shufps $0xb1, %%xmm3, %%xmm3 \n\t" \
      "shufps $0xb1, %%xmm4, %%xmm4 \n\t" \
      "shufps $0xb1, %%xmm5, %%xmm5 \n\t" \
      "mulps %0, %%xmm3 \n\t" \
      "mulps %0, %%xmm4 \n\t" \
      "mulps %0, %%xmm5 \n\t" \
      "addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
      "addps %%xmm5, %%xmm2" \
      : \
      : \
      "m" (_sse_float_sgn23))
```

7.46.2.45 #define _sse_float_vector_load(s)

Value:

```
_ASM ("movaps %0, %%xmm0 \n\t" \
      "movaps %1, %%xmm1 \n\t" \
      "movaps %2, %%xmm2" \
      : \
      : \
      "m" ((s).c1), \
      "m" ((s).c2), \
      "m" ((s).c3))
```

7.46.2.46 #define _sse_float_vector_load_up(s)

Value:

```
_ASM ("movaps %0, %%xmm3 \n\t" \
      "movaps %1, %%xmm4 \n\t" \
      "movaps %2, %%xmm5" \
      : \
      : \
      "m" ((s).c1), \
      "m" ((s).c2), \
      "m" ((s).c3))
```

7.46.2.47 #define _sse_float_vector_mul(c)

Value:

```
_ASM ("mulps %0, %%xmm0 \n\t" \
      "mulps %0, %%xmm1 \n\t" \
      "mulps %0, %%xmm2" \
      : \
      : \
      "m" (c))
```

7.46.2.48 #define _sse_float_vector_store(r)

Value:

```
_ASM ("movaps %%xmm0, %0 \n\t" \
      "movaps %%xmm1, %1 \n\t" \
      "movaps %%xmm2, %2" \
      : \
      "=m" ((r).c1), \
      "=m" ((r).c2), \
      "=m" ((r).c3))
```

7.46.2.49 #define _sse_float_vector_sub()

Value:

```
_ASM ("subps %%xmm3, %%xmm0 \n\t" \
      "subps %%xmm4, %%xmm1 \n\t" \
      "subps %%xmm5, %%xmm2" \
      : \
      :)
```

7.46.2.50 #define _sse_float_vector_subadd()

Value:

```
_ASM ("mulps %0, %%xmm3 \n\t" \
      "mulps %0, %%xmm4 \n\t" \
      "mulps %0, %%xmm5 \n\t" \
      "addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
```

```
      "addps %%xmm5, %%xmm2" \
      : \
      : \
      "m" (_sse_float_sgn12))
```

7.46.2.51 #define _sse_float_vector_xch()

Value:

```
_ASM ("shufps $0x4e, %%xmm3, %%xmm3 \n\t" \
      "shufps $0x4e, %%xmm4, %%xmm4 \n\t" \
      "shufps $0x4e, %%xmm5, %%xmm5" \
      : \
      :)
```

7.46.2.52 #define _sse_float_vector_xch_i_add()

Value:

```
_ASM ("shufps $0x1b, %%xmm3, %%xmm3 \n\t" \
      "shufps $0x1b, %%xmm4, %%xmm4 \n\t" \
      "shufps $0x1b, %%xmm5, %%xmm5 \n\t" \
      "mulps %0, %%xmm3 \n\t" \
      "mulps %0, %%xmm4 \n\t" \
      "mulps %0, %%xmm5 \n\t" \
      "addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
      "addps %%xmm5, %%xmm2" \
      : \
      : \
      "m" (_sse_float_sgn13))
```

7.46.2.53 #define _sse_float_vector_xch_i_sub()

Value:

```
_ASM ("shufps $0x1b, %%xmm3, %%xmm3 \n\t" \
      "shufps $0x1b, %%xmm4, %%xmm4 \n\t" \
      "shufps $0x1b, %%xmm5, %%xmm5 \n\t" \
      "mulps %0, %%xmm3 \n\t" \
      "mulps %0, %%xmm4 \n\t" \
      "mulps %0, %%xmm5 \n\t" \
      "addps %%xmm3, %%xmm0 \n\t" \
      "addps %%xmm4, %%xmm1 \n\t" \
      "addps %%xmm5, %%xmm2" \
      : \
      : \
      "m" (_sse_float_sgn24))
```

7.46.2.54 #define ALIGN16 __attribute__ ((aligned (16)))

7.46.2.55 #define ALIGN64 __attribute__ ((aligned (64)))

7.47 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_sse_su3.h File Reference

7.47.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Stuff for SSE/SSE2 compile with -DSSE2

Created with support from the US Department of Energy

7.48 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_actions.h File Reference

Classes

- class StaggeredAsqtadActionSlow

  Staggered/Asqtad action (SLOW: DO NOT USE IN PRODUCTION).

- class StaggeredAsqtadActionFast

  Staggered/Asqtad action.

7.48.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Stuff for SSE/SSE2 compile with -DSSE2

Created with support from the US Department of Energy

7.49 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-staggered_actions_sse2.h File Reference

7.49.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Stuff for SSE/SSE2 compile with -DSSE2

Created with support from the US Department of Energy

7.50 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_algorithms.h File Reference

Functions

- mdp_matrix Omega4x4 (mdp_site x)
- void mul_Q (staggered_field &psi_out, staggered_field &psi_in, gauge_field &U, coefficients &coeff, int parity=EVENODD)

  Executes current Staggered/Asqtad action.

- inversion_stats mul_invQ (staggered_field &psi_out, staggered_field &psi_in, gauge_field &U, coefficients &coeff, mdp_real absolute_precision=staggered_inversion_precision, mdp_real relative_precision=0, int max_steps=2000)

  Executes current Staggered/Asqtad inverter.

- mdp_array< mdp_real, 1 > lepage_coefficients (mdp_real plaquette, char type[ ])
- void lepage_improved_links (gauge_field &V, gauge_field &U, mdp_array< mdp_real, 1 > c, int project=false)
- void staggered_rephase (gauge_field &U, staggered_field &chi)

Variables

- void(∗ default_staggered_action)(staggered_field &, staggered_field &, gauge_field &, coefficients &, int) = StaggeredAsqtadActionFast::mul_Q

  Pointer to current Staggered/Asqtad action.

- inversion_stats(∗ default_staggered_inverter)(staggered_field &, staggered_field &, gauge_field &, coefficients &, mdp_real, mdp_real, int) = &(BiCGStab::inverter<staggered_field,gauge_field>)

  Pointer to current Staggered/Asqtad inverter.

7.50.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Various stuff for staggered fermions

Distributed under GPL2 License

Created with support from the US Department of Energy

7.50.2 Function Documentation

7.50.2.1 mdp_array<mdp_real,1> lepage_coefficients (mdp_real *plaquette*, char *type*[ ])

Takes a plaquette and a type of action and returns a 1D array with weights of paths required to build fat links for the action

See also:

lepage_improved_links()

7.50.2.2 void lepage_improved_links (gauge_field & *V*, gauge_field & *U*, mdp_array< mdp_real, 1 > *c*, int *project* = `false`)

Takes a gauge field U and a set of coefficients as computed by lepage_coefficients() and fills the gauge field V with fat links and Long links

Example:

```
/// gauge_field U(lattice,nc);
/// gauge_field V(lattice,nc);
/// U.load("myfield");
/// float p=1.0; // the average plaquette
/// lepage_improved_links(V,U,lepage_coefficients(p,"Full"),false);
/// /// now use V instead of U for staggered actions and inverters
///
```

Note that the type of action can be

- "Full" for full as asqtad
- "Staple+Naik"
- "Fat3"
- "Fat5"
- "Fat7" Also note that if project==true the fat links are projected back to SU(nc)

7.50.2.3 inversion_stats mul_invQ (staggered_field & *psi_out*, staggered_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, mdp_real *absolute_precision* = staggered_inversion_precision, mdp_real *relative_precision* = `0`, int *max_steps* = `2000`)

Executes current Staggered/Asqtad inverter.

7.50.2.4 void mul_Q (staggered_field & *psi_out*, staggered_field & *psi_in*, gauge_field & *U*, coefficients & *coeff*, int *parity* = EVENODD)

Executes current Staggered/Asqtad action.

7.50.2.5 mdp_matrix Omega4x4 (mdp_site *x*)

7.50.2.6 void staggered_rephase (gauge_field & *U*, staggered_field & *chi*)

7.50.3 Variable Documentation

7.50.3.1 void(∗ default_staggered_action)(staggered_field &, staggered_field &, gauge_field &, coefficients &, int) = StaggeredAsqtadActionFast::mul_Q

Pointer to current Staggered/Asqtad action.

7.50.3.2 inversion_stats(∗ default_staggered_inverter)(staggered_field &, staggered_field &, gauge_field &, coefficients &, mdp_real, mdp_real, int) = &(BiCGStab::inverter<staggered_field,gauge_field>)

Pointer to current Staggered/Asqtad inverter.

7.51 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-staggered_field.h File Reference

Classes

- class staggered_field

  staggered fermionic field

7.51.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Stuff for SSE/SSE2 compile with -DSSE2

Created with support from the US Department of Energy

7.52 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_mesons.h File Reference

Classes

- class phase_field

Functions

- void operator_staggered_meson (staggered_field &out, staggered_field &in, phase_field &phases, gauge_field &U)
- mdp_matrix make_meson (gauge_field &U, gauge_field &V, mdp_matrix GAMMA, mdp_matrix ZETA, coefficients &coeff1, coefficients &coeff2, int source1_type=wall_source, int source2_type=wall_source &local_source, mdp_real precision=1e-7)
- mdp_matrix GoldstonBoson_5x5 (gauge_field &U, gauge_field &V, coefficients &coeff, float precision=1e-6)

Variables

- const int local_source = 1
- const int wall_source = 2

7.52.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Convenience functions to make staggered mesons

Created with support from the US Department of Energy

7.52.2 Function Documentation

7.52.2.1 mdp_matrix GoldstonBoson_5x5 (gauge_field & *U*, gauge_field & *V*, coefficients & *coeff*, float *precision* = `1e-6`)

7.52.2.2 mdp_matrix make_meson (gauge_field & *U*, gauge_field & *V*, mdp_matrix *GAMMA*, mdp_matrix *ZETA*, coefficients & *coeff1*, coefficients & *coeff2*, int *source1_type* = wall_source, int *source2_type* = wall_source `&` local_source, mdp_real *precision* = `1e-7`)

7.52.2.3 void operator_staggered_meson (staggered_field & *out*, staggered_field & *in*, phase_field & *phases*, gauge_field & *U*)

7.52.3 Variable Documentation

7.52.3.1 const int local_source = 1

7.52.3.2 const int wall_source = 2

7.53 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_propagator.h File Reference

Classes

- class staggered_propagator

  staggared quark propagator

7.53.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Various stuff for staggered fermions

Created with support from the US Department of Energy

7.54 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_staggered_uml_inverter.h File Reference

Classes

- class StaggeredBiCGUML

  MILC staggered UML inverter (optimized bicgstab).

7.54.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Various stuff for staggered fermions

Created with support from the US Department of Energy

7.55 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_su_generators.h File Reference

Classes

- class SU_Generators

7.55.1 Detailed Description

Version:

11-3-2009

Author:

Simon Catterall and Massimo Di Pierro

7.56 /Users/mdipierro/fermiqcd/development/Libraries/fermiqcd_-topological_charge.h File Reference

Classes

- class ApeSmearing

Functions

- void compute_em_notrace_field (gauge_field &U)
- void topological_charge (mdp_field< float > &Q, gauge_field &U)
- float topological_charge_vtk (gauge_field &U, string filename, int t=-1)

7.56.1 Function Documentation

7.56.1.1 void compute_em_notrace_field (gauge_field & *U*)

7.56.1.2 void topological_charge (mdp_field< float > & *Q*, gauge_field & *U*)

7.56.1.3 float topological_charge_vtk (gauge_field & *U*, string *filename*, int *t* = `-1`)

7.57 /Users/mdipierro/fermiqcd/development/Libraries/make_actions.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- void test_gauge (int nt, int nx, int ny, int nz, int nc)
- void test_gauge_improved (int nt, int nx, int ny, int nz, int nc)
- void test_fermi (int nt, int nx, int ny, int nz, int nc)
- void test_staggered (int nt, int nx, int ny, int nz, int nc)
- void test_dwfermi (int nt, int nx, int ny, int nz, int nc)
- int main (int argc, char **argv)

7.57.1 Function Documentation

7.57.1.1 int main (int *argc*, char ** *argv*)

7.57.1.2 void test_dwfermi (int *nt*, int *nx*, int *ny*, int *nz*, int *nc*)

7.57.1.3 void test_fermi (int *nt*, int *nx*, int *ny*, int *nz*, int *nc*)

7.57.1.4 void test_gauge (int *nt*, int *nx*, int *ny*, int *nz*, int *nc*)

7.57.1.5 void test_gauge_improved (int *nt*, int *nx*, int *ny*, int *nz*, int *nc*)

7.57.1.6 void test_staggered (int *nt*, int *nx*, int *ny*, int *nz*, int *nc*)

7.58 /Users/mdipierro/fermiqcd/development/Libraries/make_fermi_pion_noprop.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.58.1 Function Documentation

7.58.1.1 int main (int *argc*, char ** *argv*)

7.59 /Users/mdipierro/fermiqcd/development/Libraries/make_fermi_pion_prop.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.59.1 Function Documentation

7.59.1.1 int main (int *argc*, char ** *argv*)

7.60 /Users/mdipierro/fermiqcd/development/Libraries/make_fermi_vmeson_noprop.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.60.1 Function Documentation

7.60.1.1 int main (int *argc*, char ** *argv*)

7.61 /Users/mdipierro/fermiqcd/development/Libraries/make_fermi_vmeson_prop.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.61.1 Function Documentation

7.61.1.1 int main (int *argc*, char ** *argv*)

7.62 /Users/mdipierro/fermiqcd/development/Libraries/make_gauge_cold.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.62.1 Function Documentation

7.62.1.1 int main (int *argc*, char ** *argv*)

7.63 /Users/mdipierro/fermiqcd/development/Libraries/make_gauge_configurations.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.63.1 Function Documentation

7.63.1.1 int main (int *argc*, char ** *argv*)

7.64 /Users/mdipierro/fermiqcd/development/Libraries/make_gauge_hot.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.64.1 Function Documentation

7.64.1.1 int main (int *argc*, char ** *argv*)

7.65 /Users/mdipierro/fermiqcd/development/Libraries/make_improved_gauge_configurations.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.65.1 Function Documentation

7.65.1.1 int main (int *argc*, char ** *argv*)

7.66 /Users/mdipierro/fermiqcd/development/Libraries/make_plaquettes.cpp File Reference

```
#include "fermiqcd.h"
```

Functions

- int main (int argc, char **argv)

7.66.1 Function Documentation

7.66.1.1 int main (int *argc*, char ** *argv*)

7.67 /Users/mdipierro/fermiqcd/development/Libraries/mdp.h File Reference

```
#include <iostream>
#include <fstream>
#include <cstdio>
#include <cstdlib>
#include <cmath>
#include <cstring>
#include <ctime>
#include <cassert>
#include <typeinfo>
#include <malloc.h>
#include <string>
#include <vector>
#include <map>
#include <deque>
#include <climits>
#include "glob.h"
#include <unistd.h>
#include <sys/time.h>
#include <sys/file.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/socket.h>
#include <fcntl.h>
#include "mdp_version.h"
#include "mdp_macros.h"
#include "mdp_global_vars.h"
#include "mdp_dynalloc.h"
#include "mdp_endianess_converter.h"
#include "mdp_timer.h"
#include "mdp_complex.h"
#include "mdp_delta.h"
#include "mdp_array.h"
#include "mdp_matrix.h"
#include "mdp_log.h"
```

```
#include "mdp_psim.h"
#include "mdp_communicator.h"
#include "mdp_prng.h"
#include "mdp_jackboot.h"
#include "mdp_topologies.h"
#include "mdp_partitionings.h"
#include "mdp_lattice.h"
#include "mdp_vector.h"
#include "mdp_site.h"
#include "mdp_field.h"
#include "mdp_utils.h"
#include "mdp_postscript.h"
#include "mdp_field_update.h"
#include "mdp_field_load.h"
#include "mdp_field_save.h"
#include "mdp_field_save_vtk.h"
#include "mdp_save_partitioning_vtk.h"
#include "mdp_mod2sign.h"
#include "mdp_permutations.h"
#include "mdp_complex_field.h"
#include "mdp_matrix_field.h"
#include "mdp_vector_field.h"
#include "mdp_nmatrix_field.h"
#include "mdp_compatibility_macros.h"
#include "mdp_prompt.h"
#include "mdp_measure.h"
#include "mdp_matrix_test.h"
#include "mdp_field_test.h"
```

Defines

- #define endl "\n"

7.67.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Includes all mdp_*.h header files

7.67.2 Define Documentation

7.67.2.1 #define endl "\n"

7.68 /Users/mdipierro/fermiqcd/development/Libraries/mdp_array.h File Reference

Classes

- class mdp_array< T, nc_ >

  generic container for multidimensional arrays

7.68.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains declaration of class mdp_array

7.69 /Users/mdipierro/fermiqcd/development/Libraries/mdp_communicator.h File Reference

```
#include "time.h"
```

Classes

- class mdp_communicator

  DO NOT INSTANTIATE use object mdp instead.

Typedefs

- typedef int mdp_request

Functions

- void _mpi_error_message (string a, string b, int c)
- void begin_function (string s)

  Logs in xml the start of a function with message s.

- void end_function (string s)

  Logs in xml the end of a function with message s.

Variables

- mdp_communicator mdp

  the only communicator object

- mdp_communicator & mpi = mdp

  alias for mdp

7.69.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains declaration of class mdp_array

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.69.2 Typedef Documentation

7.69.2.1 typedef int mdp_request

7.69.3 Function Documentation

7.69.3.1 void _mpi_error_message (string *a*, string *b*, int *c*)

7.69.3.2 void begin_function (string *s*) [inline]

Logs in xml the start of a function with message s.

7.69.3.3 void end_function (string *s*) [inline]

Logs in xml the end of a function with message s.

7.69.4 Variable Documentation

7.69.4.1 mdp_communicator mdp

the only communicator object

7.69.4.2 mdp_communicator& mpi = mdp

alias for mdp

7.70 /Users/mdipierro/fermiqcd/development/Libraries/mdp_compatibility_macros.h File Reference

Defines

- #define myreal mdp_real
- #define site mdp_site
- #define Complex mdp_complex
- #define Matrix mdp_matrix
- #define Random mdp_random
- #define Measure mdp_measure
- #define DynamicArray mdp_array
- #define JackBoot mdp_jackboot
- #define generic_lattice mdp_lattice
- #define generic_field mdp_field
- #define Matrix_field mdp_matrix_field
- #define Vector_field mdp_vector_field
- #define NMatrix_field mdp_nmatrix_field
- #define mdp_random_generator mdp_prng

7.70.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains macros for backward compatibility now deprecated

7.70.2 Define Documentation

7.70.2.1 #define Complex mdp_complex

7.70.2.2 #define DynamicArray mdp_array

7.70.2.3 #define generic_field mdp_field

7.70.2.4 #define generic_lattice mdp_lattice

7.70.2.5 #define JackBoot mdp_jackboot

7.70.2.6 #define Matrix mdp_matrix

7.70.2.7 #define Matrix_field mdp_matrix_field

7.70.2.8 #define mdp_random_generator mdp_prng

7.70.2.9 #define Measure mdp_measure

7.70.2.10 #define myreal mdp_real

7.70.2.11 #define NMatrix_field mdp_nmatrix_field

7.70.2.12 #define Random mdp_random

7.70.2.13 #define site mdp_site

7.70.2.14 #define Vector_field mdp_vector_field

7.71 /Users/mdipierro/fermiqcd/development/Libraries/mdp_complex.h File Reference

Classes

- class mdp_complex

  portable complex numbers

Functions

- mdp_complex operator+ (const mdp_complex &a, const mdp_complex &b)
- mdp_complex operator- (const mdp_complex &a, const mdp_complex &b)
- mdp_complex operator* (const mdp_complex &a, const mdp_complex &b)
- mdp_complex operator/ (const mdp_complex &a, const mdp_complex &b)
- mdp_complex operator+ (const mdp_complex &a, const int c)
- mdp_complex operator- (const mdp_complex &a, const int c)
- mdp_complex operator* (const mdp_complex &a, const int c)
- mdp_complex operator/ (const mdp_complex &a, const int c)
- mdp_complex operator+ (const int a, const mdp_complex &c)
- mdp_complex operator- (const int a, const mdp_complex &c)
- mdp_complex operator* (const int a, const mdp_complex &c)
- mdp_complex operator/ (const int a, const mdp_complex &c)
- mdp_complex operator+ (const mdp_complex &a, const float c)
- mdp_complex operator- (const mdp_complex &a, const float c)
- mdp_complex operator* (const mdp_complex &a, const float c)
- mdp_complex operator/ (const mdp_complex &a, const float c)
- mdp_complex operator+ (const float a, const mdp_complex &c)
- mdp_complex operator- (const float a, const mdp_complex &c)
- mdp_complex operator* (const float a, const mdp_complex &c)
- mdp_complex operator/ (const float a, const mdp_complex &c)
- mdp_complex operator+ (const mdp_complex &a, const double c)
- mdp_complex operator- (const mdp_complex &a, const double c)
- mdp_complex operator* (const mdp_complex &a, const double c)
- mdp_complex operator/ (const mdp_complex &a, const double c)
- mdp_complex operator+ (const double a, const mdp_complex &c)
- mdp_complex operator- (const double a, const mdp_complex &c)
- mdp_complex operator* (const double a, const mdp_complex &c)
- mdp_complex operator/ (const double a, const mdp_complex &c)
- mdp_real abs2 (const mdp_complex &a)
- ostream & operator<< (ostream &os, const mdp_complex &a)

Variables

- const mdp_complex I = mdp_complex(0,1)

7.71.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains delcaration of class mdp_complex for complex numbers

7.71.2 Function Documentation

7.71.2.1 mdp_real abs2 (const mdp_complex & *a*) [inline]

7.71.2.2 mdp_complex operator∗ (const double *a*, const mdp_complex & *c*) [inline]

7.71.2.3 mdp_complex operator∗ (const mdp_complex & *a*, const double *c*) [inline]

7.71.2.4 mdp_complex operator∗ (const float *a*, const mdp_complex & *c*) [inline]

7.71.2.5 mdp_complex operator∗ (const mdp_complex & *a*, const float *c*) [inline]

7.71.2.6 mdp_complex operator∗ (const int *a*, const mdp_complex & *c*) [inline]

7.71.2.7 mdp_complex operator∗ (const mdp_complex & *a*, const int *c*) [inline]

7.71.2.8 mdp_complex operator∗ (const mdp_complex & *a*, const mdp_complex & *b*) [inline]

7.71.2.9 mdp_complex operator+ (const double *a*, const mdp_complex & *c*) [inline]

7.71.2.10 mdp_complex operator+ (const mdp_complex & *a*, const double *c*) [inline]

7.71.2.11 mdp_complex operator+ (const float *a*, const mdp_complex & *c*) [inline]

7.71.2.12 mdp_complex operator+ (const mdp_complex & *a*, const float *c*) [inline]

7.71.2.13 mdp_complex operator+ (const int *a*, const mdp_complex & *c*) [inline]

7.71.2.14 mdp_complex operator+ (const mdp_complex & *a*, const int *c*) [inline]

7.71.2.15 mdp_complex operator+ (const mdp_complex & *a*, const mdp_complex & *b*) [inline]

7.71.2.16 mdp_complex operator- (const double *a*, const mdp_complex & *c*) [inline]

7.71.2.17 mdp_complex operator- (const mdp_complex & *a*, const double *c*) [inline]

7.71.2.18 mdp_complex operator- (const float *a*, const mdp_complex & *c*) [inline]

7.71.2.19 mdp_complex operator- (const mdp_complex & *a*, const float *c*) [inline]

7.71.2.20 mdp_complex operator- (const int *a*, const mdp_complex & *c*) [inline]

7.71.2.21 mdp_complex operator- (const mdp_complex & *a*, const int *c*) [inline]

7.71.2.22 mdp_complex operator- (const mdp_complex & *a*, const mdp_complex & *b*) [inline]

7.71.2.23 mdp_complex operator/ (const double *a*, const mdp_complex & *c*) [inline]

7.71.2.24 mdp_complex operator/ (const mdp_complex & *a*, const double *c*) [inline]

7.71.2.25 mdp_complex operator/ (const float *a*, const mdp_complex & *c*) [inline]

7.71.2.26 mdp_complex operator/ (const mdp_complex & *a*, const float *c*) [inline]

7.71.2.27 mdp_complex operator/ (const int *a*, const mdp_complex & *c*) [inline]

7.71.2.28 mdp_complex operator/ (const mdp_complex & *a*, const int *c*) [inline]

7.71.2.29 mdp_complex operator/ (const mdp_complex & *a*, const mdp_complex & *b*)

7.72 /Users/mdipierro/fermiqcd/development/Libraries/mdp_complex_field.h File Reference

Classes

- class mdp_complex_field

  field of complex numbers or vectors of complex numbers

Functions

- bool mdp_write_double_as_float (FILE *fp, void *data, mdp_int psize, mdp_int header_size, mdp_int position, const mdp_lattice &lattice)
- bool mdp_read_double_as_float (FILE *fp, void *data, mdp_int psize, mdp_int header_size, mdp_int position, const mdp_lattice &lattice)
- bool mdp_write_float_as_double (FILE *fp, void *data, mdp_int psize, mdp_int header_size, mdp_int position, const mdp_lattice &lattice)
- bool mdp_read_float_as_double (FILE *fp, void *data, mdp_int psize, mdp_int header_size, mdp_int position, const mdp_lattice &lattice)

7.72.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains declaration of class mdp_complex_field

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.72.2 Function Documentation

7.72.2.1 bool mdp_read_double_as_float (FILE * *fp*, void * *data*, mdp_int *psize*, mdp_int *header_size*, mdp_int *position*, const mdp_lattice & *lattice*)

7.72.2.2 bool mdp_read_float_as_double (FILE * *fp*, void * *data*, mdp_int *psize*, mdp_int *header_size*, mdp_int *position*, const mdp_lattice & *lattice*)

7.72.2.3 bool mdp_write_double_as_float (FILE * *fp*, void * *data*, mdp_int *psize*, mdp_int *header_size*, mdp_int *position*, const mdp_lattice & *lattice*)

7.72.2.4 bool mdp_write_float_as_double (FILE * *fp*, void * *data*, mdp_int *psize*, mdp_int *header_size*, mdp_int *position*, const mdp_lattice & *lattice*)

7.73 /Users/mdipierro/fermiqcd/development/Libraries/mdp_-delta.h File Reference

Functions

- template<class T >
  const bool delta (const T &i, const T &j)

  True if i==j, false otherwise.

7.73.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains declaration delta function

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.73.2 Function Documentation

7.73.2.1 template<class T > const bool delta (const T & *i*, const T & *j*) [inline]

True if i==j, false otherwise.

7.74 /Users/mdipierro/fermiqcd/development/Libraries/mdp_deprecatedIO.h File Reference

7.74.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Old functions for file IO now deprecated

7.75 /Users/mdipierro/fermiqcd/development/Libraries/mdp_dynalloc.h File Reference

```
#include "malloc.h"
```

Functions

- void * operator new (size_t size)
- void operator delete (void *pointer)
- void * operator new[ ] (size_t size)
- void operator delete[ ] (void *pointer)

7.75.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Declaration of overloaded new and delete operators to use memalign when compiled with define SSE2 Required for SSE/SSE2 assembly macros

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.75.2 Function Documentation

7.75.2.1 void operator delete (void * *pointer*)

7.75.2.2 void operator delete[ ] (void * *pointer*)

7.75.2.3 void* operator new (size_t *size*)

7.75.2.4 void* operator new[ ] (size_t *size*)

7.76 /Users/mdipierro/fermiqcd/development/Libraries/mdp_endianess_converter.h File Reference

Functions

- template<class T >
  void switch_endianess_byte4 (T &a)

  Converts endianess of object passed by reference.

- template<class T >
  void switch_endianess_byte8 (T &a)

7.76.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains declaration of function swicth_endianess_byte4()

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.76.2 Function Documentation

7.76.2.1 template<class T > void switch_endianess_byte4 (T & *a*) [inline]

Converts endianess of object passed by reference.

7.76.2.2 template<class T > void switch_endianess_byte8 (T & *a*) [inline]

7.77 /Users/mdipierro/fermiqcd/development/Libraries/mdp_field.h File Reference

Classes

- class mdp_field_file_header
  header for field file IO
- class mdp_field< T >
  most generic field object

7.77.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains declaration of class mdp_field

7.78 /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_load.h File Reference

Functions

- bool mdp_default_user_read (FILE *fp, void *p, mdp_int psize, mdp_int header_size, mdp_int position, const mdp_lattice &lattice)

  Auxiliary function.

7.78.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains file IO operations for class mdp_field

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.78.2 Function Documentation

7.78.2.1 bool mdp_default_user_read (FILE * *fp*, void * *p*, mdp_int *psize*, mdp_int *header_size*, mdp_int *position*, const mdp_lattice & *lattice*)

Auxiliary function.

7.79 /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_save.h File Reference

Functions

- bool mdp_default_user_write (FILE *fp, void *p, mdp_int psize, mdp_int header_size, mdp_int position, const mdp_lattice &lattice)

  Auxiliary function.

7.79.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains file IO operations for class mdp_field

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.79.2 Function Documentation

7.79.2.1 bool mdp_default_user_write (FILE * *fp*, void * *p*, mdp_int *psize*, mdp_int *header_size*, mdp_int *position*, const mdp_lattice & *lattice*)

Auxiliary function.

7.80 /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_save_vtk.h File Reference

Functions

- mdp_field< float > & cumulate_field (mdp_field< float > &field, string filename)

7.80.1 Function Documentation

7.80.1.1 mdp_field<float>& cumulate_field (mdp_field< float > & *field*, string *filename*)

7.81 /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_test.h File Reference

Functions

- bool mdp_field_test (int argc, char **argv)
  For debugging purposes only.

7.81.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains a sample test (main) function

7.81.2 Function Documentation

7.81.2.1 bool mdp_field_test (int *argc*, char ** *argv*)

For debugging purposes only.

7.82 /Users/mdipierro/fermiqcd/development/Libraries/mdp_field_update.h File Reference

7.82.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains mdp_field::update()

7.83 /Users/mdipierro/fermiqcd/development/Libraries/mdp_-fitting_functions.h File Reference

Typedefs

- typedef float(∗ BLM_function)(float, float ∗, mdp_int, void ∗)

Functions

- void linear_fit (float ∗x, Measure ∗y, mdp_int i0, mdp_int in, Measure ∗a)
  Fits y[i], x[i] for i0<=i<in with y=a[0]∗x+a[1].
- float golden_rule (float(∗fp)(float ∗, mdp_int, void ∗), float &xmin, float ax, float bx, float cx, float tol=0.001, mdp_int niter=100, void ∗dummy=0)
- float BLMaux (float ∗x, Measure ∗y, mdp_int i_min, mdp_int i_max, float ∗a, float ∗a0, mdp_matrix &sigma, int ma, mdp_matrix &alpha, mdp_matrix &beta, BLM_function func, float h, void ∗junk)
- float BaesyanLevenbergMarquardt (float ∗x, Measure ∗y, mdp_int i_min, mdp_int i_max, float ∗a, int ma, mdp_matrix &covar, BLM_function func, float h=0.001, mdp_int nmax=1000, void ∗junk=0)

7.83.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains mdp_field::update()

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.83.2 Typedef Documentation

7.83.2.1 typedef float(∗ BLM_function)(float, float ∗, mdp_int, void ∗)

7.83.3 Function Documentation

7.83.3.1 float BaesyanLevenbergMarquardt (float ∗ *x*, Measure ∗ *y*, mdp_int *i_min*, mdp_int *i_max*, float ∗ *a*, int *ma*, mdp_matrix & *covar*, BLM_function *func*, float *h* = 0.001, mdp_int *nmax* = 1000, void ∗ *junk* = 0)

This implements the BaesyanLevenbergMarquardt It uses mdp_matrix. Arguments are:

x[i] : an array of float y[i] : an array of Measures i_min, i_max : range to be used in the fit points within the range that have y[i].num=0 are ignored a[i], ma : vector of paramters for the fit and number of parameters they are all used in the fit the initial values are used as preons covar(i,j) : covariance matrix for the preons func(x,a,ma,junk) : the function to be used in the fit h : a float used to evaluate derivatives nmax : max number of iterations junk : junk to be passed to func

Return the Baesyan ChiSquare. To obtain the correct chi_square rerun it with same ftting values and nmax=1;

7.83.3.2 float BLMaux (float * *x*, Measure * *y*, mdp_int *i_min*, mdp_int *i_max*, float * *a*, float * *a0*, mdp_matrix & *sigma*, int *ma*, mdp_matrix & *alpha*, mdp_matrix & *beta*, BLM_function *func*, float *h*, void * *junk*)

This function is used by the BayesianLevenbergMarquardt It computes the chi_square (including the Baesyan term) and fills alpha and beta

alpha(j,k)= (Dy(x[i],a)/Da[j])*(Dy(x[i],a)/Da[k])/dy[i]^2 beta(j)=sum_i (y[i]-y(x[i],a))*(dy(x[i],a)/da[j])/dy[i]^2

chi_square= (y[i]-y(x[i],a))*(y[i]-y(x[i],a))/dy[i]^2 +{j,k} (a[j]-a0[j])*(a[k]-a0[k])*sigma(j,k)

This function take into account multipliticty factors y[i].num, i.e. the numbers of measures used to determine y[i].mean This is used as a weight factor!

7.83.3.3 float golden_rule (float(*)(float *, mdp_int, void *) *fp*, float & *xmin*, float *ax*, float *bx*, float *cx*, float *tol* = 0.001, mdp_int *niter* = 100, void * *dummy* = 0)

finds x=xmin that minimizes (*fp)(&x,1,dummy) must be: (*fp)(&ax) > (*fp)(&bx) && (*fp)(&cx) > (*fp)(&bx)

7.83.3.4 void linear_fit (float * *x*, Measure * *y*, mdp_int *i0*, mdp_int *in*, Measure * *a*)

Fits y[i], x[i] for i0<=i<in with y=a[0]*x+a[1].

7.84 /Users/mdipierro/fermiqcd/development/Libraries/mdp_-global_vars.h File Reference

Typedefs

- typedef unsigned int uint
- typedef float mdp_real
- typedef int mdp_int

Functions

- void _mpi_error_message (string, string, int)

Variables

- const int EVEN = 0
- const int ODD = 1
- const int EVENODD = 2
- const int _NprocMax_ = 256
- double PRECISION = 3.0e-6
- char * mdp_program_name = "A generic test program"

  Each program should have a name.

- char * mdp_random_seed_filename = 0

  Filename to store the random seed.

- const unsigned int mdp_local_endianess = 0x87654321

  Used to determine the local endianess of this machine.

- const double Pi = 3.14159265358979323846264338327950288419 71
- bool mdp_shutup = false
- double mdp_precision = 1e-5

7.84.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

MDP global variables

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.84.2 Typedef Documentation

7.84.2.1 typedef int mdp_int

7.84.2.2 typedef float mdp_real

7.84.2.3 typedef unsigned int uint

7.84.3 Function Documentation

7.84.3.1 void _mpi_error_message (string, string, int)

7.84.4 Variable Documentation

7.84.4.1 const int _NprocMax_ = 256

7.84.4.2 const int EVEN = 0

7.84.4.3 const int EVENODD = 2

7.84.4.4 const unsigned int mdp_local_endianess = 0x87654321

Used to determine the local endianess of this machine.

7.84.4.5 double mdp_precision = 1e-5

Default precision used by iterative algorithms such as mdp_matrix::sin(), mdp_matrix::cos() and mdp_matrix::exp()

7.84.4.6 char* mdp_program_name = "A generic test program"

Each program should have a name.

7.84.4.7 char* mdp_random_seed_filename = 0

Filename to store the random seed.

7.84.4.8 bool mdp_shutup = false

Set mdp_shutup=true to suppress default output from any part of The program

7.84.4.9 const int ODD = 1

7.84.4.10 const double Pi = 3.14159265358979323846264338327950288419 71

7.84.4.11 double PRECISION = 3.0e-6

7.85 /Users/mdipierro/fermiqcd/development/Libraries/mdp_-header.h File Reference

7.85.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Useless

7.86 /Users/mdipierro/fermiqcd/development/Libraries/mdp_jackboot.h File Reference

Classes

- class mdp_jackboot

  coniatiner class for jackknife and boostrap analysis

Functions

- float mdp_jackboot_plain (float *x, void *a)

7.86.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_jackboot

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.86.2 Function Documentation

7.86.2.1 float mdp_jackboot_plain (float * *x*, void * *a*)

7.87 /Users/mdipierro/fermiqcd/development/Libraries/mdp_lattice.h File Reference

Classes

- class mdp_lattice
  distributed lattice object

Defines

- #define MDP_LATTICE

Variables

- const mdp_int NOWHERE = INT_MAX

7.87.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_lattice

7.87.2 Define Documentation

7.87.2.1 #define MDP_LATTICE

7.87.3 Variable Documentation

7.87.3.1 const mdp_int NOWHERE = INT_MAX

7.88 /Users/mdipierro/fermiqcd/development/Libraries/mdp_log.h File Reference

Classes

- class mdp_log

  base class of class mdp_communicator (DO NOT INSTANTIATE)

7.88.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_log

7.89 /Users/mdipierro/fermiqcd/development/Libraries/mdp_macros.h File Reference

Defines

- #define CHECK_ALL
- #define MDP_MPI
- #define INCLUDE_DEPRECATED_IO
- #define forallsites(x) for(x.start(); x.is_in(); x.next())

  Loop on all local siltes of this process.

- #define forallsitesofparity(x, pofx)
- #define forallsitesandcopies(x) for(x.start(), x.idx=0; x.idx<x.lattice().nvol; x.idx++)

  Loop on all sites stored by this process.

- #define forallsitesandcopiesofparity(x, pofx)

  Loop on all sites stored by this process with given parity.

- #define ME mpi.me()

  Returns the unique id of this process.

- #define Nproc mpi.nproc()

  Returns the total number of parallel processes for this job.

- #define error(a) _mpi_error_message(a,__FILE__, __LINE__);

  Reports a runtime error and the line that caused it.

- #define TRUE true
- #define FALSE false

7.89.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_macros

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.89.2 Define Documentation

7.89.2.1 #define CHECK_ALL

7.89.2.2 #define error(a) _mpi_error_message(a,__FILE__, __LINE__);

Reports a runtime error and the line that caused it.

7.89.2.3 #define FALSE false

7.89.2.4 #define forallsites(x) for(x.start(); x.is_in(); x.next())

Loop on all local siltes of this process.

7.89.2.5 #define forallsitesandcopies(x) for(x.start(), x.idx=0; x.idx<x.lattice().nvol; x.idx++)

Loop on all sites stored by this process.

7.89.2.6 #define forallsitesandcopiesofparity(x, pofx)

Value:

```
for(int __process=0; __process<Nproc; __process++)          \
     for(x.start(), x.idx=x.lattice().start[__process][pofx % 2];           \
         x.idx<x.lattice().stop[__process][(pofx+(pofx % 2))/2]; \
         x.idx++)
```

Loop on all sites stored by this process with given parity.

7.89.2.7 #define forallsitesofparity(x, pofx)

Value:

```
for(x.start(), x.idx=x.lattice().start[ME][pofx % 2];               \
      x.idx<x.lattice().stop[ME][(pofx+(pofx % 2))/2];     \
      x.idx++)
```

Loop on all local sites of this process with given parity If pofx is EVENODD=2 then loops on even and odd sites

7.89.2.8 #define INCLUDE_DEPRECATED_IO

7.89.2.9 #define MDP_MPI

7.89.2.10 #define ME mpi.me()

Returns the unique id of this process.

7.89.2.11 #define Nproc mpi.nproc()

Returns the total number of parallel processes for this job.

7.89.2.12 #define TRUE true

7.90 /Users/mdipierro/fermiqcd/development/Libraries/mdp_matrix.h File Reference

Classes

- class mdp_matrix

  matrices of complex numbers

Functions

- ostream & operator<< (ostream &os, const mdp_matrix &a)
- void print (const mdp_matrix &a)
- void prepare (mdp_matrix &a)
- mdp_matrix operator+ (const mdp_matrix &a)
- mdp_matrix operator- (const mdp_matrix &a)
- mdp_matrix operator+ (const mdp_matrix &x, const mdp_matrix &y)
- mdp_matrix operator- (const mdp_matrix &x, const mdp_matrix &y)
- mdp_matrix operator* (const mdp_matrix &x, const mdp_matrix &y)
- mdp_matrix operator/ (const mdp_matrix &a, const mdp_matrix &b)
- mdp_matrix operator+ (const mdp_matrix &a, mdp_complex b)
- mdp_matrix operator- (const mdp_matrix &a, mdp_complex b)
- mdp_matrix operator* (const mdp_matrix &y, mdp_complex x)
- mdp_matrix operator/ (const mdp_matrix &a, mdp_complex b)
- mdp_matrix operator+ (mdp_complex b, const mdp_matrix &a)
- mdp_matrix operator- (mdp_complex b, const mdp_matrix &a)
- mdp_matrix operator* (mdp_complex x, const mdp_matrix &y)
- mdp_matrix operator/ (mdp_complex b, const mdp_matrix &a)
- mdp_matrix operator+ (const mdp_matrix &a, mdp_real b)
- mdp_matrix operator- (const mdp_matrix &a, mdp_real b)
- mdp_matrix operator* (const mdp_matrix &y, mdp_real x)
- mdp_matrix operator/ (const mdp_matrix &a, mdp_real b)
- mdp_matrix operator+ (mdp_real b, const mdp_matrix &a)
- mdp_matrix operator- (mdp_real b, const mdp_matrix &a)
- mdp_matrix operator* (mdp_real a, const mdp_matrix &b)
- mdp_matrix mdp_identity (uint i)
- mdp_matrix mdp_zero (uint i)
- mdp_real max (const mdp_matrix &a)
- mdp_matrix submatrix (const mdp_matrix &a, uint i, uint j)
- mdp_complex det (const mdp_matrix &a)
- mdp_matrix inv (const mdp_matrix &a)
- mdp_matrix pow (const mdp_matrix &a, int i)
- mdp_matrix exp (const mdp_matrix &a)
- mdp_matrix log (const mdp_matrix &a)
- mdp_matrix sin (const mdp_matrix &a)
- mdp_matrix cos (const mdp_matrix &a)
- mdp_complex trace (const mdp_matrix &a)
- mdp_matrix transpose (const mdp_matrix &a)
- mdp_matrix hermitian (const mdp_matrix &a)
- mdp_matrix conj (const mdp_matrix &a)

7.90.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_matrix

7.90.2 Function Documentation

7.90.2.1 mdp_matrix conj (const mdp_matrix & *a*) [inline]

7.90.2.2 mdp_matrix cos (const mdp_matrix & *a*)

7.90.2.3 mdp_complex det (const mdp_matrix & *a*) [inline]

7.90.2.4 mdp_matrix exp (const mdp_matrix & *a*) [inline]

7.90.2.5 mdp_matrix hermitian (const mdp_matrix & *a*) [inline]

7.90.2.6 mdp_matrix inv (const mdp_matrix & *a*) [inline]

7.90.2.7 mdp_matrix log (const mdp_matrix & *a*)

7.90.2.8 mdp_real max (const mdp_matrix & *a*) [inline]

7.90.2.9 mdp_matrix mdp_identity (uint *i*) [inline]

7.90.2.10 mdp_matrix mdp_zero (uint *i*) [inline]

7.90.2.11 mdp_matrix operator∗ (mdp_real *a*, const mdp_matrix & *b*) [inline]

7.90.2.12 mdp_matrix operator∗ (const mdp_matrix & *y*, mdp_real *x*) [inline]

7.90.2.13 mdp_matrix operator∗ (mdp_complex *x*, const mdp_matrix & *y*) [inline]

7.90.2.14 mdp_matrix operator∗ (const mdp_matrix & *y*, mdp_complex *x*) [inline]

7.90.2.15 mdp_matrix operator∗ (const mdp_matrix & *x*, const mdp_matrix & *y*) [inline]

7.90.2.16 mdp_matrix operator+ (mdp_real *b*, const mdp_matrix & *a*) [inline]

7.90.2.17 mdp_matrix operator+ (const mdp_matrix & *a*, mdp_real *b*) [inline]

7.90.2.18 mdp_matrix operator+ (mdp_complex *b*, const mdp_matrix & *a*) [inline]

7.90.2.19 mdp_matrix operator+ (const mdp_matrix & *a*, mdp_complex *b*) [inline]

7.90.2.20 mdp_matrix operator+ (const mdp_matrix & *x*, const mdp_matrix & *y*) [inline]

7.90.2.21 mdp_matrix operator+ (const mdp_matrix & *a*) [inline]

7.90.2.22 mdp_matrix operator- (mdp_real *b*, const mdp_matrix & *a*) [inline]

7.90.2.23 mdp_matrix operator- (const mdp_matrix & *a*, mdp_real *b*) [inline]

7.90.2.24 mdp_matrix operator- (mdp_complex *b*, const mdp_matrix & *a*) [inline]

7.90.2.25 mdp_matrix operator- (const mdp_matrix & *a*, mdp_complex *b*) [inline]

7.90.2.26 mdp_matrix operator- (const mdp_matrix & *x*, const mdp_matrix & *y*) [inline]

7.90.2.27 mdp_matrix operator- (const mdp_matrix & *a*) [inline]

7.90.2.28 mdp_matrix operator/ (const mdp_matrix & *a*, mdp_real *b*) [inline]

7.90.2.29 mdp_matrix operator/ (mdp_complex *b*, const mdp_matrix & *a*) [inline]

7.90.2.30 mdp_matrix operator/ (const mdp_matrix & *a*, mdp_complex *b*) [inline]

7.91 /Users/mdipierro/fermiqcd/development/Libraries/mdp_matrix_field.h File Reference

Classes

- class mdp_matrix_field
  a field of matrices

7.91.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_matrix_field

7.92 /Users/mdipierro/fermiqcd/development/Libraries/mdp_matrix_test.h File Reference

Functions

- bool mdp_matrix_test ()

  For debugging only.

7.92.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

For debugging only

7.92.2 Function Documentation

7.92.2.1 bool mdp_matrix_test ()

For debugging only.

7.93 /Users/mdipierro/fermiqcd/development/Libraries/mdp_measure.h File Reference

Classes

- class mdp_measure

  implements error propagation

7.93.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_measure

7.94 /Users/mdipierro/fermiqcd/development/Libraries/mdp_mod2sign.h File Reference

Functions

- int mdp_mod2sign (int x)

  Returns +1 is x2==0 -1 otherwise.

7.94.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains function mdp_mod2sign

7.94.2 Function Documentation

7.94.2.1 int mdp_mod2sign (int *x*)

Returns +1 is x2==0 -1 otherwise.

7.95 /Users/mdipierro/fermiqcd/development/Libraries/mdp_nmatrix_field.h File Reference

Classes

- class mdp_nmatrix_field

  field of vectors of matrices

7.95.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_nmatrix_field

7.96 /Users/mdipierro/fermiqcd/development/Libraries/mdp_nvector_field.h File Reference

Classes

- class mdp_nvector_field

  field of vectors of vectors (DEPRECATED)

7.96.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_nvector_field (deprecated)

7.97 /Users/mdipierro/fermiqcd/development/Libraries/mdp_-partitionings.h File Reference

Functions

- int default_partitioning0 (int *x, int ndim, int *nx)
- template<int dim>
  int default_partitioning (int *x, int ndim, int *nx)

7.97.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Example functions to do parallel partitioning of a lattice

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.97.2 Function Documentation

7.97.2.1 template<int dim> int default_partitioning (int * *x*, int *ndim*, int * *nx*) [inline]

7.97.2.2 int default_partitioning0 (int * *x*, int *ndim*, int * *nx*)

7.98 /Users/mdipierro/fermiqcd/development/Libraries/mdp_-permutations.h File Reference

Functions

- mdp_int mdp_permutations (int n)
- void mdp_permutation_sort (int map[ ], int k)
- int mdp_permutation (int n, int k, int i)

7.98.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Functions to compute permutations

7.98.2 Function Documentation

7.98.2.1 int mdp_permutation (int *n*, int *k*, int *i*)

Returns j-th element of the k-th permutations of n numbers For example if n=4 [0123] k=0 [0132] k=1 ... [3210] k=23 Returns -1 on error when (i>n || k>n_permutations(n))

7.98.2.2 void mdp_permutation_sort (int *map*[ ], int *k*)

7.98.2.3 mdp_int mdp_permutations (int *n*)

7.99 /Users/mdipierro/fermiqcd/development/Libraries/mdp_postscript.h File Reference

Classes

- class mdp_postscript

  to output and draw in postscript

7.99.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Yes...MDP can print and draw in postscript

7.100 /Users/mdipierro/fermiqcd/development/Libraries/mdp_prng.h File Reference

Classes

- class mdp_prng

  Marsaglia's random number generator (same as UKQCD).

Variables

- class mdp_prng mdp_random

  Marsaglia's random number generator (same as UKQCD).

7.100.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Class mdp_prng (the random number generator of MDP)

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.100.2 Variable Documentation

7.100.2.1 class mdp_prng mdp_random

Marsaglia's random number generator (same as UKQCD). You should not instantiate this class because:

- there is a global object mdp_random
- each field "lattice" has a parallel generator "lattice.random(x)" Example:

```
///    // print a uniform number in (0,1)
///    cout << mdp_random.plain() << endl;
///    // print a gaussian number
///    cout << mdp_random.gaussian() << endl;
///    // print a random SU(10) matrix
///    cout << mdp_random.SU(10) << endl;
///
```

7.101 /Users/mdipierro/fermiqcd/development/Libraries/mdp_-prng_sfmt.h File Reference

```
#include "assert.h"
```

Classes

- class mdp_prng_sfmt
- struct **mdp_prng_sfmt::W128_T**

Defines

- #define MSK1 0xdfffffefU
- #define MSK2 0xddfecb7fU
- #define MSK3 0xbffaffffU
- #define MSK4 0xbfffff6U
- #define PARITY1 0x00000001U
- #define PARITY2 0x00000000U
- #define PARITY3 0x00000000U
- #define PARITY4 0x13c9e684U

7.101.1 Define Documentation

7.101.1.1 #define MSK1 0xdfffffefU

7.101.1.2 #define MSK2 0xddfecb7fU

7.101.1.3 #define MSK3 0xbffaffffU

7.101.1.4 #define MSK4 0xbfffff6U

7.101.1.5 #define PARITY1 0x00000001U

7.101.1.6 #define PARITY2 0x00000000U

7.101.1.7 #define PARITY3 0x00000000U

7.101.1.8 #define PARITY4 0x13c9e684U

7.102 /Users/mdipierro/fermiqcd/development/Libraries/mdp_prompt.h File Reference

Functions

- double val (string s)

  Converts string to float.

- string prompt (string filename, string variable, string def_val="0.0", int p=0)

Variables

- const char STD_INPUT [ ] = ""
- const char STD_INPUT_FILE [ ] = "<stdin>"

7.102.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Functions to parse user input of parameters in a way safe to parallel programs

7.102.2 Function Documentation

7.102.2.1 string prompt (string *filename*, string *variable*, string *def_val* = "0.0", int *p* = 0)

Try prompt("<stdin>","VALUE","4.0") It will prompt the user for variable VALUE and take 4.0 as default

7.102.2.2 double val (string *s*)

Converts string to float.

7.102.3 Variable Documentation

7.102.3.1 const char STD_INPUT[ ] = ""

7.102.3.2 const char STD_INPUT_FILE[ ] = "<stdin>"

7.103 /Users/mdipierro/fermiqcd/development/Libraries/mdp_psim.h File Reference

```
#include "cstdio"
#include "cstdlib"
#include "string"
#include "iostream"
#include "vector"
#include "map"
#include <sys/file.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/socket.h>
#include <fcntl.h>
```

Classes

- class mdp_psim

  Parallel SIMulator used by class mdp_communicator.

7.103.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_psim (the parallel simulator)

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.104 /Users/mdipierro/fermiqcd/development/Libraries/mdp_save_partitioning_vtk.h File Reference

Functions

- void save_partitioning_vtk (mdp_lattice &lattice, string filename)

7.104.1 Function Documentation

7.104.1.1 void save_partitioning_vtk (mdp_lattice & *lattice*, string *filename*)

7.105 /Users/mdipierro/fermiqcd/development/Libraries/mdp_sfmt.cpp File Reference

```
#include "assert.h"
#include "iostream"
#include "cmath"
#include "mdp.h"
```

Classes

- class MDP_SFMT19937
- struct **MDP_SFMT19937::W128_T**

Defines

- #define MSK1 0xdfffffefU
- #define MSK2 0xddfecb7fU
- #define MSK3 0xbffaffffU
- #define MSK4 0xbfffff6U
- #define PARITY1 0x00000001U
- #define PARITY2 0x00000000U
- #define PARITY3 0x00000000U
- #define PARITY4 0x13c9e684U

Functions

- int main ()

7.105.1 Define Documentation

7.105.1.1 #define MSK1 0xdfffffefU

7.105.1.2 #define MSK2 0xddfecb7fU

7.105.1.3 #define MSK3 0xbffaffffU

7.105.1.4 #define MSK4 0xbfffffff6U

7.105.1.5 #define PARITY1 0x00000001U

7.105.1.6 #define PARITY2 0x00000000U

7.105.1.7 #define PARITY3 0x00000000U

7.105.1.8 #define PARITY4 0x13c9e684U

7.105.2 Function Documentation

7.105.2.1 int main ()

7.106 /Users/mdipierro/fermiqcd/development/Libraries/mdp_site.h File Reference

Classes

- class mdp_site

  site object to loop on a lattice

Functions

- int on_which_process (mdp_lattice &a, int x0=0, int x1=0, int x2=0, int x3=0, int x4=0, int x5=0, int x6=0, int x7=0, int x8=0, int x9=0)
- int in_block (mdp_site x)
- ostream & operator<< (ostream &os, mdp_site &x)

7.106.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains delcaration of class mdp_site for complex numbers

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.106.2 Function Documentation

7.106.2.1 int in_block (mdp_site *x*) [inline]

When compiled with TWISTED_BOUNDARY the mdp_site class keeps track of sites that moved around the boundary of the torus topology. this function Returns false if this is one such site, true otherwise.

7.106.2.2 int on_which_process (mdp_lattice & *a*, int *x0* = 0, int *x1* = 0, int *x2* = 0, int *x3* = 0, int *x4* = 0, int *x5* = 0, int *x6* = 0, int *x7* = 0, int *x8* = 0, int *x9* = 0)

checks which process of the lattice a stores locally the site of coordinates x0,x1,x2,...,x9 to be used before calling mdp_site::set() (note: prototyping of friend functions is required by some compilers)

7.106.2.3 ostream& operator<< (ostream & *os*, mdp_site & *x*)

7.107 /Users/mdipierro/fermiqcd/development/Libraries/mdp_swap.h File Reference

Functions

- template<class T >
  void swap (T &a, T &b)
- template<class T >
  void swap (T *a, T *b, int n)

7.107.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains swap function

7.107.2 Function Documentation

7.107.2.1 template<class T > void swap (T * *a*, T * *b*, int *n*) [inline]

7.107.2.2 template<class T > void swap (T & *a*, T & *b*) [inline]

7.108 /Users/mdipierro/fermiqcd/development/Libraries/mdp_timer.h File Reference

Functions

- double walltime ()
- string getname ()
- void getcpuusage (double &user, double &total)

7.108.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains timeing functions including functions to get cpu usage

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.108.2 Function Documentation

7.108.2.1 void getcpuusage (double & *user*, double & *total*)

7.108.2.2 string getname ()

7.108.2.3 double walltime ()

7.109 /Users/mdipierro/fermiqcd/development/Libraries/mdp_topologies.h File Reference

Functions

- void torus_topology (int mu, int *x_dw, int *x, int *x_up, int ndim, int *nx)
- void box_topology (int mu, int *x_dw, int *x, int *x_up, int ndim, int *nx)
- void moebious_topolgy (int mu, int *x_dw, int *x, int *x_up, int ndim, int *nx)

7.109.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Examples of lattice topologies

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.109.2 Function Documentation

7.109.2.1 void box_topology (int *mu*, int * *x_dw*, int * *x*, int * *x_up*, int *ndim*, int * *nx*)

7.109.2.2 void moebious_topolgy (int *mu*, int * *x_dw*, int * *x*, int * *x_up*, int *ndim*, int * *nx*)

7.109.2.3 void torus_topology (int *mu*, int * *x_dw*, int * *x*, int * *x_up*, int *ndim*, int * *nx*)

7.110 /Users/mdipierro/fermiqcd/development/Libraries/mdp_utils.h File Reference

Functions

- string tostring (int k)
- vector< string > glob (string pattern)
- string latest_file (string pattern)
- string next_to_latest_file (string pattern)
- string tostring (float k)
- int is_file (string filename, char permission[ ]="r")
- mdp_field_file_header get_info (string filename, int proc=0)
- int mail (string email, string message)
- int mail_file (string email, string filename)

7.110.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Other junk that did not fit anywhere else

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.110.2 Function Documentation

7.110.2.1 mdp_field_file_header get_info (string *filename*, int *proc* = 0)

7.110.2.2 vector<string> glob (string *pattern*)

7.110.2.3 int is_file (string *filename*, char *permission*[ ] = "r")

7.110.2.4 string latest_file (string *pattern*)

7.110.2.5 int mail (string *email*, string *message*)

7.110.2.6 int mail_file (string *email*, string *filename*)

7.110.2.7 string next_to_latest_file (string *pattern*)

7.110.2.8 string tostring (float *k*)

7.110.2.9 string tostring (int *k*)

7.111 /Users/mdipierro/fermiqcd/development/Libraries/mdp_vector.h File Reference

Classes

- class mdp_vector

  discerete vectors to navigate on a lattice

Functions

- mdp_vector binary2versor (mdp_int a)
- int versor2binary (int x0, int x1=0, int x2=0, int x3=0, int x4=0, int x5=0, int x6=0, int x7=0, int x8=0, int x9=0)
- mdp_int vector2binary (mdp_vector v)

7.111.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_vector

Licensed under GPL2 license Read attached license in file mdp_license.pdf This file cannot be distributed without file mdp_license.pdf

7.111.2 Function Documentation

7.111.2.1 mdp_vector binary2versor (mdp_int *a*) [inline]

7.111.2.2 mdp_int vector2binary (mdp_vector *v*) [inline]

7.111.2.3 int versor2binary (int *x0*, int *x1* = 0, int *x2* = 0, int *x3* = 0, int *x4* = 0, int *x5* = 0, int *x6* = 0, int *x7* = 0, int *x8* = 0, int *x9* = 0) [inline]

7.112 /Users/mdipierro/fermiqcd/development/Libraries/mdp_vector_field.h File Reference

Classes

- class mdp_vector_field

  a field of vectors of complex numbers

7.112.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_vector_field

7.113 /Users/mdipierro/fermiqcd/development/Libraries/mdp_version.h File Reference

Variables

- const char mdp_version [ ] = "MDP version 4.0"

7.113.1 Detailed Description

Version:

2009-12-21

Author:

Massimo Di Pierro <mdipierro@cs.depaul.edu>

Contains class mdp_vector

7.113.2 Variable Documentation

7.113.2.1 const char mdp_version[ ] = "MDP version 4.0"

7.114 /Users/mdipierro/fermiqcd/development/Libraries/searchandreplace.py File Reference

Namespaces

- namespace searchandreplace

Variables

- tuple searchandreplace::sin = raw_input('pattern to replace: ')
- tuple searchandreplace::sout = raw_input('replace with: ')
- string searchandreplace::choice = 'y'
- tuple searchandreplace::file = open(filename,'r')
- string searchandreplace::s = ''
- tuple searchandreplace::line = line.replace(sin,sout)

7.115 /Users/mdipierro/fermiqcd/development/Libraries/searchandreplace2.py File Reference

Namespaces

- namespace searchandreplace2

Variables

- string searchandreplace2::choice = 'y'
- tuple searchandreplace2::file = open(filename,'r')
- string searchandreplace2::s = ''
- list searchandreplace2::c = line[4:5]
- list searchandreplace2::line2 = line[:4]
- searchandreplace2::line = line2