SpinDec

Generated by Doxygen 1.6.1

Thu Jun 25 19:25:50 2015

Contents

1 Class Index **1**

1.1 Class Hierarchy 1

2 Class Index **3**

2.1 Class List 3

3 File Index **5**

3.1 File List 5

4 Class Documentation **7**

4.1 SpinDec::AdiabaticLabel Class Reference 7

4.1.1 Detailed Description 7

4.2 SpinDec::BoostEigen Class Reference 9

4.2.1 Detailed Description 9

4.2.2 Member Function Documentation 9

4.2.2.1 cosAngleBetween 9

4.2.2.2 maxAbsCoeff 10

4.2.2.3 partialTrace 10

4.2.2.4 spectralDecomposition 10

4.2.2.5 tensorProduct 10

4.2.2.6 tensorProduct 10

4.2.2.7 unitarySpectralDecomposition 10

4.3 SpinDec::CCE Class Reference 11

4.3.1 Detailed Description 11

4.3.2 Member Function Documentation 12

4.3.2.1 calculate 12

4.4 SpinDec::Cluster Class Reference 13

4.4.1 Detailed Description 13

4.5 SpinDec::ClusterDatabase Class Reference 14

4.5.1 Detailed Description . . . 15
4.6 SpinDec::ClusterDatabaseEntry Class Reference . . . 16
4.6.1 Detailed Description . . . 16
4.7 SpinDec::Constants Class Reference . . . 17
4.7.1 Detailed Description . . . 17
4.7.2 Member Data Documentation . . . 17
4.7.2.1 kElectronGyromagneticRatio . . . 17
4.7.2.2 kPi . . . 17
4.7.2.3 kPlanck . . . 17
4.7.2.4 kReducedPlanck . . . 17
4.8 SpinDec::CPMG Class Reference . . . 18
4.8.1 Detailed Description . . . 18
4.9 SpinDec::CPMGDephasing Class Reference . . . 19
4.9.1 Detailed Description . . . 19
4.10 SpinDec::CrystalBasis Class Reference . . . 20
4.10.1 Detailed Description . . . 20
4.11 SpinDec::CrystalStructure Class Reference . . . 21
4.11.1 Detailed Description . . . 22
4.11.2 Member Function Documentation . . . 22
4.11.2.1 fill_site_vectors . . . 22
4.12 SpinDec::CSDProblem Class Reference . . . 23
4.12.1 Detailed Description . . . 23
4.12.2 Member Data Documentation . . . 24
4.12.2.1 system_bath_edges_ . . . 24
4.13 SpinDec::DensityOperator Class Reference . . . 25
4.13.1 Detailed Description . . . 25
4.14 SpinDec::DiamondCubic Class Reference . . . 26
4.14.1 Detailed Description . . . 26
4.14.2 Member Function Documentation . . . 26
4.14.2.1 int_range_centred_cube . . . 26
4.15 SpinDec::Dipolar Class Reference . . . 27
4.15.1 Detailed Description . . . 27
4.16 SpinDec::Eigenspectrum Class Reference . . . 29
4.16.1 Detailed Description . . . 29
4.17 SpinDec::ElectronSpinParameters Class Reference . . . 30
4.17.1 Detailed Description . . . 30

4.18 SpinDec::Errors Class Reference . . . 31
4.18.1 Detailed Description . . . 31
4.19 SpinDec::EvolutionOperator Class Reference . . . 32
4.19.1 Detailed Description . . . 32
4.19.2 Constructor & Destructor Documentation . . . 32
4.19.2.1 EvolutionOperator . . . 32
4.19.3 Member Data Documentation . . . 33
4.19.3.1 eigenvalues_ . . . 33
4.19.3.2 eigenvectors_ . . . 33
4.20 SpinDec::FileProperties Class Reference . . . 34
4.20.1 Detailed Description . . . 34
4.21 SpinDec::FreeEvolution Class Reference . . . 35
4.21.1 Detailed Description . . . 35
4.22 SpinDec::HermitianEigenspectrum Class Reference . . . 36
4.22.1 Detailed Description . . . 36
4.22.2 Member Function Documentation . . . 36
4.22.2.1 spectralDecomposition . . . 36
4.23 SpinDec::Hyperfine Class Reference . . . 37
4.23.1 Detailed Description . . . 37
4.24 SpinDec::HyperfineParameters Class Reference . . . 39
4.24.1 Detailed Description . . . 39
4.25 SpinDec::IdentityOperator Class Reference . . . 40
4.25.1 Detailed Description . . . 40
4.26 SpinDec::IdentityPulse Class Reference . . . 41
4.26.1 Detailed Description . . . 41
4.27 SpinDec::LatticeVectors Class Reference . . . 42
4.27.1 Detailed Description . . . 42
4.28 SpinDec::MatrixRepresentation Class Reference . . . 43
4.28.1 Detailed Description . . . 43
4.29 SpinDec::PiPulse Class Reference . . . 44
4.29.1 Detailed Description . . . 44
4.29.2 Constructor & Destructor Documentation . . . 44
4.29.2.1 PiPulse . . . 44
4.30 SpinDec::Pulse Class Reference . . . 45
4.30.1 Detailed Description . . . 45
4.31 SpinDec::PulseExperiment Class Reference . . . 46

4.31.1 Detailed Description . . . 46
4.32 SpinDec::PulseSequence Class Reference . . . 47
4.32.1 Detailed Description . . . 47
4.33 SpinDec::PulseSequenceBase Class Reference . . . 48
4.33.1 Detailed Description . . . 48
4.34 SpinDec::RandomNumberGenerator Class Reference . . . 49
4.34.1 Detailed Description . . . 49
4.34.2 Member Function Documentation . . . 49
4.34.2.1 uniform_c_rand . . . 49
4.35 SpinDec::ReducedProblem Class Reference . . . 50
4.35.1 Detailed Description . . . 50
4.36 SpinDec::Sign Class Reference . . . 51
4.36.1 Detailed Description . . . 51
4.37 SpinDec::SimpleCubicLatticeVectors Class Reference . . . 52
4.37.1 Detailed Description . . . 52
4.38 SpinDec::SpinBasis Class Reference . . . 53
4.38.1 Detailed Description . . . 54
4.38.2 Member Function Documentation . . . 54
4.38.2.1 operator+ . . . 54
4.38.2.2 operator^ . . . 54
4.39 SpinDec::SpinBath Class Reference . . . 55
4.39.1 Detailed Description . . . 55
4.39.2 Member Function Documentation . . . 56
4.39.2.1 get_bath_state . . . 56
4.39.3 Member Data Documentation . . . 56
4.39.3.1 intrabath_edges_ . . . 56
4.39.3.2 spin_system_base_ . . . 56
4.40 SpinDec::SpinDonor Class Reference . . . 57
4.40.1 Detailed Description . . . 59
4.40.2 Constructor & Destructor Documentation . . . 60
4.40.2.1 SpinDonor . . . 60
4.40.3 Member Function Documentation . . . 60
4.40.3.1 build_basis . . . 60
4.40.3.2 build_basis . . . 60
4.40.3.3 build_basis . . . 60
4.40.3.4 build_truncated_basis . . . 60

4.40.3.5 calc_adiabatic_level_labels . . . 61
4.40.3.6 energy . . . 61
4.40.3.7 orthogonal_adiabatic_level_label . . . 61
4.40.3.8 orthogonal_level_label . . . 61
4.40.3.9 polarization . . . 61
4.40.3.10 polarization . . . 61
4.40.3.11 set_transition . . . 61
4.40.4 Member Data Documentation . . . 62
4.40.4.1 adiabatic_level_labels_ . . . 62
4.40.4.2 complete_basis_ . . . 62
4.40.4.3 orthogonal_level_labels_ . . . 62
4.40.4.4 sorted_level_labels_ . . . 62
4.40.4.5 transition_level_labels_ . . . 62
4.41 SpinDec::SpinDown Class Reference . . . 63
4.41.1 Detailed Description . . . 63
4.41.2 Member Function Documentation . . . 63
4.41.2.1 init . . . 63
4.42 SpinDec::SpinHalf Class Reference . . . 64
4.42.1 Detailed Description . . . 64
4.43 SpinDec::SpinHalfParameters Class Reference . . . 65
4.43.1 Detailed Description . . . 65
4.44 SpinDec::SpinHalfStates Class Reference . . . 66
4.44.1 Detailed Description . . . 66
4.44.2 Member Function Documentation . . . 66
4.44.2.1 init . . . 66
4.45 SpinDec::SpinHamiltonian Class Reference . . . 67
4.45.1 Detailed Description . . . 68
4.45.2 Member Function Documentation . . . 68
4.45.2.1 fill_zeeman . . . 68
4.46 SpinDec::SpinInteraction Class Reference . . . 69
4.46.1 Detailed Description . . . 69
4.46.2 Member Function Documentation . . . 70
4.46.2.1 fill_ising_flipflop . . . 70
4.47 SpinDec::SpinInteractionEdge Class Reference . . . 71
4.47.1 Detailed Description . . . 71
4.48 SpinDec::SpinInteractionGraph Class Reference . . . 72

4.48.1 Detailed Description . . . 73
4.48.2 Member Function Documentation . . . 73
4.48.2.1 add_vertex . . . 73
4.48.2.2 add_vertex . . . 73
4.48.2.3 join . . . 73
4.48.2.4 join . . . 73
4.48.2.5 join_in_place . . . 73
4.48.2.6 join_in_place . . . 74
4.49 SpinDec::SpinInteractionVertex Class Reference . . . 75
4.49.1 Detailed Description . . . 75
4.50 SpinDec::SpinOperator Class Reference . . . 76
4.50.1 Detailed Description . . . 77
4.51 SpinDec::SpinParameters Class Reference . . . 78
4.51.1 Detailed Description . . . 78
4.52 SpinDec::SpinParametersVector Class Reference . . . 80
4.52.1 Detailed Description . . . 80
4.53 SpinDec::SpinState Class Reference . . . 81
4.53.1 Detailed Description . . . 82
4.54 SpinDec::SpinSystem Class Reference . . . 83
4.54.1 Detailed Description . . . 83
4.55 SpinDec::SpinSystemBase Class Reference . . . 84
4.55.1 Detailed Description . . . 85
4.55.2 Member Function Documentation . . . 85
4.55.2.1 print . . . 85
4.55.2.2 set_state . . . 85
4.56 SpinDec::SpinUp Class Reference . . . 86
4.56.1 Detailed Description . . . 86
4.56.2 Member Function Documentation . . . 86
4.56.2.1 init . . . 86
4.57 SpinDec::StringOptions Class Reference . . . 87
4.57.1 Detailed Description . . . 87
4.58 SpinDec::TimeArray Class Reference . . . 88
4.58.1 Detailed Description . . . 89
4.58.2 Member Function Documentation . . . 89
4.58.2.1 logarithmic_time . . . 89
4.59 SpinDec::TimeEvolution Class Reference . . . 90

4.59.1 Detailed Description 91
4.60 SpinDec::TwoStateSuperposition Class Reference 92
4.60.1 Detailed Description 92
4.61 SpinDec::UniformMagneticField Class Reference 93
4.61.1 Detailed Description 93

5 File Documentation **95**
5.1 /home/sbalian/spindec/include/SpinDec/typedefs.h File Reference 95
5.1.1 Detailed Description 96

Chapter 1

Class Index

1.1 Class Hierarchy

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

SpinDec::AdiabaticLabel . . . 7
SpinDec::BoostEigen . . . 9
SpinDec::CCE . . . 11
SpinDec::Cluster . . . 13
SpinDec::ClusterDatabase . . . 14
SpinDec::ClusterDatabaseEntry . . . 16
SpinDec::Constants . . . 17
SpinDec::CrystalBasis . . . 20
SpinDec::CrystalStructure . . . 21
  SpinDec::DiamondCubic . . . 26
SpinDec::CSDProblem . . . 23
SpinDec::Eigenspectrum . . . 29
  SpinDec::HermitianEigenspectrum . . . 36
SpinDec::Errors . . . 31
SpinDec::FileProperties . . . 34
SpinDec::HyperfineParameters . . . 39
SpinDec::LatticeVectors . . . 42
  SpinDec::SimpleCubicLatticeVectors . . . 52
SpinDec::MatrixRepresentation . . . 43
  SpinDec::SpinOperator . . . 76
    SpinDec::DensityOperator . . . 25
    SpinDec::EvolutionOperator . . . 32
    SpinDec::IdentityOperator . . . 40
    SpinDec::SpinHamiltonian . . . 67
  SpinDec::SpinState . . . 81
    SpinDec::SpinHalfStates . . . 66
      SpinDec::SpinDown . . . 63
      SpinDec::SpinUp . . . 86
    SpinDec::TwoStateSuperposition . . . 92
SpinDec::Pulse . . . 45
  SpinDec::FreeEvolution . . . 35
  SpinDec::IdentityPulse . . . 41

SpinDec::PiPulse . . . 44
SpinDec::PulseExperiment . . . 46
SpinDec::CPMGDephasing . . . 19
SpinDec::PulseSequenceBase . . . 48
SpinDec::CPMG . . . 18
SpinDec::PulseSequence . . . 47
SpinDec::RandomNumberGenerator . . . 49
SpinDec::ReducedProblem . . . 50
SpinDec::Sign . . . 51
SpinDec::SpinBasis . . . 53
SpinDec::SpinBath . . . 55
SpinDec::SpinInteraction . . . 69
SpinDec::Dipolar . . . 27
SpinDec::Hyperfine . . . 37
SpinDec::SpinInteractionEdge . . . 71
SpinDec::SpinInteractionGraph . . . 72
SpinDec::SpinInteractionVertex . . . 75
SpinDec::SpinParameters . . . 78
SpinDec::SpinHalfParameters . . . 65
SpinDec::ElectronSpinParameters . . . 30
SpinDec::SpinParametersVector . . . 80
SpinDec::SpinSystemBase . . . 84
SpinDec::SpinHalf . . . 64
SpinDec::SpinSystem . . . 83
SpinDec::SpinDonor . . . 57
SpinDec::StringOptions . . . 87
SpinDec::TimeArray . . . 88
SpinDec::TimeEvolution . . . 90
SpinDec::UniformMagneticField . . . 93

Chapter 2

Class Index

2.1 Class List

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

SpinDec::AdiabaticLabel (Adiabatic eigenstates of a spin donor) 7
SpinDec::BoostEigen (Static methods to extend Eigen functionality) 9
SpinDec::CCE (Solves for a CSD problem using the CCE) 11
SpinDec::Cluster (Contains labels for a cluster of spins) 13
SpinDec::ClusterDatabase (Stores clusters and associated complex time evolutions) 14
SpinDec::ClusterDatabaseEntry (Entry for ClusterDatabase) 16
SpinDec::Constants (Mathematical and physical constants) 17
SpinDec::CPMG (Carr-Purcell-Meiboom-Gill (CPMG) pulse sequence) 18
SpinDec::CPMGDephasing (CPMG pulse sequence) 19
SpinDec::CrystalBasis (Basis vectors for crystal structures in 3D) 20
SpinDec::CrystalStructure ($i\mathbf{a}_1 + j\mathbf{a}_2 + k\mathbf{a}_3 + \sum_n \mathbf{b}_n$) 21
SpinDec::CSDProblem (Central spin decoherence problem) 23
SpinDec::DensityOperator 25
SpinDec::DiamondCubic (Diamond cubic crystal structure) 26
SpinDec::Dipolar (Secular dipolar interaction strength between a pair of spins) 27
SpinDec::Eigenspectrum (Holds the eigenvectors and eigenvalues of a matrix) 29
SpinDec::ElectronSpinParameters (Parameters for an electron spin) 30
SpinDec::Errors (Error and warning handling) 31
SpinDec::EvolutionOperator (Free evolution operator) 32
SpinDec::FileProperties (Static methods for ASCII file properties) 34
SpinDec::FreeEvolution ((Pulse)-free evolution) 35
SpinDec::HermitianEigenspectrum (Diagonalizes a Hermitian matrix) 36
SpinDec::Hyperfine (Calculates the electron-nuclear hyperfine interaction in a lattice) 37
SpinDec::HyperfineParameters 39
SpinDec::IdentityOperator (Identity) 40
SpinDec::IdentityPulse (Identity pulse (no duration)) 41
SpinDec::LatticeVectors (Lattice vectors for 3D crystal structures) 42
SpinDec::MatrixRepresentation 43
SpinDec::PiPulse (π-pulse or refocusing pulse) 44
SpinDec::Pulse (Pulses in a pulse sequence) 45
SpinDec::PulseExperiment (Abstract base class for a pulse sequence experiment) 46
SpinDec::PulseSequence (Concrete general pulse sequence) 47
SpinDec::PulseSequenceBase (Abstract base class for pulse sequences) 48

SpinDec::RandomNumberGenerator (Static methods for generating random numbers) 49
SpinDec::ReducedProblem (SpinSystemBase and an order) 50
SpinDec::Sign (±) 51
SpinDec::SimpleCubicLatticeVectors (Simple cubic lattice vectors) 52
SpinDec::SpinBasis (Holds magnetic quantum numbers for (in general) multiple spins) 53
SpinDec::SpinBath (Spin bath (single spin species)) 55
SpinDec::SpinDonor (Special spin system: mixed electron-nuclear spin donors) 57
SpinDec::SpinDown (Spin down state) 63
SpinDec::SpinHalf (A spin-1/2 spin system) 64
SpinDec::SpinHalfParameters (Spin-1/2 spin parameters) 65
SpinDec::SpinHalfStates (Spin-1/2 spin states) 66
SpinDec::SpinHamiltonian (Effective spin Hamiltonian built from a spin interaction graph) . . . 67
SpinDec::SpinInteraction (Abstract base class for interaction between a pair of spins) 69
SpinDec::SpinInteractionEdge (Edge for a spin interaction graph) 71
SpinDec::SpinInteractionGraph (Spin interaction graph from which spin Hamiltonians are built) 72
SpinDec::SpinInteractionVertex (Vertex for a spin interaction graph) 75
SpinDec::SpinOperator (Quantum spin operator in the Zeeman basis) 76
SpinDec::SpinParameters (Contains basic parameters associated with a spin) 78
SpinDec::SpinParametersVector (Multiple SpinParameters container) 80
SpinDec::SpinState (Quantum spin state in the Zeeman basis) 81
SpinDec::SpinSystem (Diagonalizable concrete spin system) 83
SpinDec::SpinSystemBase (Abstract base class for spin systems) 84
SpinDec::SpinUp (Spin up state) 86
SpinDec::StringOptions (String flags composed of A-Z, a-z, each character representing a flag) 87
SpinDec::TimeArray (Time array in microseconds) 88
SpinDec::TimeEvolution (Time evolution of a complex variable) 90
SpinDec::TwoStateSuperposition (2-level superposition state) 92
SpinDec::UniformMagneticField (Uniform magnetic field parallel to some direction in Cartesian coordinates) 93

Chapter 3

File Index

3.1 File List

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

/home/sbalian/spindec/include/SpinDec/**AdiabaticLabel.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**base.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**BoostEigen.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**CCE.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**Cluster.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**ClusterDatabase.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**ClusterDatabaseEntry.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**Constants.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**CPMG.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**CPMGDephasing.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**CrystalBasis.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**CrystalStructure.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**CSDProblem.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**DensityOperator.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**DiamondCubic.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**Dipolar.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**Eigenspectrum.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**ElectronSpinParameters.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**Errors.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**EvolutionOperator.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**FileProperties.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**FreeEvolution.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**HermitianEigenspectrum.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**Hyperfine.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**HyperfineParameters.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**IdentityOperator.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**IdentityPulse.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**LatticeVectors.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**MatrixRepresentation.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**PiPulse.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**Pulse.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**PulseExperiment.h** . . . . . **??**
/home/sbalian/spindec/include/SpinDec/**PulseSequence.h** . . . . . **??**

/home/sbalian/spindec/include/SpinDec/**PulseSequenceBase.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**RandomNumberGenerator.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**ReducedProblem.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**Sign.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SimpleCubicLatticeVectors.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinBasis.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinBath.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinDonor.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinDown.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinHalf.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinHalfParameters.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinHalfStates.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinHamiltonian.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinInteraction.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinInteractionEdge.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinInteractionGraph.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinInteractionVertex.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinOperator.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinParameters.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinParametersVector.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinState.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinSystem.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinSystemBase.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**SpinUp.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**StringOptions.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**TimeArray.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**TimeEvolution.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/**TwoStateSuperposition.h** . . . . . ??
/home/sbalian/spindec/include/SpinDec/typedefs.h (Typedefs and "usings") 95
/home/sbalian/spindec/include/SpinDec/**UniformMagneticField.h** . . . . . ??

Chapter 4

Class Documentation

4.1 SpinDec::AdiabaticLabel Class Reference

Adiabatic eigenstates of a spin donor.

`#include <AdiabaticLabel.h>`

Public Member Functions

- **AdiabaticLabel** (const Sign &sign, const int quantum_number)
- const Sign & **get_sign** () const
- int **get_quantum_number** () const

Private Attributes

- Sign sign_
  Plus or minus.
- int quantum_number_
  $m = S + I$.

Friends

- std::ostream & operator<< (std::ostream &os, AdiabaticLabel const &label)
  Print.

4.1.1 Detailed Description

Adiabatic eigenstates of a spin donor. For a spin donor, the adiabatic energy eigenstates are labeled as follows:

$|\pm, m\rangle$, where $m = S + I$ is an integer, S and I are the electron and nuclear spin quantum numbers.

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

- /home/sbalian/spindec/include/SpinDec/AdiabaticLabel.h

4.2 SpinDec::BoostEigen Class Reference

Static methods to extend Eigen functionality.

`#include <BoostEigen.h>`

Static Public Member Functions

- static double cosAngleBetween (const ThreeVector &a, const ThreeVector &b)
  Cosine of angle between real vectors.
- static double maxAbsCoeff (const ThreeVector &a)
  Maximum absolute coefficient.
- static ComplexVector exp (const ComplexVector &a)
  Element-wise exponentiation for complex vectors.
- static ComplexMatrix tensorProduct (const ComplexMatrix &A, const ComplexMatrix &B)
  Tensor product for complex matrices.
- static ComplexVector tensorProduct (const ComplexVector &a, const ComplexVector &b)
  Tensor product for complex vectors.
- static ComplexMatrix partialTrace (const ComplexMatrix &AB, const unsigned int dimension_B)
  Partial trace for complex matrices.
- static ComplexMatrix spectralDecomposition (const ComplexMatrix &eigenvectors, const ComplexVector &eigenvalues)
  Spectral decomposition of a complex matrix.
- static ComplexMatrix unitarySpectralDecomposition (const ComplexMatrix &eigenvectors, const ComplexVector &eigenvalues)
  Spectral decomposition of a unitary matrix.
- static bool isWithinDistance (const ThreeVector &r, const double distance)
  Is $|\mathbf{r}| \leq d$*?*

4.2.1 Detailed Description

Static methods to extend Eigen functionality. Note that the naming convention complies with that of Eigen, and is different from the rest of SpinDec. Eigen is for linear algebra and can be obtained for free: http://eigen.tuxfamily.org/ .

4.2.2 Member Function Documentation

4.2.2.1 static double SpinDec::BoostEigen::cosAngleBetween (const ThreeVector & *a*, const ThreeVector & *b*) [static]

Cosine of angle between real vectors. $\cos\theta = \frac{\mathbf{a}\cdot\mathbf{b}}{|\mathbf{a}||\mathbf{b}|}$.

4.2.2.2 static double SpinDec::BoostEigen::maxAbsCoeff (const ThreeVector & *a*) [static]

Maximum absolute coefficient. Maximum of $[|a_1|, |a_2|, |a_3|]$ for real vector $\mathbf{a} = (a_1, a_2, a_3)$.

4.2.2.3 static ComplexMatrix SpinDec::BoostEigen::partialTrace (const ComplexMatrix & *AB*, const unsigned int *dimension_B*) [static]

Partial trace for complex matrices. Given $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$, and the dimension of $\mathbf{B}$, this method outputs $\mathrm{Tr}_{\mathbf{B}} \mathbf{A}$.

4.2.2.4 static ComplexMatrix SpinDec::BoostEigen::spectralDecomposition (const ComplexMatrix & *eigenvectors*, const ComplexVector & *eigenvalues*) [static]

Spectral decomposition of a complex matrix. This is $\mathbf{A} = \mathbf{V}\mathbf{D}\mathbf{V}^{-1}$, where $\mathbf{D}$ is the diagonal of eigenvalues of $\mathbf{A}$ and $\mathbf{V}$ is the columnwise eigenvector matrix.

4.2.2.5 static ComplexVector SpinDec::BoostEigen::tensorProduct (const ComplexVector & *a*, const ComplexVector & *b*) [static]

Tensor product for complex vectors. Evaluates $\mathbf{c} = \mathbf{a} \otimes \mathbf{b}$. For example, for 2-vectors, this is

$$\mathbf{c} = \begin{pmatrix} a_1 b_1 \\ a_1 b_2 \\ a_2 b_1 \\ a_2 b_2 \end{pmatrix}$$

4.2.2.6 static ComplexMatrix SpinDec::BoostEigen::tensorProduct (const ComplexMatrix & *A*, const ComplexMatrix & *B*) [static]

Tensor product for complex matrices. Evaluates $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$. For example, for 2×2 matrices, this is

$$\mathbf{C} = \begin{pmatrix} A_{11}\mathbf{B} & A_{12}\mathbf{B} \\ A_{21}\mathbf{B} & A_{22}\mathbf{B} \end{pmatrix}$$

4.2.2.7 static ComplexMatrix SpinDec::BoostEigen::unitarySpectralDecomposition (const ComplexMatrix & *eigenvectors*, const ComplexVector & *eigenvalues*) [static]

Spectral decomposition of a unitary matrix. For a unitary matrix, $\mathbf{A}^{-1} = \mathbf{V}^{\dagger}$

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

- /home/sbalian/spindec/include/SpinDec/BoostEigen.h

4.3 SpinDec::CCE Class Reference

Solves for a CSD problem using the CCE.

`#include <CCE.h>`

Public Member Functions

- **CCE** (const UInt max_truncation_order, const auto_ptr< PulseExperiment > &pulse_experiment, const ClusterDatabase &cluster_database, const bool include_one_clusters)
- UInt **get_max_truncation_order** () const
- void calculate (const UInt order)

  Calculate the CCE.

- void **calculate** (const UInt order, const bool no_divisions)
- void calculate ()

  Calls CCE::calculate(const UInt order,false).

- TimeEvolution evolution (const UInt order) const

  Get the time evolution (has to be calculated with above method first).

- const ClusterDatabase & **get_database** () const

Private Member Functions

- void **check_order** (const UInt order) const
- TimeEvolution **reducible_correlation** (const Cluster &cluster)
- TimeEvolution **true_correlation** (const Cluster &cluster)

Private Attributes

- vector< TimeEvolution > **product_correlations_by_order_**
- UInt max_truncation_order_

  Maximum CCE truncation order.

- bool **include_one_clusters_**
- auto_ptr< PulseExperiment > **pulse_experiment_**
- ClusterDatabase **cluster_database_**

4.3.1 Detailed Description

Solves for a CSD problem using the CCE. The cluster correlation expansion (CCE) is used to solve for a central spin decoherence (CSD) problem. Relevant references are:

- Phys. Rev. B 74, 035322 (2006),
- Phys. Rev. B 78, 085315 (2008),
- Phys. Rev. B 78, 129901(E) (2008),
- Phys. Rev. B 79, 115320 (2009),
- Phys. Rev. B 86, 035452 (2012).

4.3.2 Member Function Documentation

4.3.2.1 void SpinDec::CCE::calculate (const UInt *order*)

Calculate the CCE. Input CCE truncation order to calculate. This input cannot exceed CCE::max_truncation_order_.

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

- /home/sbalian/spindec/include/SpinDec/CCE.h

4.4 SpinDec::Cluster Class Reference

Contains labels for a cluster of spins.

`#include <Cluster.h>`

Public Member Functions

- **Cluster** (const UIntArray &labels)
- void **add** (const UInt label)
- bool **operator==** (const Cluster &rhs) const
- UInt num_spins () const

  Number of labels.

- UInt **get_label** (const UInt index) const
- const UIntArray & **get_labels** () const
- vector< Cluster > subsets () const

  Get all subsets (exludes the empty set).

- vector< Cluster > proper_subsets () const

  Get proper subsets (excludes the empty set).

Private Member Functions

- vector< UIntArray > **subsets** (const UIntArray &v, const UInt size) const
- vector< UIntArray > **subsets** (const UIntArray &v) const

Private Attributes

- UIntArray **labels_**

Friends

- std::ostream & operator<< (std::ostream &os, Cluster const &cluster)

  Print.

4.4.1 Detailed Description

Contains labels for a cluster of spins. Labels are always sorted in increasing order.

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

- /home/sbalian/spindec/include/SpinDec/Cluster.h

4.5 SpinDec::ClusterDatabase Class Reference

Stores clusters and associated complex time evolutions.

`#include <ClusterDatabase.h>`

Public Member Functions

- ClusterDatabase (const SpinBath &spin_bath, const UInt max_order, const double cluster_cutoff, const string &build_method)

  Build method is "global" or "local".

- const ClusterDatabaseEntry & **get_entry** (const UInt order, const UInt index) const
- const Cluster & **get_cluster** (const UInt order, const UInt index) const
- void **set_time_evolution** (const Cluster &cluster, const TimeEvolution &time_evolution)
- bool **is_solved** (const Cluster &cluster) const
- UInt **get_max_order** () const
- UInt **num_clusters** (const UInt order) const
- const TimeEvolution & **get_time_evolution** (const Cluster &cluster) const
- void **print** () const

Private Member Functions

- void build_pairs ()

  Build 2-clusters.

- void build_ones ()

  Build 1-clusters.

- void build_with_local_cutoff ()

  Higher order clusters with local cutoff.

- void build_with_global_cutoff ()

  Higher order clusters with global cutoff.

- UInt **get_index** (const Cluster &cluster) const
- void add_unsolved_entry (const Cluster &cluster)

  Add cluster if it does not exist.

- bool **is_order_built** (const UInt order) const
- bool **cluster_exists** (const Cluster &cluster) const

Private Attributes

- UInt max_order_

  Maximum build order.

- SpinBath **spin_bath_**
- database_map database_

Cluster size (order), vector of database entries. All unique clusters.

- double cluster_cutoff_

  In Å.

4.5.1 Detailed Description

Stores clusters and associated complex time evolutions. For use with SpinDec::CCE (cluster correlation expansion). Has two build methods, one with a local cluster cutoff where the maximum separation between any pair of spins in any cluster is cluster_cutoff. The other (global) method builds pairs with cluster_cutoff maximum separation, then adds spins which are at a maximum of cluster_cutoff from any of the two spins to form 3-clusters, and so on for 4-clusters etc.

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

- /home/sbalian/spindec/include/SpinDec/ClusterDatabase.h

4.6 SpinDec::ClusterDatabaseEntry Class Reference

Entry for ClusterDatabase.

`#include <ClusterDatabaseEntry.h>`

Public Member Functions

- ClusterDatabaseEntry (const Cluster &cluster)

  is_solved_ = false.

- const Cluster & **get_cluster** () const
- bool **is_solved** () const
- const TimeEvolution & **get_time_evolution** () const
- void **set_time_evolution** (const TimeEvolution &time_evolution)

Private Attributes

- Cluster **cluster_**
- TimeEvolution **time_evolution_**
- bool **is_solved_**

4.6.1 Detailed Description

Entry for ClusterDatabase.

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

- /home/sbalian/spindec/include/SpinDec/ClusterDatabaseEntry.h

4.7 SpinDec::Constants Class Reference

Mathematical and physical constants.

`#include <Constants.h>`

Static Public Attributes

- static const double kPi
  Pi.
- static const double kReducedPlanck
  Reduced Planck constant (J s).
- static const double kPlanck
  Plank constant (J s).
- static const double kElectronGyromagneticRatio
  Electronic gyromagnetic ratio (M rad s^{-1} T^{-1}).

4.7.1 Detailed Description

Mathematical and physical constants.

4.7.2 Member Data Documentation

4.7.2.1 const double SpinDec::Constants::kElectronGyromagneticRatio [static]

Electronic gyromagnetic ratio (M rad s^{-1} T^{-1}). From CODATA 22/10/2013.

4.7.2.2 const double SpinDec::Constants::kPi [static]

Pi. From Wikipedia 06/12/2012.

4.7.2.3 const double SpinDec::Constants::kPlanck [static]

Plank constant (J s). From CODATA 06/12/2012.

4.7.2.4 const double SpinDec::Constants::kReducedPlanck [static]

Reduced Planck constant (J s). From CODATA 06/12/2012.

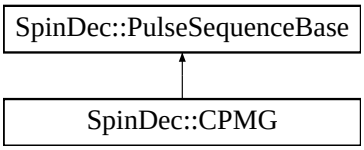
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/Constants.h

4.8 SpinDec::CPMG Class Reference

Carr-Purcell-Meiboom-Gill (CPMG) pulse sequence.

`#include <CPMG.h>`Inheritance diagram for SpinDec::CPMG::



Public Member Functions

- **CPMG** (const UInt order, const EvolutionOperator &evolution_operator, const Pulse &pi_pulse)
- void **set_time** (const double time_value)
- virtual auto_ptr< PulseSequenceBase > **clone** () const

Private Attributes

- UInt **order_**
- EvolutionOperator **evolution_operator_**
- vector< bool > **is_unitary_**

4.8.1 Detailed Description

Carr-Purcell-Meiboom-Gill (CPMG) pulse sequence.

- Order $N = 0$ (FID): evolve for duration t.
- $N = 1$ (Hahn): evolve for $t/2$, π-pulse, evolve for $t/2$.
- $N > 1$: [evolve for $t/(2N)$, π-pulse, evolve for $t/(2N)$]^N.

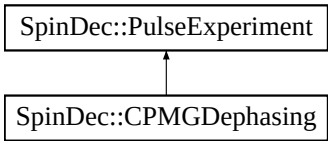
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/CPMG.h

4.9 SpinDec::CPMGDephasing Class Reference

CPMG pulse sequence.

`#include <CPMGDephasing.h>`Inheritance diagram for SpinDec::CPMGDephasing::



Public Member Functions

- **CPMGDephasing** (const CSDProblem &csd_problem, const TimeArray &time_array, const UInt cpmg_order, const CDouble &c0, const UInt level_label0, const CDouble &c1, const UInt level_label1)
- virtual TimeEvolution **time_evolution** (const UIntArray bath_indices)
- virtual auto_ptr< PulseExperiment > **clone** () const

Private Attributes

- UInt **cpmg_order_**
- TwoStateSuperposition **initial_system_state_**
- PiPulse **system_pi_pulse_**
- vector< pair< UInt, Pulse > > **pulses_**

4.9.1 Detailed Description

CPMG pulse sequence.

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

- /home/sbalian/spindec/include/SpinDec/CPMGDephasing.h

4.10 SpinDec::CrystalBasis Class Reference

Basis vectors for crystal structures in 3D.

`#include <CrystalBasis.h>`

Public Member Functions

- void **add_basis_vector** (const ThreeVector &basis_vector)
- const std::vector< ThreeVector > & **get_basis_vectors** () const
- const ThreeVector & **get_basis_vector** (const UInt index) const
- UInt **num_basis_vectors** () const

Protected Attributes

- std::vector< ThreeVector > **basis_vectors_**

4.10.1 Detailed Description

Basis vectors for crystal structures in 3D. Length units are Å. They are in the basis of lattice vectors $(\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3)$, NOT (x, y, z)!

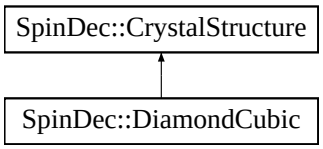
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/CrystalBasis.h

4.11 SpinDec::CrystalStructure Class Reference

$i\mathbf{a}_1 + j\mathbf{a}_2 + k\mathbf{a}_3 + \sum_n \mathbf{b}_n$.

`#include <CrystalStructure.h>`Inheritance diagram for SpinDec::CrystalStructure::



Public Member Functions

- CrystalStructure (const LatticeVectors &lattice_vectors, const CrystalBasis &basis, const int min_i, const int max_i, const int min_j, const int max_j, const int min_k, const int max_k, const double min_x, const double max_x, const double min_y, const double max_y, const double min_z, const double max_z, const double fractional_abundance)

  Calls fill_site_vectors.

- CrystalStructure (const string &file_name)

  Reads from file. Three columns: x, y, z.

- **CrystalStructure** (const vector< ThreeVector > &site_vectors)
- const std::vector< ThreeVector > & **get_site_vectors** () const
- const ThreeVector & **get_site_vector** (const UInt index) const
- UInt **num_site_vectors** () const
- double **max_site_vector_length** () const
- double max_abs_component () const

  $\max[\max(|x_1|, |y_1|, |z_1|), \ldots, \max(|x_n|, |y_n|, |z_n|)]$, *for* n *site vectors.*

- double **average_site_vector_separation** () const
- void **write_site_vectors** (const string &file_name) const

Protected Member Functions

- void fill_site_vectors (const LatticeVectors &lattice_vectors, const CrystalBasis &basis, const int min_i, const int max_i, const int min_j, const int max_j, const int min_k, const int max_k, const double min_x, const double max_x, const double min_y, const double max_y, const double min_z, const double max_z, const double fractional_abundance)

  Fills site vectors.

- void **add_site_vector** (const ThreeVector &site_vector)
- void **scale_site_vectors** (const double scale_factor)
- std::vector< ThreeVector > cartesian_basis_vectors (const LatticeVectors &lattice_vectors, const CrystalBasis &basis) const

  Get the basis vectors in Cartesian coordinates.

Protected Attributes

- std::vector< ThreeVector > **site_vectors_**

Private Member Functions

- void **read_site_vectors** (const string &file_name)

Friends

- std::ostream & operator<< (std::ostream &os, CrystalStructure const &crystal_structure)

  Print with cout (x, y, z).

4.11.1 Detailed Description

$i\mathbf{a}_1 + j\mathbf{a}_2 + k\mathbf{a}_3 + \sum_n \mathbf{b}_n$. The above = crystal structure, where $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ are linearly independent lattice vectors, the $\mathbf{b}_n$ are n basis vectors and i, j, k are integers.

The length units are Å. Using Cartesian coordinates.

4.11.2 Member Function Documentation

4.11.2.1 void SpinDec::CrystalStructure::fill_site_vectors (const LatticeVectors & *lattice_vectors*, const CrystalBasis & *basis*, const int *min_i*, const int *max_i*, const int *min_j*, const int *max_j*, const int *min_k*, const int *max_k*, const double *min_x*, const double *max_x*, const double *min_y*, const double *max_y*, const double *min_z*, const double *max_z*, const double *fractional_abundance*) `[protected]`

Fills site vectors. Here, the integer arguments are the i, j, k. The double arguments define the spatial ranges for all the x, y, z components for shaping the final crystal structure. The fractional abundance (converted to parts per million) is the fraction of site vectors added using a uniform distribution (using cstdlib rand(); see RandomNumberGenerator.h). If the fractional abundance is 1.0, then all site vectors are included.

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

- /home/sbalian/spindec/include/SpinDec/CrystalStructure.h

4.12 SpinDec::CSDProblem Class Reference

Central spin decoherence problem.

`#include <CSDProblem.h>`

Public Member Functions

- **CSDProblem** (const CSDProblem &csd_problem)
- CSDProblem & **operator=** (const CSDProblem &csd_problem)
- **CSDProblem** (const auto_ptr< SpinSystemBase > &central_spin_system_base, const SpinBath &spin_bath, const vector< SpinInteractionEdge > &system_bath_edges, const UniformMagneticField &field)
- **CSDProblem** (const auto_ptr< SpinSystemBase > &central_spin_system_base, const SpinBath &spin_bath, const SpinInteractionEdge &system_bath_edge, const UniformMagneticField &field)
- void **set_central_spin_state** (const SpinState &spin_state) const
- SpinSystem **get_reduced_problem** (const UIntArray bath_indices)
- const SpinBath & **get_spin_bath** () const
- auto_ptr< SpinSystemBase > **get_central_spin_system** () const

Private Member Functions

- void **init** (const auto_ptr< SpinSystemBase > &central_spin_system_base, const SpinBath &spin_bath, const vector< SpinInteractionEdge > &system_bath_edges, const UniformMagneticField &field)
- vector< SpinInteractionEdge > **make_system_bath_edges** (const UInt order, const SpinInteractionEdge &edge) const
- vector< SpinInteractionEdge > **make_system_bath_edges** (const UInt order) const
- SpinSystem **construct_reduced_problem** (const UInt order) const
- UIntArray **get_bath_vertex_labels** (const UInt order) const

Private Attributes

- auto_ptr< SpinSystemBase > **central_spin_system_**
- SpinBath **spin_bath_**
- UniformMagneticField **field_**
- vector< SpinInteractionEdge > system_bath_edges_
- vector< pair< UInt, SpinSystem > > **reduced_problems_**

4.12.1 Detailed Description

Central spin decoherence problem. Currently supports a single spin bath.

4.12.2 Member Data Documentation

4.12.2.1 vector<SpinInteractionEdge> SpinDec::CSDProblem::system_bath_edges_ [private]

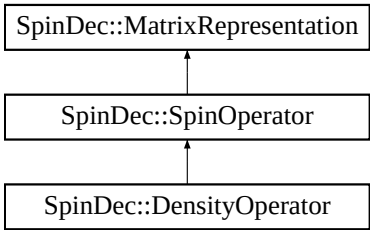
Edges should comply as in the join methods for SpinInteractionGraph, with the first graph being the central spin graph and the second being the graph for a single bath system.

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

- /home/sbalian/spindec/include/SpinDec/CSDProblem.h

4.13 SpinDec::DensityOperator Class Reference

`#include <DensityOperator.h>`Inheritance diagram for SpinDec::DensityOperator::



Public Member Functions

- DensityOperator (const SpinState &state, const SpinState &state0, const SpinState &state1)

  State is the combined (tensor product) qubit-other system state.

- DensityOperator reduced () const

  Return the qubit reduced density matrix.

- CDouble off_diagonal_reduced () const

  Off-diagonal of reduced density matrix.

Private Member Functions

- **DensityOperator** (const ComplexMatrix &matrix, const SpinBasis &basis, const SpinState &state0, const SpinState &state1)

Private Attributes

- SpinState **state0_**
- SpinState **state1_**

4.13.1 Detailed Description

In general, composite density operator for spins in the Zeeman basis. $\rho_{\mathrm{qubit}} \otimes \rho_{\mathrm{other}}$, qubit states $|0\rangle$ and $|1\rangle$.

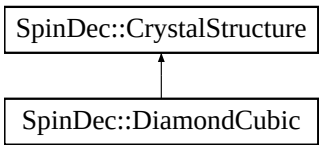
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/DensityOperator.h

4.14 SpinDec::DiamondCubic Class Reference

Diamond cubic crystal structure.

`#include <DiamondCubic.h>`Inheritance diagram for SpinDec::DiamondCubic::



Public Member Functions

- DiamondCubic (const double lattice_constant, const double side_length)

  Cubic lattice constant and side length of superlattice cube in Å.

- **DiamondCubic** (const double lattice_constant, const double side_length, const double fractional_-abundance)
- void **make_sphere** (const double radius)
- void **make_shell** (const double min_radius, const double max_radius)

Private Member Functions

- SimpleCubicLatticeVectors construct_lattice_vectors (const double lattice_constant) const

  Set up and return the lattice vectors.

- CrystalBasis construct_basis_vectors () const

  Set up and return the basis vectors.

- int int_range_centred_cube (const double side_length, const double lattice_constant) const

4.14.1 Detailed Description

Diamond cubic crystal structure. This is implemented as a simple cubic lattice with 8 basis vectors. Source: `http://en.wikipedia.org/wiki/Diamond_cubic.`

4.14.2 Member Function Documentation

4.14.2.1 int SpinDec::DiamondCubic::int_range_centred_cube (const double *side_length*, const double *lattice_constant*) const `[private]`

Get integer range (for constructing crystal structure) in a centred cube given the side length.

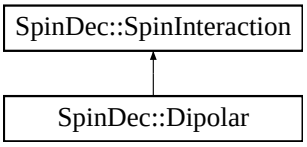
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/DiamondCubic.h

4.15 SpinDec::Dipolar Class Reference

Secular dipolar interaction strength between a pair of spins.

`#include <Dipolar.h>`Inheritance diagram for SpinDec::Dipolar::



Public Member Functions

- Dipolar ()

  If you wish to calculate.

- Dipolar (const double strength)

  If you don't wish to calculate.

- virtual void **calculate** (const SpinParameters &spin_parameters1, const SpinParameters &spin_parameters2, const ThreeVector &position1, const ThreeVector &position2, const UniformMagneticField &field)
- virtual void **fill** (ComplexMatrix *hamiltonian, const SpinParametersVector &spin_parameters_vector, const SpinBasis &basis, const UInt spin_label1, const UInt spin_label2) const
- virtual auto_ptr< SpinInteraction > **clone** () const
- virtual string **get_type** () const

4.15.1 Detailed Description

Secular dipolar interaction strength between a pair of spins. For two spins, $\hat{\mathbf{S}}_1, \hat{\mathbf{S}}_2$,

$$\hat{H}_D = D\hat{S}_1^z\hat{S}_2^z - \frac{D}{4}[\hat{S}_1^+\hat{S}_2^- + \hat{S}_1^-\hat{S}_2^+]$$

This class calculates

$$D = \frac{D_0\gamma_1\gamma_2(1-3\cos^2[\theta])}{(10^{-10}r)^3}$$

and fills the Hamiltonian matrix elements for the Hamiltonian $\hat{H}_D$ above. The energy units are M rad s^{-1}.

The parameters are:

- γ_1, γ_2 [M rad $s^{-1}T^{-1}$]: gyromagnetic ratios of the two interacting spins.
- θ [rad]: angle between the magnetic field direction and the vector parallel to the line connecting the two spins.
- r [Å]: distance between the two spins.
- $D_0 = 10^6(\mu_0/(4\pi))\hbar$ [M rad s^{-1} m^3].
- $\hbar$ [J s]: reduced Plank constant.

- $\mu_0/(4\pi) = 10^{-7}$ NA^{-2} (μ_0 is the vacuum permeability).

From: arXiv:cond-mat/0211567 (Phys. Rev. B 68, 115322 (2003)).

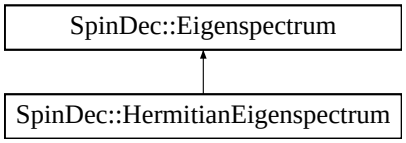
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/Dipolar.h

4.16 SpinDec::Eigenspectrum Class Reference

Holds the eigenvectors and eigenvalues of a matrix.

`#include <Eigenspectrum.h>`Inheritance diagram for SpinDec::Eigenspectrum::



Public Member Functions

- **Eigenspectrum** (const ComplexMatrix &matrix)
- const ComplexVector & **get_eigenvalues** () const
- const ComplexMatrix & **get_eigenvectors** () const
- CDouble **get_eigenvalue** (const UInt index) const
- ComplexVector **get_eigenvector** (const UInt index) const
- virtual ComplexMatrix spectralDecomposition () const

  Note the Eigen naming convention here.

- void set_spectrum (const ComplexMatrix &eigenvectors, const ComplexVector &eigenvalues)

  NOTE: use with care.

Protected Member Functions

- virtual void **diagonalize** (const ComplexMatrix &matrix)

Protected Attributes

- ComplexMatrix **eigenvectors_**
- ComplexVector **eigenvalues_**

4.16.1 Detailed Description

Holds the eigenvectors and eigenvalues of a matrix. Note: eigenvectors stored columnwise. General complex diagonalizer: ComplexEigenSolver in Eigen.

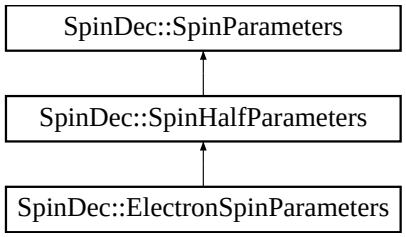
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/Eigenspectrum.h

4.17 SpinDec::ElectronSpinParameters Class Reference

Parameters for an electron spin.

`#include <ElectronSpinParameters.h>`Inheritance diagram for SpinDec::ElectronSpinParameters::



Public Member Functions

- ElectronSpinParameters ()

  Free electron gyromagnetic ratio.

- **ElectronSpinParameters** (const double gyromagnetic_ratio)

4.17.1 Detailed Description

Parameters for an electron spin. Gyromagnetic ratio of the free electron by default.

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

- /home/sbalian/spindec/include/SpinDec/ElectronSpinParameters.h

4.18 SpinDec::Errors Class Reference

Error and warning handling.

`#include <Errors.h>`

Static Public Member Functions

- static void **quit** ()
- static void **quit** (const string &message)
- static void **warning** (const string &message)

4.18.1 Detailed Description

Error and warning handling.

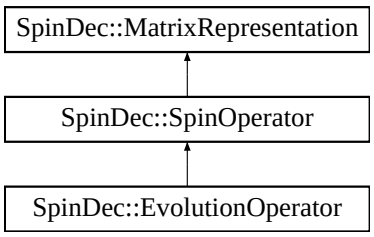
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/Errors.h

4.19 SpinDec::EvolutionOperator Class Reference

Free evolution operator.

`#include <EvolutionOperator.h>`Inheritance diagram for SpinDec::EvolutionOperator::



Public Member Functions

- EvolutionOperator (const SpinBasis &basis, const ComplexMatrix &eigenvectors, const RealVector &eigenvalues, const double time)
- void set_time (const double time)

  Updates matrix.

- double **get_time** () const

Private Member Functions

- void **set_matrix** ()

Private Attributes

- double **time_**
- ComplexMatrix eigenvectors_
- RealVector eigenvalues_

4.19.1 Detailed Description

Free evolution operator.

4.19.2 Constructor & Destructor Documentation

4.19.2.1 SpinDec::EvolutionOperator::EvolutionOperator (const SpinBasis & *basis*, const ComplexMatrix & *eigenvectors*, const RealVector & *eigenvalues*, const double *time*)

Unitary operator

$$\hat{U} = \sum_n \left(|E_n\rangle \exp[-iE_n t] \langle E_n| \right)$$

- E_n: eigenvalues (real).
- $|E_n\rangle$: eigenvectors.

- t: time (real double) in microseconds (energies in M rad s^{-1}).

For time independent Hamiltonians.

4.19.3 Member Data Documentation

4.19.3.1 RealVector SpinDec::EvolutionOperator::eigenvalues_ [private]

Not for the resulting opertor! These are used to construct the operator, for example from a Hamiltonian.

4.19.3.2 ComplexMatrix SpinDec::EvolutionOperator::eigenvectors_ [private]

Not for the resulting opertor! These are used to construct the operator, for example from a Hamiltonian.

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

- /home/sbalian/spindec/include/SpinDec/EvolutionOperator.h

4.20 SpinDec::FileProperties Class Reference

Static methods for ASCII file properties.

`#include <FileProperties.h>`

Static Public Member Functions

- static bool **exists** (const string file_name)
- static UInt num_lines (const string file_name)

  NOTE: empty lines not counted.

4.20.1 Detailed Description

Static methods for ASCII file properties.

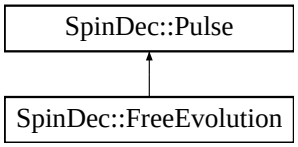
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/FileProperties.h

4.21 SpinDec::FreeEvolution Class Reference

(Pulse)-free evolution.

`#include <FreeEvolution.h>`Inheritance diagram for SpinDec::FreeEvolution::



Public Member Functions

- FreeEvolution (const EvolutionOperator &evolution_operator)

  "Pulse" duration taken from evolution operator.

4.21.1 Detailed Description

(Pulse)-free evolution.

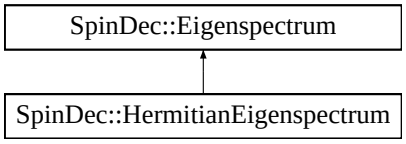
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/FreeEvolution.h

4.22 SpinDec::HermitianEigenspectrum Class Reference

Diagonalizes a Hermitian matrix.

`#include <HermitianEigenspectrum.h>`Inheritance diagram for SpinDec::HermitianEigenspectrum::



Public Member Functions

- **HermitianEigenspectrum** (const ComplexMatrix &matrix)
- virtual ComplexMatrix spectralDecomposition () const

Private Member Functions

- void diagonalize_eigen (const ComplexMatrix &matrix)

  Diagonalizer.

- virtual void **diagonalize** (const ComplexMatrix &matrix)

4.22.1 Detailed Description

Diagonalizes a Hermitian matrix. Eigenvectors are orthonormal, eigenvalues are always real.

4.22.2 Member Function Documentation

4.22.2.1 virtual ComplexMatrix SpinDec::HermitianEigenspectrum::spectralDecomposition () const [virtual]

Since eigenvectors orthonormal, $V^{-1} = V^{\dagger}$, use faster (unitary) decomposition.

Reimplemented from SpinDec::Eigenspectrum.

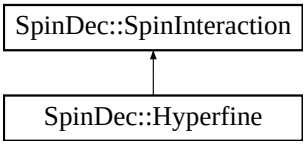
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/HermitianEigenspectrum.h

4.23 SpinDec::Hyperfine Class Reference

Calculates the electron-nuclear hyperfine interaction in a lattice.

`#include <Hyperfine.h>`Inheritance diagram for SpinDec::Hyperfine::



Public Member Functions

- Hyperfine (const HyperfineParameters &parameters)

  If you wish to calculate.

- Hyperfine (const double strength)

  If you don't wish to calculate.

- virtual void **calculate** (const SpinParameters &electron_parameters, const SpinParameters &nuclear_parameters, const ThreeVector &electron_position, const ThreeVector &nuclear_position, const UniformMagneticField &field)
- virtual void **fill** (ComplexMatrix *hamiltonian, const SpinParametersVector &spin_parameters_vector, const SpinBasis &basis, const UInt spin_label1, const UInt spin_label2) const
- virtual auto_ptr< SpinInteraction > **clone** () const
- virtual string **get_type** () const

Private Member Functions

- double **envelope_function** (const UInt index, const ThreeVector &separation) const
- double **n_parameter** () const
- double **n_times_a** () const
- double **n_times_b** () const
- double **scaled_probability_density** (const ThreeVector &separation) const

Private Attributes

- HyperfineParameters **parameters_**

4.23.1 Detailed Description

Calculates the electron-nuclear hyperfine interaction in a lattice. The Hamiltonian is

$$\hat{H}_A = A\hat{S}^z\hat{I}^z + \frac{A}{2}\left[\hat{S}^+\hat{I}^- + \hat{S}^-\hat{I}^+\right]$$

This class calculates A for the above Hamiltonian, $\hat{H}_A$, where $\hat{\mathbf{S}}$ and $\hat{\mathbf{I}}$ are the electronic and nuclear spin operators. Also fills the Hamiltonian matrix elements for $\hat{H}_A$. Uses the Kohn-Luttinger electronic wavefunction. Energy units are M rad s^{-1}.

$$A = pq - D(R)\theta(|R| - na)$$

The first term pq is the isotropic Fermi contact part.

$$p = \frac{16}{9}\pi\hbar\gamma_e\gamma_n\eta$$

where

- $\hbar$: reduced Planck constant [J s].
- γ_e: electron gyromagnetic ratio [M rad $s^{-1}T^{-1}$].
- γ_n: nuclear gyromagnetic ratio [M rad $s^{-1}T^{-1}$].
- η: charge density.

$$q = (10^{30})|F_1(R)\cos(k_0 x) + F_3(R)\cos(k_0 y) + F_5(R)\cos(k_0 z)|^2$$

where

- R: vector between nucleus and electron ((x, y, z)) components) [Å].
- $k_0 = 0.85 \times 2\pi a_0$ [$Å^{-1}$]
- $F_{1,2}(R) = \exp[-\sqrt{x^2/(nb)^2 + (y^2 + z^2)/(na)^2}]/\sqrt{\pi(na)^2 nb}$
- $F_{3,4}(R) : xyz \rightarrow yzx$.
- $F_{5,6}(R) : xyz \rightarrow zxy$.
- $n = \sqrt{0.029/E_i}$, E_i: electron ionization energy [eV].
- a and b are lattice parameters [Å].

The second term $[-D(R)\theta(|R| - na)]$ is the dipolar part which requires the direction of the magnetic field and where θ here is the Heaviside step function. See Dipolar.h for $D(R)$, the dipolar interaction (units M rad s^{-1}).

From: arXiv:cond-mat/0211567 (Phys. Rev. B 68, 115322 (2003)).

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

- /home/sbalian/spindec/include/SpinDec/Hyperfine.h

4.24 SpinDec::HyperfineParameters Class Reference

`#include <HyperfineParameters.h>`

Public Member Functions

- **HyperfineParameters** (const double lattice_constant, const double lattice_parameter_a, const double lattice_parameter_b, const double electron_ionization_energy, const double charge_density, const bool ising_only, bool fermi_contact_only_)
- double **get_lattice_constant** () const
- double **get_lattice_parameter_a** () const
- double **get_lattice_parameter_b** () const
- double **get_electron_ionization_energy** () const
- double **get_charge_density** () const
- bool **is_ising_only** () const
- bool **is_fermi_contact_only** () const

Private Attributes

- double lattice_constant_

  Lattice constant in Å.

- double lattice_parameter_a_

  Lattice parameter in Å.

- double lattice_parameter_b_

  Lattice parameter in Å.

- double electron_ionization_energy_

  Donor electron ionization energy in eV.

- double charge_density_

  Charge density (dimensionless).

- bool ising_only_

  Ignore flip-flop part of interaction.

- bool fermi_contact_only_

  ignore dipolar part of hyperfine interaction.

4.24.1 Detailed Description

Parameters to cacluate the hyperfine interaction between an electron spin and a nuclear spin in a lattice.

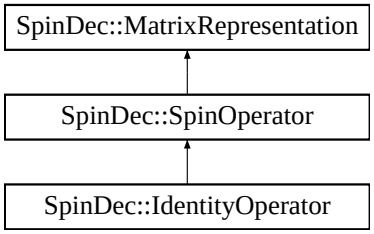
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/HyperfineParameters.h

4.25 SpinDec::IdentityOperator Class Reference

Identity.

`#include <IdentityOperator.h>`Inheritance diagram for SpinDec::IdentityOperator::



Public Member Functions

- **IdentityOperator** (const SpinBasis &basis)

4.25.1 Detailed Description

Identity.

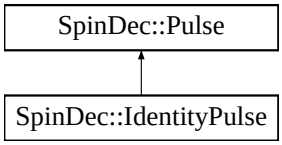
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/IdentityOperator.h

4.26 SpinDec::IdentityPulse Class Reference

Identity pulse (no duration).

`#include <IdentityPulse.h>`Inheritance diagram for SpinDec::IdentityPulse::



Public Member Functions

- **IdentityPulse** (const SpinBasis &basis)

4.26.1 Detailed Description

Identity pulse (no duration).

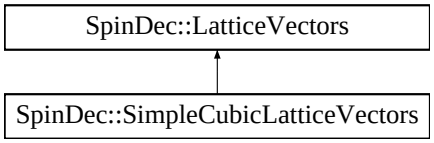
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/IdentityPulse.h

4.27 SpinDec::LatticeVectors Class Reference

Lattice vectors for 3D crystal structures.

`#include <LatticeVectors.h>`Inheritance diagram for SpinDec::LatticeVectors::



Public Member Functions

- **LatticeVectors** (const ThreeVector &a1, const ThreeVector &a2, const ThreeVector &a3)
- const ThreeVector & **get_a1** () const
- const ThreeVector & **get_a2** () const
- const ThreeVector & **get_a3** () const

Protected Member Functions

- bool **linearly_independent** () const
- void **set_lattice_vectors** (const ThreeVector &a1, const ThreeVector &a2, const ThreeVector &a3)

Protected Attributes

- ThreeVector **a1_**
- ThreeVector **a2_**
- ThreeVector **a3_**

4.27.1 Detailed Description

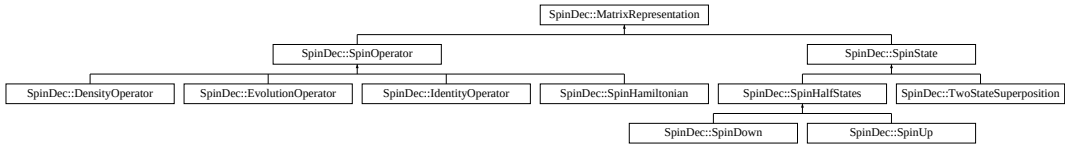
Lattice vectors for 3D crystal structures. Length units: Å. In Cartesian coordinates. Vectors must be linearly independent.

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

- /home/sbalian/spindec/include/SpinDec/LatticeVectors.h

4.28 SpinDec::MatrixRepresentation Class Reference

`#include <MatrixRepresentation.h>`Inheritance diagram for SpinDec::MatrixRepresentation::



Public Member Functions

- UInt **get_dimension** () const
- const SpinBasis & **get_basis** () const
- bool **is_basis_equal** (const auto_ptr< MatrixRepresentation > &to_check) const
- virtual void set_zero ()=0

  Set all elements to zero.

- virtual auto_ptr< MatrixRepresentation > **clone** () const =0

Protected Member Functions

- **MatrixRepresentation** (const SpinBasis &basis)
- virtual void **quit_if_dimension_mismatch** () const =0
- void **quit_if_basis_mismatch** (const auto_ptr< MatrixRepresentation > &to_check) const

Protected Attributes

- UInt dimension_

  Dimension of Hilbert space.

- SpinBasis **basis_**

4.28.1 Detailed Description

Abstract base class for spin states and operators in the matrix representation. Has a SpinBasis (Zeeman basis) and a dimension.

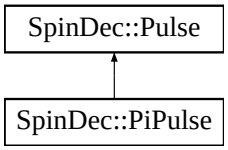
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/MatrixRepresentation.h

4.29 SpinDec::PiPulse Class Reference

π-pulse or refocusing pulse.

`#include <PiPulse.h>`Inheritance diagram for SpinDec::PiPulse::



Public Member Functions

- PiPulse (const SpinState &state0, const SpinState &state1)

  Instantaneous.

- PiPulse (const SpinState &state0, const SpinState &state1, const vector< SpinState > &states2_plus)

4.29.1 Detailed Description

π-pulse or refocusing pulse. $|0\rangle\langle 1| + |1\rangle\langle 0|$.

4.29.2 Constructor & Destructor Documentation

4.29.2.1 SpinDec::PiPulse::PiPulse (const SpinState & *state0*, const SpinState & *state1*, const vector< SpinState > & *states2_plus*)

Third parameter: these just add $|n\rangle\langle n|$ for $n = 2, 3, \ldots$

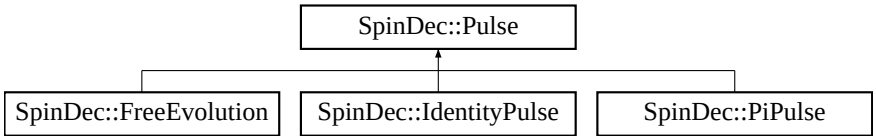
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/PiPulse.h

4.30 SpinDec::Pulse Class Reference

Pulses in a pulse sequence.

`#include <Pulse.h>`Inheritance diagram for SpinDec::Pulse::



Public Member Functions

- **Pulse** (const double duration, const SpinOperator &pulse_operator)
- const SpinOperator & **get_pulse_operator** () const
- double **get_duration** () const
- Pulse **operator*** (const Pulse &pulse) const
- Pulse **operator^** (const Pulse &pulse) const

Protected Attributes

- double **duration_**
- SpinOperator **pulse_operator_**

4.30.1 Detailed Description

Pulses in a pulse sequence. Includes the no-pulse free evolution.

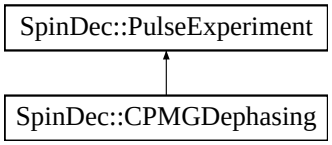
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/Pulse.h

4.31 SpinDec::PulseExperiment Class Reference

Abstract base class for a pulse sequence experiment.

`#include <PulseExperiment.h>`Inheritance diagram for SpinDec::PulseExperiment::



Public Member Functions

- **PulseExperiment** (const CSDProblem &csd_problem, const TimeArray &time_array)
- virtual TimeEvolution **time_evolution** (const UIntArray bath_indices)=0
- virtual auto_ptr< PulseExperiment > **clone** () const =0
- const TimeArray & **get_time_array** () const
- const CSDProblem & **get_csd_problem** () const

Protected Attributes

- CSDProblem **csd_problem_**
- TimeArray **time_array_**

4.31.1 Detailed Description

Abstract base class for a pulse sequence experiment.

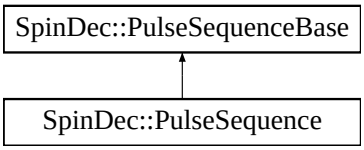
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/PulseExperiment.h

4.32 SpinDec::PulseSequence Class Reference

Concrete general pulse sequence.

`#include <PulseSequence.h>`Inheritance diagram for SpinDec::PulseSequence::



Public Member Functions

- void **clear** ()
- void **add_pulse** (const Pulse &pulse)
- virtual auto_ptr< PulseSequenceBase > **clone** () const

4.32.1 Detailed Description

Concrete general pulse sequence.

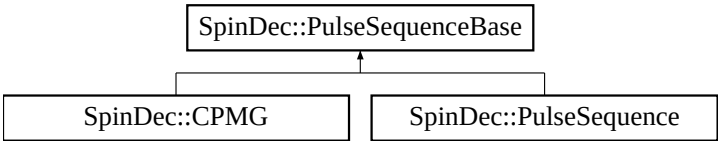
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/PulseSequence.h

4.33 SpinDec::PulseSequenceBase Class Reference

Abstract base class for pulse sequences.

`#include <PulseSequenceBase.h>`Inheritance diagram for SpinDec::PulseSequenceBase::



Public Member Functions

- SpinState **final_state** (const SpinState &initial_state) const
- double **get_duration** () const
- UInt **num_pulses** () const
- virtual auto_ptr< PulseSequenceBase > **clone** () const =0

Protected Attributes

- vector< Pulse > **pulses_**
- double **duration_**

4.33.1 Detailed Description

Abstract base class for pulse sequences.

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

- /home/sbalian/spindec/include/SpinDec/PulseSequenceBase.h

4.34 SpinDec::RandomNumberGenerator Class Reference

Static methods for generating random numbers.

`#include <RandomNumberGenerator.h>`

Static Public Member Functions

- static int uniform_c_rand (const int min, const int max)
- static void seed_uniform_c_rand (const int seed)

  If seed is negative, calls clock_seed_uniform_c_rand().

- static void clock_seed_uniform_c_rand ()

  Seed with current time.

- static double **normal_c_rand** (const double mean, const double stdev)

4.34.1 Detailed Description

Static methods for generating random numbers.

4.34.2 Member Function Documentation

4.34.2.1 static int SpinDec::RandomNumberGenerator::uniform_c_rand (const int *min*, const int *max*) [static]

Random integer $\min \leq i \leq \max$ from a uniform distribution using cstdlib rand().

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

- /home/sbalian/spindec/include/SpinDec/RandomNumberGenerator.h

4.35 SpinDec::ReducedProblem Class Reference

SpinSystemBase and an order.

`#include <ReducedProblem.h>`

Public Member Functions

- **ReducedProblem** (const UInt order, const auto_ptr< SpinSystemBase > &spin_system_base)
- **ReducedProblem** (const ReducedProblem &rhs)
- ReducedProblem & **operator=** (const ReducedProblem &rhs)
- UInt **get_order** () const
- auto_ptr< SpinSystemBase > **get_spin_system** () const

Private Attributes

- UInt **order_**
- auto_ptr< SpinSystemBase > **spin_system_base_**

4.35.1 Detailed Description

SpinSystemBase and an order.

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

- /home/sbalian/spindec/include/SpinDec/ReducedProblem.h

4.36 SpinDec::Sign Class Reference

±.

`#include <Sign.h>`

Public Member Functions

- int **as_int** () const
- bool **isPlus** () const
- bool **isMinus** () const
- bool **operator==** (const Sign &rhs) const

Static Public Attributes

- static const Sign **Plus**
- static const Sign **Minus**

Private Member Functions

- **Sign** (const int value)

Private Attributes

- int **value_**

Friends

- std::ostream & operator<< (std::ostream &os, Sign const &sign)

  Print with cout.

4.36.1 Detailed Description

±.

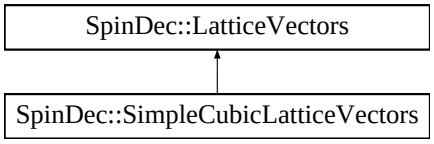
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/Sign.h

4.37 SpinDec::SimpleCubicLatticeVectors Class Reference

Simple cubic lattice vectors.

`#include <SimpleCubicLatticeVectors.h>`Inheritance diagram for SpinDec::SimpleCubicLatticeVectors::



Public Member Functions

- SimpleCubicLatticeVectors (const double lattice_constant)

  Lattice constant a_0 in Å.

4.37.1 Detailed Description

Simple cubic lattice vectors. $|\mathbf{a}_1| = |\mathbf{a}_2| = |\mathbf{a}_3| = a_0$ and $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ mutually orthogonal (parallel to x, y, z respectively in Cartesian coordinates).

Source: http://en.wikipedia.org/wiki/Cubic_crystal_system.

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

- /home/sbalian/spindec/include/SpinDec/SimpleCubicLatticeVectors.h

4.38 SpinDec::SpinBasis Class Reference

Holds magnetic quantum numbers for (in general) multiple spins.

`#include <SpinBasis.h>`

Public Member Functions

- SpinBasis (const SpinParametersVector &spin_parameters_vector)

  Automatically build using spin multiplicities.

- SpinBasis (const SpinParameters &spin_parameters)

  Automatically build using spin multiplicity.

- SpinBasis (const Eigen::ArrayXXd &basis_as_array)

  Custom build.

- const Eigen::ArrayXXd & **get_basis_as_array** () const
- UInt **num_basis_states** () const
- UInt **num_spins** () const
- double **get_element** (const UInt index, const UInt slot) const
- SpinBasis operator+ (const SpinBasis &to_append) const
- SpinBasis operator^ (const SpinBasis &to_combine) const
- bool operator== (const SpinBasis to_compare) const

  Check if bases are identical.

- bool **is_equal** (const SpinBasis &basis) const

Private Member Functions

- Eigen::ArrayXXd build (const SpinParametersVector &spin_parameters_vector)

  Automatically build using spin multiplicities.

- Eigen::ArrayXXd build (const SpinParameters &spin_parameters)

  Build using multiplicity.

Private Attributes

- Eigen::ArrayXXd **basis_as_array_**

Friends

- std::ostream & operator<< (std::ostream &os, SpinBasis const &basis)

  Print with cout.

4.38.1 Detailed Description

Holds magnetic quantum numbers for (in general) multiple spins. Implements the Zeeman basis and includes build methods.

Columns: spins. Rows: magnetic quantum numbers.

For example, for two electrons, this is

$$\begin{matrix} 0.5 & 0.5 \\ 0.5 & -0.5 \\ -0.5 & 0.5 \\ -0.5 & 0.5 \end{matrix}$$

For example, the first row corresponds to $|m_{S1} = 0.5, m_{S2} = 0.5\rangle$.

4.38.2 Member Function Documentation

4.38.2.1 SpinBasis SpinDec::SpinBasis::operator+ (const SpinBasis & *to_append*) const

For example,

$$\text{basis1} = \begin{matrix} 0.5 \\ -0.5 \end{matrix}$$

and

$$\text{basis2} = \begin{matrix} 4.5 \\ -4.5 \end{matrix}$$

basis1 + basis2 gives

$$\begin{matrix} 0.5 & 4.5 \\ -0.5 & -4.5 \end{matrix}$$

4.38.2.2 SpinBasis SpinDec::SpinBasis::operator^ (const SpinBasis & *to_combine*) const

Like tensor product. For example,

$$\text{basis1} = \begin{matrix} 4.5 \\ -4.5 \end{matrix}$$

and

$$\text{basis2} = \begin{matrix} 0.5 \\ -0.5 \end{matrix}$$

basis1^(basis2) is

$$\begin{matrix} 4.5 & 0.5 \\ 4.5 & -0.5 \\ -4.5 & 0.5 \\ -4.5 & -0.5 \end{matrix}$$

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

- /home/sbalian/spindec/include/SpinDec/SpinBasis.h

4.39 SpinDec::SpinBath Class Reference

Spin bath (single spin species).

`#include <SpinBath.h>`

Public Member Functions

- **SpinBath** (const CrystalStructure &crystal_structure, const auto_ptr< SpinSystemBase > &spin_system_base, const vector< SpinInteractionEdge > &intrabath_edges)
- **SpinBath** (const CrystalStructure &crystal_structure, const auto_ptr< SpinSystemBase > &spin_system_base, const SpinInteractionEdge &intrabath_edge)
- **SpinBath** (const SpinBath &spin_bath)
- SpinBath & **operator=** (const SpinBath &spin_bath)
- const SpinState & get_bath_state (const UInt index) const

  Gets the state of the spin system.

- UInt **num_bath_states** () const
- SpinState get_bath_product_state (const UIntArray &indices) const

  Tensor product state for multiple populated sites.

- const CrystalStructure & **get_crystal_structure** () const
- const vector< SpinInteractionEdge > & **get_intrabath_edges** () const
- auto_ptr< SpinSystemBase > **get_spin_system** () const
- SpinInteractionGraph **reduced_problem_graph** (const UInt order) const
- ThreeVector **get_position** (const UInt vertex_label, const UInt bath_index) const
- void **set_bath_state** (const UInt index, const UInt level)

Private Member Functions

- void **init** (const CrystalStructure &crystal_structure, const auto_ptr< SpinSystemBase > &spin_system_base, const vector< SpinInteractionEdge > &intrabath_edges)
- vector< SpinInteractionEdge > **make_intrabath_edges** (const UInt order, const SpinInteractionEdge &intrabath_edge) const
- vector< SpinInteractionEdge > **make_intrabath_edges** (const UInt order) const

Private Attributes

- vector< SpinState > **bath_states_**
- CrystalStructure **crystal_structure_**
- auto_ptr< SpinSystemBase > spin_system_base_
- vector< SpinInteractionEdge > intrabath_edges_

4.39.1 Detailed Description

Spin bath (single spin species). For the central spin decoherence problem in a crystal. Infinite temperature, so that all states are equally likely.

4.39.2 Member Function Documentation

4.39.2.1 const SpinState& SpinDec::SpinBath::get_bath_state (const UInt *index*) const

Gets the state of the spin system. Index is for populated sites in the crystal structure. Infinite temperature ensemble ...

4.39.3 Member Data Documentation

4.39.3.1 vector<SpinInteractionEdge> SpinDec::SpinBath::intrabath_edges_ [private]

Take the graph of the spin system and consider joining the graph to itself. For example, if the vertices of the original graph were 0, 1, 2, the new graph will have vertices 0, 1, 2 (originals), 3, 4, 5 (copies). Use these new labels for the intrabath edges. This is like the joining methods in SpinInteractionGraph.

4.39.3.2 auto_ptr<SpinSystemBase> SpinDec::SpinBath::spin_system_base_ [private]

This spin system is placed at every occupied site in the crystal structure. The positions of each of the spins in the spin systems are added to the site vector.

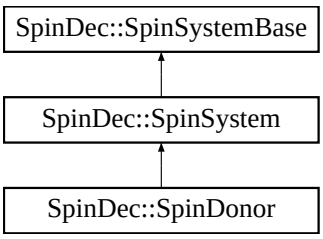
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinBath.h

4.40 SpinDec::SpinDonor Class Reference

Special spin system: mixed electron-nuclear spin donors.

`#include <SpinDonor.h>`Inheritance diagram for SpinDec::SpinDonor::



Public Member Functions

- SpinDonor (const double field_strength, const double nuclear_quantum_number, const double electron_gyromagnetic_ratio, const double nuclear_gyromagnetic_ratio, const double hyperfine_-strength, const unsigned int lower_level_label, const unsigned int upper_level_label, const Three-Vector &electron_position, const ThreeVector &nuclear_position, const bool complete_basis)
- const ElectronSpinParameters & **get_electron_parameters** () const
- const SpinParameters & **get_nuclear_parameters** () const
- const Hyperfine & **get_hyperfine** () const
- int max_quantum_number () const

  $I + S$, I is nuclear spin quantum number.

- virtual UInt dimension () const

  This is the Hamiltonian dimension.

- UInt total_multiplicity () const

  The complete spin basis dimension. This is $(2S + 1)(2I + 1)$.

- virtual SpinState eigenstate (const UInt level_label)

  Levels 0,1,2, ... dimension(Hamiltonian)-1 (some levels may be excluded!).

- virtual double energy (const UInt level_label)

  Energy eigenvalue in M rad s^{-1}.

- double polarization (const UInt level_label) const
- const SpinInteractionVertex & **electron_vertex** () const
- const SpinInteractionVertex & **nuclear_vertex** () const
- const UIntArray **get_orthogonal_level_labels** () const
- SpinState **get_lower_level** ()
- SpinState **get_upper_level** ()
- vector< SpinState > **get_orthogonal_levels** ()
- virtual PiPulse pi_pulse (const UInt level_label1, const UInt level_label2)

  π-pulse (refocusing pulse).

- virtual auto_ptr< SpinSystemBase > **clone** () const

Private Member Functions

- void sort_level_labels ()

  Sets and sorts sorted_level_labels_ in increasing energy.

- UInt level_label_index (const UInt level_label) const

  Given level label, returns the index in sorted_level_labels_.

- virtual void **check_level_label** (const UInt level_label) const
- void calc_adiabatic_level_labels ()
- double delta () const

  Ratio of gyromagnetic ratios $\delta = -\gamma_n/\gamma_e$.

- double omega () const

  $\omega = \gamma_e B$ *[M rad* s^{-1}*], where* B *is the Zeeman field strength.*

- double scaled_omega () const

  $\omega' = \omega/A$*, where* A *is the electron-nuclear hyperfine strength in M rad* s^{-1}.

- double D (const int quantum_number) const

  $D = m + \omega'(1 + \delta)$.

- double O (const int quantum_number) const

  $O = \sqrt{I(I+1) + 1/4 - m^2}$.

- double R (const int quantum_number) const

  $R = \sqrt{D^2 + O^2}$.

- double energy (const AdiabaticLabel &adiabatic_level_label) const
- UInt adiabatic_label_to_int_label (const AdiabaticLabel &adiabatic_level_label) const

  Convert $|\pm, m\rangle$ *levels to* $0, 1, 2, \ldots,$ *dim(donor)* $- 1$.

- AdiabaticLabel int_label_to_adiabatic_label (const UInt level_label) const

  Other way to adiabatic_label_to_int_label.

- AdiabaticLabel orthogonal_adiabatic_level_label (AdiabaticLabel adiabatic_level_label) const

  Given $|\pm, m\rangle$*, returns* $|\mp, m\rangle$.

- double cos_theta (const int quantum_number) const

  $\cos\theta = D/R$.

- double sin_theta (const int quantum_number) const

  $\sin\theta = O/R$.

- double a (const int quantum_number) const

  $[1/\sqrt{2}]\sqrt{1 + \cos\theta}$.

- double b (const int quantum_number) const

  $[1/\sqrt{2}]\sqrt{1 - \cos\theta}$.

- double polarization (const AdiabaticLabel &adiabatic_level_label) const

  Expectation value of the electron z-component of spin.

- SpinBasis build_basis (const AdiabaticLabel &adiabatic_level_label) const
- SpinBasis build_basis (const std::vector< AdiabaticLabel > &adiabatic_level_labels) const
- SpinBasis build_basis (const UIntArray &level_labels) const
- SpinBasis build_truncated_basis () const
- void set_transition (const UInt lower_level_label, const UInt upper_level_label)
- void **set_orthogonal_level_labels** (const UInt lower_level_label, const UInt upper_level_label)
- UIntArray **get_orthogonal_level_labels** (const UInt lower_level_label, const UInt upper_level_label) const
- UInt orthogonal_level_label (const UInt level_label) const

  Given $|\pm, m\rangle$, returns $|\mp, m\rangle$.

- void init (const double field_strength, const double nuclear_quantum_number, const double electron_gyromagnetic_ratio, const double nuclear_gyromagnetic_ratio, const double hyperfine_strength, const unsigned int lower_level_label, const unsigned int upper_level_label, const ThreeVector &electron_position, const ThreeVector &nuclear_position, const bool complete_basis)

  See constructors.

Private Attributes

- bool complete_basis_

  Use full Zeeman basis.

- ElectronSpinParameters electron_parameters_

  For the spin interaction graph.

- SpinParameters nuclear_parameters_

  For the spin interaction graph.

- Hyperfine hyperfine_

  For the spin interaction graph.

- UInt transition_level_labels_ [2]

  Upper and lower transition energy levels.

- UIntArray orthogonal_level_labels_
- UIntArray sorted_level_labels_
- std::vector< AdiabaticLabel > adiabatic_level_labels_

4.40.1 Detailed Description

Special spin system: mixed electron-nuclear spin donors. Electron coupled to a nucleus via a preset hyperfine interaction. Analytical methods include energy levels, eigenstates and polarization obtained from Phys. Rev. Lett. 105, 067602 (2010).

Energy levels are (adiabatic states) labeled as follows, with adiabatic level labels:

$$|\pm, m\rangle\,,$$

with integer quantum number $m = -|I + S|, \ldots, I + S$, separated by 1. S and I are the electron and nuclear spin quantum numbers respectively.

Alternatively, energy levels are labeled according to increasing energy: $0, 1, \ldots, (2S + 1)(2I + 1) - 1$.

4.40.2 Constructor & Destructor Documentation

4.40.2.1 SpinDec::SpinDonor::SpinDonor (const double *field_strength*, const double *nuclear_quantum_number*, const double *electron_gyromagnetic_ratio*, const double *nuclear_gyromagnetic_ratio*, const double *hyperfine_strength*, const unsigned int *lower_level_label*, const unsigned int *upper_level_label*, const ThreeVector & *electron_position*, const ThreeVector & *nuclear_position*, const bool *complete_basis*)

Field strength in Tesla (T), gyromagnetic ratios in M rad $s^{-1}T^{-1}$, hyperfine strength in M rad s^{-1}, and positions in Å.

4.40.3 Member Function Documentation

4.40.3.1 SpinBasis SpinDec::SpinDonor::build_basis (const UIntArray & *level_labels*) const [private]

Same as build_basis(const std::vector<AdiabaticLabel>&) but with integer labels for energy levels as the arguments.

4.40.3.2 SpinBasis SpinDec::SpinDonor::build_basis (const std::vector< AdiabaticLabel > & *adiabatic_level_labels*) const [private]

Like build_basis(const AdiabaticLabel&), but for multiple levels (non-zero coefficient basis states).

4.40.3.3 SpinBasis SpinDec::SpinDonor::build_basis (const AdiabaticLabel & *adiabatic_level_label*) const [private]

Gives the (at most two) Zeeman basis states: $|m_S = \pm 1/2, m_I = m \mp 1/2\rangle$ and $|m_S = \mp 1/2, m_I = m \pm 1/2\rangle$ where $m_S = \pm 1/2$ and $m_I = -I, \ldots, +I$ in integer steps, for the energy state:

$$|\pm, m\rangle = a_m \, |m_S = \pm 1/2, m_I = m \mp 1/2\rangle \pm b_m \, |m_S = \mp 1/2, m_I = m \pm 1/2\rangle \, .$$

- For $m = -M$, $|-, -M\rangle = |-1/2, -I\rangle$ (only one Zeeman state) ($M = \max(m)$).
- For $m = +M$, $|+, +M\rangle = |+1/2, +I\rangle$ (only one Zeeman state).

States $|+, -M\rangle$ and $|-, M\rangle$ don't exist. For all the other $|\pm, m\rangle$, there are two basis states.

4.40.3.4 SpinBasis SpinDec::SpinDonor::build_truncated_basis () const [private]

This calls build_basis(const UIntArray&) const for UIntArray containing the upper and lower levels only. Output basis has the Zeeman basis states in which the upper and lower levels as well as the levels orthogonal to these (if they exist) can be completely represented. This is a complete basis for interactions of the donor via the z-component of the electron spin only.

4.40.3.5 void SpinDec::SpinDonor::calc_adiabatic_level_labels () [private]

Calculates and sets all the adiabatic level labels and adiabatic_level_labels_.

4.40.3.6 double SpinDec::SpinDonor::energy (const AdiabaticLabel & *adiabatic_level_label*) const [private]

$$E = \frac{A}{2}\left(-\frac{1}{2}\left[1 + 4\omega' m\delta\right] \pm R(m)\right)$$

for level $|\pm, m\rangle$ and A is the hyperfine strength. Units are M rad s^{-1}.

4.40.3.7 AdiabaticLabel SpinDec::SpinDonor::orthogonal_adiabatic_level_label (AdiabaticLabel *adiabatic_level_label*) const [private]

Given $|\pm, m\rangle$, returns $|\mp, m\rangle$. If $|\mp, m\rangle$ does not exist ($m = |M|$), returns the input.

4.40.3.8 UInt SpinDec::SpinDonor::orthogonal_level_label (const UInt *level_label*) const [private]

Given $|\pm, m\rangle$, returns $|\mp, m\rangle$. If $|\mp, m\rangle$ does not exist ($m = |M|$), returns the input.

This method takes and outputs integer energy level inputs $0, 1, \ldots,$ donor(dim) $- 1$.

4.40.3.9 double SpinDec::SpinDonor::polarization (const UInt *level_label*) const

See polarization(const AdiabaticLabel& label) const. This takes an integer energy level label. Gives expectation value of the electron z-component of spin (for field along z) for input level.

4.40.3.10 double SpinDec::SpinDonor::polarization (const AdiabaticLabel & *adiabatic_level_label*) const [private]

Expectation value of the electron z-component of spin. For field along z, for level $\pm, m$.

This is

$$\langle \pm, m | \hat{S}^z | \pm, m \rangle = \pm\frac{1}{2}\left[a^2(m) - b^2(m)\right].$$

4.40.3.11 void SpinDec::SpinDonor::set_transition (const UInt *lower_level_label*, const UInt *upper_level_label*) [private]

Also sets orthogonal levels if they exist.

4.40.4 Member Data Documentation

4.40.4.1 std::vector<AdiabaticLabel> SpinDec::SpinDonor::adiabatic_level_labels_ [private]

In order of increasing energy, these are:

$$|-, M-1\rangle, |-, M-2\rangle, \ldots, |-, -M\rangle, |-, -M+1\rangle, \ldots |+, +M\rangle,$$

where $M = I + S$.

4.40.4.2 bool SpinDec::SpinDonor::complete_basis_ [private]

Use full Zeeman basis. Computationally intensive for CCE - 20D for Si:Bi.). Otherwise, the truncated basis is used, which consists of the subspaces for the two levels involved in the transition.

4.40.4.3 UIntArray SpinDec::SpinDonor::orthogonal_level_labels_ [private]

If the donor interacts via the electron z-component spin operator (with the field along z), Hamiltonian matrix elements involving the orthogonal levels must be included in the Hilbert space for completeness. Orthogonal levels are $|\pm, m\rangle \leftrightarrow |\mp, m\rangle$ (if they exist). These orthogonal levels are included so that all the relevant Zeeman basis states are included. Elements are:

- Orthogonal lower energy level.
- Orthogonal upper energy level.

At most of size 2.

4.40.4.4 UIntArray SpinDec::SpinDonor::sorted_level_labels_ [private]

These are the upper, lower and orthogonal levels for a non-complete basis and all the energy levels for the complete basis, sorted in order of increasing energy.

4.40.4.5 UInt SpinDec::SpinDonor::transition_level_labels_[2] [private]

Upper and lower transition energy levels. Labeled $0, 1, 2, \ldots, (2S+1)(2I+1) - 1$. Elements are:

- Lower energy level label for a transition.
- Upper energy level label for a transition.

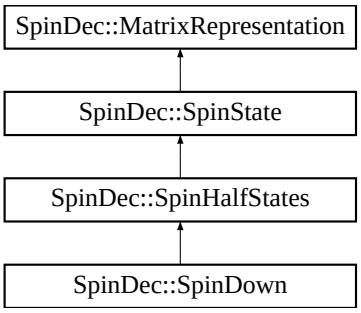
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinDonor.h

4.41 SpinDec::SpinDown Class Reference

Spin down state.

`#include <SpinDown.h>`Inheritance diagram for SpinDec::SpinDown::



Public Member Functions

- **SpinDown** (const SpinHalfParameters &spin_half_parameters)

Private Member Functions

- virtual void init (const double gyromagnetic_ratio)

4.41.1 Detailed Description

Spin down state. For a positive (or zero) gyromagnetic ratio (and positive spin quantum number) this is (0,1) in the (0.5,-0.5) basis. See also SpinUp.

4.41.2 Member Function Documentation

4.41.2.1 virtual void SpinDec::SpinDown::init (const double *gyromagnetic_ratio*) [private, virtual]

Depending on sign of gyromagnetic_ratio, spin up or spin down. See derived classes SpinUp and SpinDown.

Implements SpinDec::SpinHalfStates.

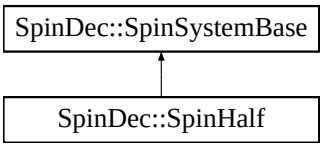
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinDown.h

4.42 SpinDec::SpinHalf Class Reference

A spin-1/2 spin system.

`#include <SpinHalf.h>`Inheritance diagram for SpinDec::SpinHalf::



Public Member Functions

- **SpinHalf** (const double gyromagnetic_ratio, const double field_strength, const ThreeVector &position)
- virtual UInt dimension () const

  Hamiltonian dimension.

- virtual auto_ptr< SpinSystemBase > **clone** () const

Private Member Functions

- virtual void solve_once ()

  Calculate and set energies and eigenstates.

- virtual void **check_level_label** (const UInt level_label) const

Private Attributes

- double **gyromagnetic_ratio_**

4.42.1 Detailed Description

A spin-1/2 spin system.

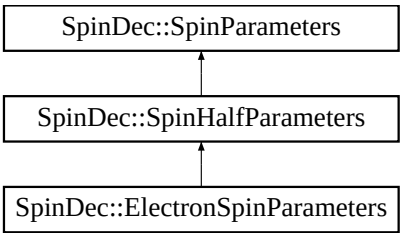
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinHalf.h

4.43 SpinDec::SpinHalfParameters Class Reference

Spin-1/2 spin parameters.

`#include <SpinHalfParameters.h>`Inheritance diagram for SpinDec::SpinHalfParameters::



Public Member Functions

- **SpinHalfParameters** (const double gyromagnetic_ratio)

4.43.1 Detailed Description

Spin-1/2 spin parameters.

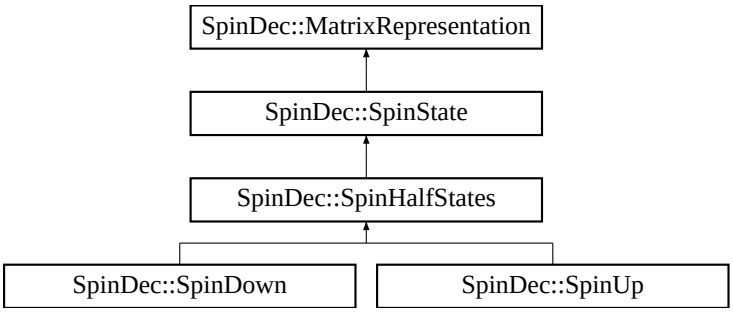
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinHalfParameters.h

4.44 SpinDec::SpinHalfStates Class Reference

Spin-1/2 spin states.

`#include <SpinHalfStates.h>`Inheritance diagram for SpinDec::SpinHalfStates::



Protected Member Functions

- SpinHalfStates ()

  Basis is (0.5,-0.5).

- virtual void init (const double gyromagnetic_ratio)=0
- virtual void **set_state_vector** (const ComplexVector &state_vector)
- virtual void **set_element** (const UInt index, const CDouble &element)
- virtual void set_element (const UInt index, const double element)

  Stored as complex.

4.44.1 Detailed Description

Spin-1/2 spin states.

4.44.2 Member Function Documentation

4.44.2.1 virtual void SpinDec::SpinHalfStates::init (const double *gyromagnetic_ratio*) [protected, pure virtual]

Depending on sign of gyromagnetic_ratio, spin up or spin down. See derived classes SpinUp and SpinDown.

Implemented in SpinDec::SpinDown, and SpinDec::SpinUp.

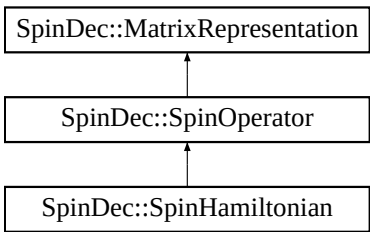
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinHalfStates.h

4.45 SpinDec::SpinHamiltonian Class Reference

Effective spin Hamiltonian built from a spin interaction graph.

`#include <SpinHamiltonian.h>`Inheritance diagram for SpinDec::SpinHamiltonian::



Public Member Functions

- **SpinHamiltonian** (const SpinInteractionGraph &graph, const UniformMagneticField &field)
- UniformMagneticField **get_field** () const
- const SpinInteractionGraph & **get_graph** () const
- void **update_positions** (const UIntArray &vertex_labels, const vector< ThreeVector > &positions)

Private Member Functions

- void fill_zeeman ()

  Fill diagonal elements with γBm.

- void fill_interactions ()

  Fill elements for all spin interactions.

- void init_terms ()

  Sets to zero.

- void **sum_zeeman_terms** ()
- void **sum_interaction_terms** ()
- void **fill_zeeman** (const UInt vertex_label)
- void **fill_interaction** (const UInt edge_index)

Private Attributes

- UniformMagneticField **field_**
- SpinInteractionGraph **graph_**
- vector< ComplexMatrix > zeeman_terms_

  For each vertex, there is a Zeeman Hamiltonian. Store these.

- vector< ComplexMatrix > interaction_terms_

  For each edge, there is an interaction Hamiltonian. Store these.

- ComplexMatrix zeeman_hamiltonian_

Summed Zeeman Hamiltonian.

- ComplexMatrix interaction_hamiltonian_

  Summed interaction Hamiltonian.

4.45.1 Detailed Description

Effective spin Hamiltonian built from a spin interaction graph. Can diagonalize and get unitary time evolution matrix. No time dependence in Hamiltonian. Units: M rad s^{-1}.

4.45.2 Member Function Documentation

4.45.2.1 void SpinDec::SpinHamiltonian::fill_zeeman () [private]

Fill diagonal elements with γBm. B-field strength is for all spins in the graph.

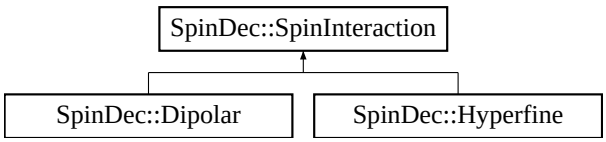
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinHamiltonian.h

4.46 SpinDec::SpinInteraction Class Reference

Abstract base class for interaction between a pair of spins.

`#include <SpinInteraction.h>`Inheritance diagram for SpinDec::SpinInteraction::



Public Member Functions

- virtual void **calculate** (const SpinParameters &spin_parameters1, const SpinParameters &spin_parameters2, const ThreeVector &position1, const ThreeVector &position2, const UniformMagneticField &field)=0
- double **get_strength** () const
- bool **strength_preset** () const
- virtual void **fill** (ComplexMatrix *hamiltonian, const SpinParametersVector &spins, const SpinBasis &basis, const UInt spin_label1, const UInt spin_label2) const =0
- virtual auto_ptr< SpinInteraction > **clone** () const =0
- virtual string **get_type** () const =0

Protected Member Functions

- SpinInteraction ()

  If you wish to calculate.

- SpinInteraction (const double strength)

  If you don't wish to calculate.

- void fill_ising_flipflop (ComplexMatrix *hamiltonian, const SpinParametersVector &spin_parameters_vector, const SpinBasis &basis, const UInt spin_label1, const UInt spin_label2, const bool ising_only, const CDouble &flipflop_form) const
- void **warn_if_preset_then_calculated** () const

Protected Attributes

- double strength_

  M rad s^{-1} (calculated or set).

- bool strength_preset_

  Set to true, this indicates that the strength was set in the constructor.

4.46.1 Detailed Description

Abstract base class for interaction between a pair of spins.

4.46.2 Member Function Documentation

4.46.2.1 void SpinDec::SpinInteraction::fill_ising_flipflop (ComplexMatrix * *hamiltonian*, const SpinParametersVector & *spin_parameters_vector*, const SpinBasis & *basis*, const UInt *spin_label1*, const UInt *spin_label2*, const bool *ising_only*, const CDouble & *flipflop_form*) const [protected]

Fill

$$J\hat{S}_1^z\hat{S}_2^z + FJ(\hat{S}_1^+\hat{S}_2^- + \hat{S}_1^-\hat{S}_2^+),$$

where S is the strength, F is a factor depending on the type of interaction, the first term is known as "Ising" and the second term as "Flip-Flop".

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

- /home/sbalian/spindec/include/SpinDec/SpinInteraction.h

4.47 SpinDec::SpinInteractionEdge Class Reference

Edge for a spin interaction graph.

`#include <SpinInteractionEdge.h>`

Public Member Functions

- **SpinInteractionEdge** (const SpinInteractionEdge &other)
- **SpinInteractionEdge** (const UInt label1, const UInt label2, const auto_ptr< SpinInteraction > &interaction)
- UInt **get_label1** () const
- UInt **get_label2** () const
- auto_ptr< SpinInteraction > **get_interaction** () const
- SpinInteractionEdge & **operator=** (const SpinInteractionEdge &other)

Private Attributes

- pair< UInt, UInt > **labels_**
- auto_ptr< SpinInteraction > **interaction_**

4.47.1 Detailed Description

Edge for a spin interaction graph. Contains a pair of vertex labels and a spin interaction.

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

- /home/sbalian/spindec/include/SpinDec/SpinInteractionEdge.h

4.48 SpinDec::SpinInteractionGraph Class Reference

Spin interaction graph from which spin Hamiltonians are built.

`#include <SpinInteractionGraph.h>`

Public Member Functions

- void **set_basis** (const SpinBasis &basis)
- void add_vertex (const SpinParameters &spin_parameters, const ThreeVector &position)
- void add_vertex (const SpinParameters &spin_parameters, const SpinBasis &basis, const ThreeVector &position)
- void add_vertex_appending_basis (const SpinParameters &spin_parameters, const ThreeVector &position)

  This vertex add method appends the new basis to basis_.

- void add_vertex_appending_basis (const SpinParameters &spin_parameters, const SpinBasis &basis, const ThreeVector &position)

  This vertex add method appends the new basis to basis_.

- void **add_edge** (unsigned int label1, unsigned int label2, const auto_ptr< SpinInteraction > &interaction)
- void **add_edges** (const vector< SpinInteractionEdge > &edges)
- unsigned int **num_vertices** () const
- unsigned int **num_edges** () const
- void **clear** ()
- const SpinBasis & **get_basis** () const
- const SpinParameters & **get_spin_parameters** (const unsigned int label) const
- const ThreeVector & **get_position** (const unsigned int label) const
- auto_ptr< SpinInteraction > **get_interaction** (const unsigned int index) const
- void **set_interaction** (const unsigned int index, const auto_ptr< SpinInteraction > &interaction)
- SpinParametersVector **spin_parameters_vector** () const
- const SpinInteractionVertex & **get_vertex** (const unsigned int label) const
- const SpinInteractionEdge & **get_edge** (const unsigned int index) const
- void join_in_place (const SpinInteractionGraph &to_join)
- void join_in_place (const SpinInteractionGraph &to_join, const std::vector< SpinInteractionEdge > &edges)
- SpinInteractionGraph join (const SpinInteractionGraph &to_join) const
- SpinInteractionGraph join (const SpinInteractionGraph &to_join, const std::vector< SpinInteractionEdge > &edges) const
- void **set_position** (const UInt label, const ThreeVector &position)
- const SpinInteractionVertex & **get_vertex1** (const UInt index) const
- const SpinInteractionVertex & **get_vertex2** (const UInt index) const
- void **set_positions** (const UIntArray &vertex_labels, const vector< ThreeVector > &positions)

Private Member Functions

- void **quit_if_vertex_label_out_of_bounds** (const unsigned int label) const
- void **quit_if_edge_index_out_of_bounds** (const unsigned int index) const
- void **set_vertex** (const unsigned int label, const SpinInteractionVertex &vertex)
- void **set_edge** (const unsigned int index, const SpinInteractionEdge &edge)

Private Attributes

- vector< SpinInteractionVertex > vertices_

  Vertices.

- vector< SpinInteractionEdge > edges_

  Edges.

- SpinBasis **basis_**

4.48.1 Detailed Description

Spin interaction graph from which spin Hamiltonians are built.

4.48.2 Member Function Documentation

4.48.2.1 void SpinDec::SpinInteractionGraph::add_vertex (const SpinParameters & *spin_parameters*, const SpinBasis & *basis*, const ThreeVector & *position*)

This vertex add method combines the new basis to basis_ (like tensor product).

4.48.2.2 void SpinDec::SpinInteractionGraph::add_vertex (const SpinParameters & *spin_parameters*, const ThreeVector & *position*)

This vertex add method combines the new basis to basis_ (like tensor product). Basis built from spin parameters.

4.48.2.3 SpinInteractionGraph SpinDec::SpinInteractionGraph::join (const SpinInteractionGraph & *to_join*, const std::vector< SpinInteractionEdge > & *edges*) const

Like join_in_place(const SpinInteractionGraph &, const std::vector<SpinInteractionEdge> &) but the original graph remains unchanged and output is a new graph.

4.48.2.4 SpinInteractionGraph SpinDec::SpinInteractionGraph::join (const SpinInteractionGraph & *to_join*) const

Like join_in_place(const SpinInteractionGraph &) but the original graph remains unchanged and output is a new graph.

4.48.2.5 void SpinDec::SpinInteractionGraph::join_in_place (const SpinInteractionGraph & *to_join*, const std::vector< SpinInteractionEdge > & *edges*)

Same as join_in_place(const SpinInteractionGraph &), but with edges connecting the two graphs. Edges should be valid for the graph after joining!

4.48.2.6 void SpinDec::SpinInteractionGraph::join_in_place (const SpinInteractionGraph & *to_join*)

Adds input graph to current graph, preserving all vertices and edges. Vertex labels of given graph $0, 1, 2, \ldots$ become $n, n+1, n+2, \ldots$, where n is the number of vertices of the original graph. There are no edges connecting the two graphs. Uses the add_vertex method (combining bases).

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

- /home/sbalian/spindec/include/SpinDec/SpinInteractionGraph.h

4.49 SpinDec::SpinInteractionVertex Class Reference

Vertex for a spin interaction graph.

`#include <SpinInteractionVertex.h>`

Public Member Functions

- SpinInteractionVertex (const UInt label, const SpinParameters &spin_parameters, const ThreeVector &position)

  Zeeman basis built from spin parameters.

- **SpinInteractionVertex** (const UInt label, const SpinParameters &spin_parameters, const SpinBasis &basis, const ThreeVector &position)
- UInt **get_label** () const
- const SpinParameters & **get_spin_parameters** () const
- const SpinBasis & **get_basis** () const
- const ThreeVector & **get_position** () const
- void **set_position** (const ThreeVector &position)

Private Attributes

- UInt **label_**
- SpinParameters **spin_parameters_**
- SpinBasis **basis_**
- ThreeVector **position_**

4.49.1 Detailed Description

Vertex for a spin interaction graph. Contains a label, spin parameters, a spin basis and a position in real space.

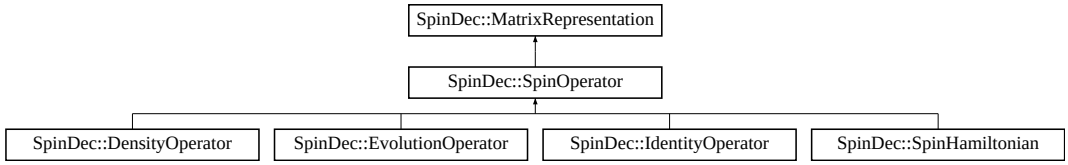
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinInteractionVertex.h

4.50 SpinDec::SpinOperator Class Reference

Quantum spin operator in the Zeeman basis.

`#include <SpinOperator.h>`Inheritance diagram for SpinDec::SpinOperator::



Public Member Functions

- **SpinOperator** (const ComplexMatrix &matrix, const SpinBasis &basis)
- SpinOperator (const SpinBasis &basis)

  Zero matrix.

- const ComplexMatrix & **get_matrix** () const
- void **set_matrix** (const ComplexMatrix &matrix)
- const CDouble & **get_element** (const UInt i, const UInt j) const
- void **set_element** (const UInt i, const UInt j, const CDouble &element)
- void set_element (const UInt i, const UInt j, const double element)

  Stored as complex.

- void **add_to_element** (const UInt i, const UInt j, const CDouble &to_add)
- SpinOperator operator^ (const SpinOperator &rhs) const

  Operators: tensor product ^, Bases: combine ^ (like tensor product).

- SpinState **operator*** (const SpinState &operand) const
- SpinOperator **operator+** (const SpinOperator &rhs) const
- SpinOperator **operator-** (const SpinOperator &rhs) const
- virtual void set_zero ()

  Set all elements to zero.

- virtual auto_ptr< MatrixRepresentation > **clone** () const

Protected Member Functions

- virtual void **quit_if_dimension_mismatch** () const

Protected Attributes

- ComplexMatrix **matrix_**

Friends

- std::ostream & operator<< (std::ostream &os, SpinOperator const &spin_operator)

  Print with cout.

4.50.1 Detailed Description

Quantum spin operator in the Zeeman basis.

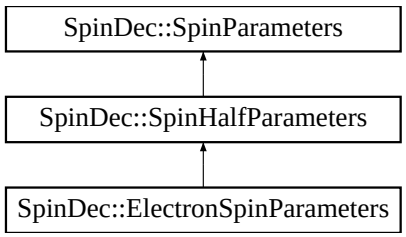
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinOperator.h

4.51 SpinDec::SpinParameters Class Reference

Contains basic parameters associated with a spin.

`#include <SpinParameters.h>`Inheritance diagram for SpinDec::SpinParameters::



Public Member Functions

- **SpinParameters** (const double quantum_number, const double gyromagnetic_ratio)
- double **get_quantum_number** () const
- double **get_gyromagnetic_ratio** () const
- UInt **get_multiplicity** () const

Protected Member Functions

- UInt **multiplicity** (const double quantum_number) const

Protected Attributes

- double **quantum_number_**
- double **gyromagnetic_ratio_**
- UInt **multiplicity_**

Friends

- std::ostream & operator<< (std::ostream &os, SpinParameters const &spin_parameters)

  Print with cout.

4.51.1 Detailed Description

Contains basic parameters associated with a spin.

- Quantum number S.
- Gyromagnetic ratio γ [M rad s^{-1} T^{-1}].
- Spin multiplicity $2S + 1$.

$\gamma = \omega/B$ where ω is the Larmor frequency in M rad s^{-1} and B is the magnetic field in Tesla (T).

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

- /home/sbalian/spindec/include/SpinDec/SpinParameters.h

4.52 SpinDec::SpinParametersVector Class Reference

Multiple SpinParameters container.

`#include <SpinParametersVector.h>`

Public Member Functions

- SpinParametersVector (const SpinParameters &spin_parameters)
  First element set.
- SpinParameters & operator[ ] (const UInt index)
  Get and set like std::vector.
- const SpinParameters & operator[ ] (const UInt index) const
  Get and set like std::vector.
- void push_back (const SpinParameters &spin_parameters)
  Add element.
- UInt size () const
  Number of SpinParameter objects.
- void clear ()
  Clear the std::vector data member.
- UInt multiplicity () const
  Total spin multiplicity = product of individual multiplicities.

Private Attributes

- std::vector< SpinParameters > **spin_parameters_vector_**

4.52.1 Detailed Description

Multiple SpinParameters container.

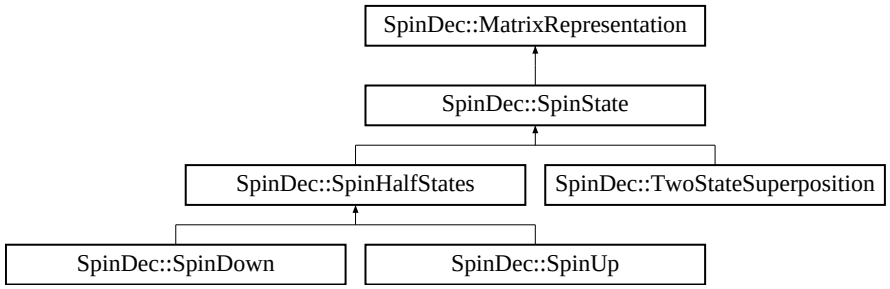
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinParametersVector.h

4.53 SpinDec::SpinState Class Reference

Quantum spin state in the Zeeman basis.

`#include <SpinState.h>`Inheritance diagram for SpinDec::SpinState::



Public Member Functions

- **SpinState** (const ComplexVector &state_vector, const SpinBasis &basis)
- SpinState (const SpinBasis &basis)

  Zero state vector.

- const ComplexVector & **get_state_vector** () const
- virtual void **set_state_vector** (const ComplexVector &state_vector)
- const CDouble & **get_element** (const UInt index) const
- virtual void **set_element** (const UInt index, const CDouble &element)
- virtual void set_element (const UInt index, const double element)

  Stored as complex.

- SpinState operator^ (const SpinState &rhs) const

  States: tensor product ^, Bases: combine ^ (like tensor product).

- CDouble operator∗ (const SpinState &rhs) const

  Inner product.

- SpinOperator operator% (const SpinState &rhs) const

  Outer product.

- SpinState **operator+** (const SpinState &rhs) const
- SpinState **operator-** (const SpinState &rhs) const
- SpinState **operator**∗ (const CDouble &c) const
- void time_evolve (const ComplexMatrix &unitary_evolution_matrix)

  Multiplies state_vector_ with unitary_evolution_matrix.

- virtual void set_zero ()

  Set all elements to zero.

- void **normalize** ()
- SpinState **normalized** () const
- virtual auto_ptr< MatrixRepresentation > **clone** () const

Protected Member Functions

- virtual void **quit_if_dimension_mismatch** () const

Protected Attributes

- ComplexVector **state_vector_**

4.53.1 Detailed Description

Quantum spin state in the Zeeman basis.

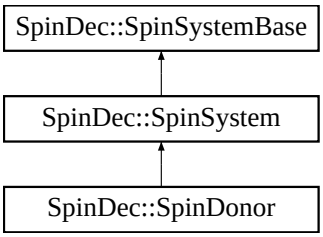
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinState.h

4.54 SpinDec::SpinSystem Class Reference

Diagonalizable concrete spin system.

`#include <SpinSystem.h>`Inheritance diagram for SpinDec::SpinSystem::



Public Member Functions

- **SpinSystem** (const SpinHamiltonian &hamiltonian)
- virtual UInt dimension () const

  Hamiltonian dimension.

- virtual auto_ptr< SpinSystemBase > **clone** () const

Protected Member Functions

- virtual void solve_once ()

  Calculate and set energies and eigenstates.

- virtual void **check_level_label** (const UInt level_label) const

Protected Attributes

- HermitianEigenspectrum **eigenspectrum_**

4.54.1 Detailed Description

Diagonalizable concrete spin system.

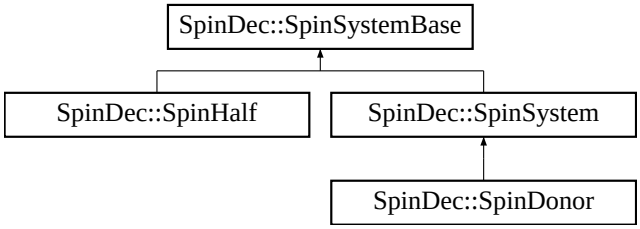
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinSystem.h

4.55 SpinDec::SpinSystemBase Class Reference

Abstract base class for spin systems.

`#include <SpinSystemBase.h>`Inheritance diagram for SpinDec::SpinSystemBase::



Public Member Functions

- **SpinSystemBase** (const SpinHamiltonian &hamiltonian)
- const SpinHamiltonian & **get_hamiltonian** () const
- virtual SpinState eigenstate (const UInt level_label)

  Levels 0,1,2, ... dimension(Hamiltonian)-1 (some levels may be excluded!).

- virtual double energy (const UInt level_label)

  Energy eigenvalue in M rad s^{-1}.

- void **set_state** (const SpinState &state)
- void set_state (const UInt level_label)

  Sets state to the level_label'th eigenstate.

- void set_state (const CDouble &c0, const UInt level_label0, const CDouble &c1, const UInt level_label1)
- const SpinState & **get_state** () const
- EvolutionOperator **evolution_operator** (const double time)
- virtual UInt dimension () const =0

  Hamiltonian dimension.

- const ComplexMatrix & **get_eigenvector_matrix** ()
- const RealVector & **get_eigenvalue_vector** ()
- virtual PiPulse pi_pulse (const UInt level_label1, const UInt level_label2)

  π-pulse (refocusing pulse).

- void print (const char option)
- virtual auto_ptr< SpinSystemBase > **clone** () const =0
- void **update_positions** (const UIntArray &vertex_labels, const vector< ThreeVector > &positions)

Protected Member Functions

- virtual void solve_once ()=0

  Calculate and set energies and eigenstates.

- virtual void **check_level_label** (const UInt level_label) const =0

Protected Attributes

- SpinHamiltonian **hamiltonian_**
- RealVector **energies_**
- ComplexMatrix **eigenstates_**
- SpinState **state_**
- bool is_solved_

  So as not to solve for eigenvectors and eigenvalues more than once.

- bool **is_state_set_**

4.55.1 Detailed Description

Abstract base class for spin systems.

4.55.2 Member Function Documentation

4.55.2.1 void SpinDec::SpinSystemBase::print (const char *option*)

Print options:

- e: energies.
- E: eigenstates.
- H: Hamiltonian.

4.55.2.2 void SpinDec::SpinSystemBase::set_state (const CDouble & *c0*, const UInt *level_label0*, const CDouble & *c1*, const UInt *level_label1*)

Sets the state to a superposition state of the eigenstates labelled $|0\rangle$ and $|1\rangle$. Also normalizes the state.

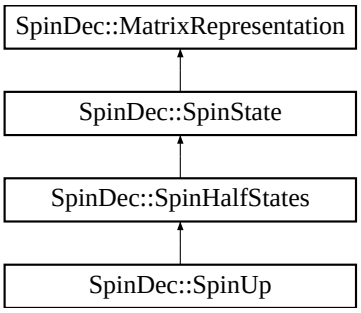
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/SpinSystemBase.h

4.56 SpinDec::SpinUp Class Reference

Spin up state.

`#include <SpinUp.h>`Inheritance diagram for SpinDec::SpinUp::



Public Member Functions

- SpinUp ()

  Default for positive gyromagnetic ratio.

- **SpinUp** (const SpinHalfParameters &spin_half_parameters)

Private Member Functions

- virtual void init (const double gyromagnetic_ratio)

4.56.1 Detailed Description

Spin up state. For a positive (or zero) gyromagnetic ratio (and positive spin quantum number) this is (1,0) in the (0.5,-0.5) basis. See also SpinDown.

4.56.2 Member Function Documentation

4.56.2.1 virtual void SpinDec::SpinUp::init (const double *gyromagnetic_ratio*) [private, virtual]

Depending on sign of gyromagnetic_ratio, spin up or spin down. See derived classes SpinUp and SpinDown.

Implements SpinDec::SpinHalfStates.

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

- /home/sbalian/spindec/include/SpinDec/SpinUp.h

4.57 SpinDec::StringOptions Class Reference

String flags composed of A-Z, a-z, each character representing a flag.

`#include <StringOptions.h>`

Public Member Functions

- **StringOptions** (const string &options)
- bool **found_option** (const char option) const
- bool **is_empty** () const

Private Attributes

- string **options_**

4.57.1 Detailed Description

String flags composed of A-Z, a-z, each character representing a flag.

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

- /home/sbalian/spindec/include/SpinDec/StringOptions.h

4.58 SpinDec::TimeArray Class Reference

Time array in microseconds.

`#include <TimeArray.h>`

Public Member Functions

- TimeArray (const double initial_time, const double final_time, const UInt num_steps)

  Calls initialize().

- **TimeArray** (const double single_time)
- bool operator== (const TimeArray &time_array) const

  is time_vector == time_vector_ ? (element by element).

- void logarithmic_time ()
- double **get_time** (const UInt index) const
- const DoubleArray & **get_time_vector** () const
- UInt num_steps () const

  dimension_ - 1.

- UInt **get_dimension** () const
- void **scale_time** (const double scalar)

Private Member Functions

- void clear ()

  std::clear time_vector_ and set dimension to zero.

- void initialize (const double initial_time, const double final_time, const UInt num_steps)

  Clears existing time sets new time to zeros.

Private Attributes

- DoubleArray time_vector_

  Microseconds.

- UInt dimension_

  Evolution grid size (1D) (number of time steps + 1).

Friends

- std::ostream & operator<< (std::ostream &os, TimeArray const &time_array)

  Print with cout.

4.58.1 Detailed Description

Time array in microseconds.

4.58.2 Member Function Documentation

4.58.2.1 void SpinDec::TimeArray::logarithmic_time ()

Convert to initial_time ... final_time with logarithmic steps (base 10) = num_steps.

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

- /home/sbalian/spindec/include/SpinDec/TimeArray.h

4.59 SpinDec::TimeEvolution Class Reference

Time evolution of a complex variable.

`#include <TimeEvolution.h>`

Public Member Functions

- **TimeEvolution** (const TimeArray &time_array, const CDoubleArray &evolution)
- **TimeEvolution** (const TimeArray &time_array)
- const CDouble & **evolution** (const UInt index) const
- void **set_evolution_zeros** ()
- void **set_evolution_ones** ()
- const CDoubleArray & **get_evolution** () const
- const TimeArray & **get_time_array** () const
- UInt dimension () const

  Number of time steps.

- TimeEvolution operator+ (const TimeEvolution &to_add) const

  Adds evolutions element by element.

- TimeEvolution operator∗ (const TimeEvolution &to_multiply) const

  Multiplies evolutions element by element.

- TimeEvolution operator/ (const TimeEvolution &to_divide) const

  Divides evolutions element by element.

- void **print** () const
- void **print_real** () const
- void **print_imag** () const
- void **print_abs** () const
- void **print** (const string &file_name) const
- void **print_real** (const string &file_name) const
- void **print_imag** (const string &file_name) const
- void **print_abs** (const string &file_name) const
- void **scale_time** (const double scalar)
- void finite_zeros ()

  If evolution $\leq 10^{-4}$*, set to* 10^{-4}.

- bool has_greater_than_one () const

  True if at least one time step has abs(evolution) exceeding unity.

Private Member Functions

- void **print** (const char option) const
- void **print** (const string &file_name, const char option) const

Private Attributes

- TimeArray time_array_

  In microseconds.

- CDoubleArray **evolution_**

4.59.1 Detailed Description

Time evolution of a complex variable. Time in microseconds.

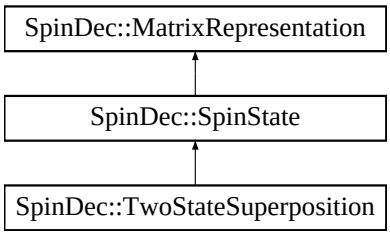
The documentation for this class was generated from the following file:

- /home/sbalian/spindec/include/SpinDec/TimeEvolution.h

4.60 SpinDec::TwoStateSuperposition Class Reference

2-level superposition state.

`#include <TwoStateSuperposition.h>`Inheritance diagram for SpinDec::TwoStateSuperposition::



Public Member Functions

- **TwoStateSuperposition** (const CDouble &c0, const SpinState &state0, const CDouble &c1, const SpinState &state1)
- const SpinState & **get_state0** () const
- const SpinState & **get_state1** () const
- const CDouble & **get_c0** () const
- const CDouble & **get_c1** () const

Private Attributes

- SpinState **state0_**
- SpinState **state1_**
- CDouble **c0_**
- CDouble **c1_**

4.60.1 Detailed Description

2-level superposition state.

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

- /home/sbalian/spindec/include/SpinDec/TwoStateSuperposition.h

4.61 SpinDec::UniformMagneticField Class Reference

Uniform magnetic field parallel to some direction in Cartesian coordinates.

`#include <UniformMagneticField.h>`

Public Member Functions

- UniformMagneticField (const double magnitude)

  Null direction (0 0 0).

- **UniformMagneticField** (const double magnitude, const ThreeVector &direction)
- double **get_magnitude** () const
- ThreeVector **get_direction** () const
- void **set_magnitude** (const double magnitude)
- void **set_direction** (const ThreeVector &direction)

Private Attributes

- double magnitude_

  Tesla.

- ThreeVector direction_

  Normalized.

4.61.1 Detailed Description

Uniform magnetic field parallel to some direction in Cartesian coordinates.

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

- /home/sbalian/spindec/include/SpinDec/UniformMagneticField.h

Chapter 5

File Documentation

5.1 /home/sbalian/spindec/include/SpinDec/typedefs.h File Reference

typedefs and "usings" `#include "SpinDec/config.h"`

```
#include <Eigen/Dense>
#include <complex>
#include <vector>
#include <string>
#include <iostream>
#include <memory>
#include <utility>
#include <map>
```

Typedefs

- typedef unsigned int SpinDec::UInt
  Unsigned int.
- typedef Eigen::Vector3d SpinDec::ThreeVector
  Real 3-vector.
- typedef std::complex< double > SpinDec::CDouble
  Complex double.
- typedef Eigen::MatrixXcd SpinDec::ComplexMatrix
  Matrix of complex doubles.
- typedef Eigen::VectorXcd SpinDec::ComplexVector
  Vector of complex doubles.
- typedef Eigen::VectorXd SpinDec::RealVector

Vector of doubles.

- typedef std::vector< double > SpinDec::DoubleArray

  STL vector of doubles.

- typedef std::vector< int > SpinDec::IntArray

  STL vector of integers.

- typedef std::vector< unsigned int > SpinDec::UIntArray

  STL vector of unsigned integers.

- typedef std::vector< CDouble > SpinDec::CDoubleArray

  STL vector of complex doubles.

5.1.1 Detailed Description

typedefs and "usings" Custom STL and Eigen typedefs for SpinDec. Also includes some usings, all in the SpinDec namespace.

Index

/home/sbalian/spindec/include/SpinDec/typedefs.h, 95

add_vertex
    SpinDec::SpinInteractionGraph, 73
adiabatic_level_labels_
    SpinDec::SpinDonor, 62

build_basis
    SpinDec::SpinDonor, 60
build_truncated_basis
    SpinDec::SpinDonor, 60

calc_adiabatic_level_labels
    SpinDec::SpinDonor, 60
calculate
    SpinDec::CCE, 12
complete_basis_
    SpinDec::SpinDonor, 62
cosAngleBetween
    SpinDec::BoostEigen, 9

eigenvalues_
    SpinDec::EvolutionOperator, 33
eigenvectors_
    SpinDec::EvolutionOperator, 33
energy
    SpinDec::SpinDonor, 61
EvolutionOperator
    SpinDec::EvolutionOperator, 32

fill_ising_flipflop
    SpinDec::SpinInteraction, 70
fill_site_vectors
    SpinDec::CrystalStructure, 22
fill_zeeman
    SpinDec::SpinHamiltonian, 68

get_bath_state
    SpinDec::SpinBath, 56

init
    SpinDec::SpinDown, 63
    SpinDec::SpinHalfStates, 66
    SpinDec::SpinUp, 86
int_range_centred_cube
    SpinDec::DiamondCubic, 26
intrabath_edges_
    SpinDec::SpinBath, 56

join
    SpinDec::SpinInteractionGraph, 73
join_in_place
    SpinDec::SpinInteractionGraph, 73

kElectronGyromagneticRatio
    SpinDec::Constants, 17
kPi
    SpinDec::Constants, 17
kPlanck
    SpinDec::Constants, 17
kReducedPlanck
    SpinDec::Constants, 17

logarithmic_time
    SpinDec::TimeArray, 89

maxAbsCoeff
    SpinDec::BoostEigen, 9

operator^
    SpinDec::SpinBasis, 54
operator+
    SpinDec::SpinBasis, 54
orthogonal_adiabatic_level_label
    SpinDec::SpinDonor, 61
orthogonal_level_label
    SpinDec::SpinDonor, 61
orthogonal_level_labels_
    SpinDec::SpinDonor, 62

partialTrace
    SpinDec::BoostEigen, 10
PiPulse
    SpinDec::PiPulse, 44
polarization
    SpinDec::SpinDonor, 61
print
    SpinDec::SpinSystemBase, 85

set_state
    SpinDec::SpinSystemBase, 85

set_transition
SpinDec::SpinDonor, 61
sorted_level_labels_
SpinDec::SpinDonor, 62
spectralDecomposition
SpinDec::BoostEigen, 10
SpinDec::HermitianEigenspectrum, 36
spin_system_base_
SpinDec::SpinBath, 56
SpinDec::AdiabaticLabel, 7
SpinDec::BoostEigen, 9
cosAngleBetween, 9
maxAbsCoeff, 9
partialTrace, 10
spectralDecomposition, 10
tensorProduct, 10
unitarySpectralDecomposition, 10
SpinDec::CCE, 11
calculate, 12
SpinDec::Cluster, 13
SpinDec::ClusterDatabase, 14
SpinDec::ClusterDatabaseEntry, 16
SpinDec::Constants, 17
kElectronGyromagneticRatio, 17
kPi, 17
kPlanck, 17
kReducedPlanck, 17
SpinDec::CPMG, 18
SpinDec::CPMGDephasing, 19
SpinDec::CrystalBasis, 20
SpinDec::CrystalStructure, 21
fill_site_vectors, 22
SpinDec::CSDProblem, 23
system_bath_edges_, 24
SpinDec::DensityOperator, 25
SpinDec::DiamondCubic, 26
int_range_centred_cube, 26
SpinDec::Dipolar, 27
SpinDec::Eigenspectrum, 29
SpinDec::ElectronSpinParameters, 30
SpinDec::Errors, 31
SpinDec::EvolutionOperator, 32
eigenvalues_, 33
eigenvectors_, 33
EvolutionOperator, 32
SpinDec::FileProperties, 34
SpinDec::FreeEvolution, 35
SpinDec::HermitianEigenspectrum, 36
spectralDecomposition, 36
SpinDec::Hyperfine, 37
SpinDec::HyperfineParameters, 39
SpinDec::IdentityOperator, 40
SpinDec::IdentityPulse, 41
SpinDec::LatticeVectors, 42
SpinDec::MatrixRepresentation, 43
SpinDec::PiPulse, 44
PiPulse, 44
SpinDec::Pulse, 45
SpinDec::PulseExperiment, 46
SpinDec::PulseSequence, 47
SpinDec::PulseSequenceBase, 48
SpinDec::RandomNumberGenerator, 49
uniform_c_rand, 49
SpinDec::ReducedProblem, 50
SpinDec::Sign, 51
SpinDec::SimpleCubicLatticeVectors, 52
SpinDec::SpinBasis, 53
operator^, 54
operator+, 54
SpinDec::SpinBath, 55
get_bath_state, 56
intrabath_edges_, 56
spin_system_base_, 56
SpinDec::SpinDonor, 57
adiabatic_level_labels_, 62
build_basis, 60
build_truncated_basis, 60
calc_adiabatic_level_labels, 60
complete_basis_, 62
energy, 61
orthogonal_adiabatic_level_label, 61
orthogonal_level_label, 61
orthogonal_level_labels_, 62
polarization, 61
set_transition, 61
sorted_level_labels_, 62
SpinDonor, 60
transition_level_labels_, 62
SpinDec::SpinDown, 63
init, 63
SpinDec::SpinHalf, 64
SpinDec::SpinHalfParameters, 65
SpinDec::SpinHalfStates, 66
init, 66
SpinDec::SpinHamiltonian, 67
fill_zeeman, 68
SpinDec::SpinInteraction, 69
fill_ising_flipflop, 70
SpinDec::SpinInteractionEdge, 71
SpinDec::SpinInteractionGraph, 72
add_vertex, 73
join, 73
join_in_place, 73
SpinDec::SpinInteractionVertex, 75
SpinDec::SpinOperator, 76
SpinDec::SpinParameters, 78
SpinDec::SpinParametersVector, 80
SpinDec::SpinState, 81

SpinDec::SpinSystem, 83
SpinDec::SpinSystemBase, 84
    print, 85
    set_state, 85
SpinDec::SpinUp, 86
    init, 86
SpinDec::StringOptions, 87
SpinDec::TimeArray, 88
    logarithmic_time, 89
SpinDec::TimeEvolution, 90
SpinDec::TwoStateSuperposition, 92
SpinDec::UniformMagneticField, 93
SpinDonor
    SpinDec::SpinDonor, 60
system_bath_edges_
    SpinDec::CSDProblem, 24

tensorProduct
    SpinDec::BoostEigen, 10
transition_level_labels_
    SpinDec::SpinDonor, 62

uniform_c_rand
    SpinDec::RandomNumberGenerator, 49
unitarySpectralDecomposition
    SpinDec::BoostEigen, 10