Quantum++

v1.0-rc3

Generated by Doxygen 1.8.13

Contents

1 Quantum++ 1

2 Namespace Index 3

2.1 Namespace List 3

3 Hierarchical Index 5

3.1 Class Hierarchy 5

4 Class Index 7

4.1 Class List 7

5 File Index 11

5.1 File List 11

6 Namespace Documentation 13

6.1 qpp Namespace Reference 13

6.1.1 Detailed Description 22

6.1.2 Typedef Documentation 22

6.1.2.1 bigint 22

6.1.2.2 bra 22

6.1.2.3 cmat 22

6.1.2.4 cplx 22

6.1.2.5 dmat 22

6.1.2.6 dyn_col_vect 23

6.1.2.7 dyn_mat 23

6.1.2.8 dyn_row_vect 23

6.1.2.9 idx 23
6.1.2.10 ket 24
6.1.2.11 to_void 24
6.1.3 Function Documentation 24
6.1.3.1 apply() [1/5] 24
6.1.3.2 apply() [2/5] 25
6.1.3.3 apply() [3/5] 25
6.1.3.4 apply() [4/5] 26
6.1.3.5 apply() [5/5] 26
6.1.3.6 applyCTRL() [1/2] 27
6.1.3.7 applyCTRL() [2/2] 27
6.1.3.8 avg() 28
6.1.3.9 choi2kraus() 28
6.1.3.10 choi2super() 29
6.1.3.11 compperm() 29
6.1.3.12 concurrence() 30
6.1.3.13 contfrac2x() 30
6.1.3.14 cor() 30
6.1.3.15 cov() 31
6.1.3.16 disp() [1/5] 31
6.1.3.17 disp() [2/5] 32
6.1.3.18 disp() [3/5] 32
6.1.3.19 disp() [4/5] 33
6.1.3.20 disp() [5/5] 33
6.1.3.21 egcd() 34
6.1.3.22 entanglement() [1/2] 34
6.1.3.23 entanglement() [2/2] 35
6.1.3.24 entropy() [1/2] 35
6.1.3.25 entropy() [2/2] 36
6.1.3.26 factors() 36

6.1.3.27 gcd() [1/2] 36
6.1.3.28 gcd() [2/2] 37
6.1.3.29 gconcurrence() 37
6.1.3.30 invperm() 38
6.1.3.31 ip() [1/2] 38
6.1.3.32 ip() [2/2] 39
6.1.3.33 isprime() 39
6.1.3.34 kraus2choi() 39
6.1.3.35 kraus2super() 40
6.1.3.36 lcm() [1/2] 40
6.1.3.37 lcm() [2/2] 41
6.1.3.38 load() 41
6.1.3.39 loadMATLAB() [1/2] 42
6.1.3.40 loadMATLAB() [2/2] 42
6.1.3.41 lognegativity() [1/2] 43
6.1.3.42 lognegativity() [2/2] 44
6.1.3.43 marginalX() 44
6.1.3.44 marginalY() 44
6.1.3.45 measure() [1/9] 45
6.1.3.46 measure() [2/9] 45
6.1.3.47 measure() [3/9] 46
6.1.3.48 measure() [4/9] 46
6.1.3.49 measure() [5/9] 47
6.1.3.50 measure() [6/9] 47
6.1.3.51 measure() [7/9] 48
6.1.3.52 measure() [8/9] 49
6.1.3.53 measure() [9/9] 49
6.1.3.54 measure_seq() [1/2] 50
6.1.3.55 measure_seq() [2/2] 51
6.1.3.56 modinv() 51

6.1.3.57 modmul() 52
6.1.3.58 modpow() 52
6.1.3.59 negativity() [1/2] 53
6.1.3.60 negativity() [2/2] 53
6.1.3.61 omega() 53
6.1.3.62 operator"""""" _i() [1/2] 55
6.1.3.63 operator"""""" _i() [2/2] 55
6.1.3.64 ptrace() [1/2] 55
6.1.3.65 ptrace() [2/2] 56
6.1.3.66 ptrace1() [1/2] 56
6.1.3.67 ptrace1() [2/2] 57
6.1.3.68 ptrace2() [1/2] 57
6.1.3.69 ptrace2() [2/2] 58
6.1.3.70 ptranspose() [1/2] 58
6.1.3.71 ptranspose() [2/2] 59
6.1.3.72 qmutualinfo() [1/2] 59
6.1.3.73 qmutualinfo() [2/2] 60
6.1.3.74 rand() [1/5] 60
6.1.3.75 rand() [2/5] 61
6.1.3.76 rand() [3/5] 61
6.1.3.77 rand() [4/5] 62
6.1.3.78 rand() [5/5] 62
6.1.3.79 randH() 63
6.1.3.80 randidx() 63
6.1.3.81 randket() 64
6.1.3.82 randkraus() 64
6.1.3.83 randn() [1/4] 64
6.1.3.84 randn() [2/4] 65
6.1.3.85 randn() [3/4] 65
6.1.3.86 randn() [4/4] 66

6.1.3.87 randperm() 66
6.1.3.88 randprime() 67
6.1.3.89 randprob() 67
6.1.3.90 randrho() 67
6.1.3.91 randU() 68
6.1.3.92 randV() 68
6.1.3.93 renyi() [1/2] 69
6.1.3.94 renyi() [2/2] 69
6.1.3.95 save() 70
6.1.3.96 saveMATLAB() [1/2] 70
6.1.3.97 saveMATLAB() [2/2] 71
6.1.3.98 schmidtA() [1/2] 71
6.1.3.99 schmidtA() [2/2] 72
6.1.3.100 schmidtB() [1/2] 72
6.1.3.101 schmidtB() [2/2] 72
6.1.3.102 schmidtcoeffs() [1/2] 73
6.1.3.103 schmidtcoeffs() [2/2] 73
6.1.3.104 schmidtprobs() [1/2] 74
6.1.3.105 schmidtprobs() [2/2] 74
6.1.3.106 sigma() 75
6.1.3.107 super2choi() 75
6.1.3.108 syspermute() [1/2] 77
6.1.3.109 syspermute() [2/2] 77
6.1.3.110 tsallis() [1/2] 78
6.1.3.111 tsallis() [2/2] 78
6.1.3.112 uniform() 79
6.1.3.113 var() 79
6.1.3.114 x2contfrac() 79
6.1.4 Variable Documentation 80
6.1.4.1 chop 80

6.1.4.2 ee 80
6.1.4.3 eps 80
6.1.4.4 infty 81
6.1.4.5 maxn 81
6.1.4.6 pi 81
6.2 qpp::exception Namespace Reference 81
6.2.1 Detailed Description 82
6.3 qpp::experimental Namespace Reference 83
6.3.1 Detailed Description 83
6.4 qpp::internal Namespace Reference 83
6.4.1 Detailed Description 84
6.4.2 Function Documentation 84
6.4.2.1 check_cvector() 84
6.4.2.2 check_dims() 84
6.4.2.3 check_dims_match_cvect() 85
6.4.2.4 check_dims_match_mat() 85
6.4.2.5 check_dims_match_rvect() 85
6.4.2.6 check_eq_dims() 85
6.4.2.7 check_matching_sizes() 85
6.4.2.8 check_nonzero_size() 85
6.4.2.9 check_perm() 86
6.4.2.10 check_qubit_cvector() 86
6.4.2.11 check_qubit_matrix() 86
6.4.2.12 check_qubit_rvector() 86
6.4.2.13 check_qubit_vector() 86
6.4.2.14 check_rvector() 86
6.4.2.15 check_square_mat() 87
6.4.2.16 check_subsys_match_dims() 87
6.4.2.17 check_vector() 87
6.4.2.18 dirsum2() 87
6.4.2.19 get_dim_subsys() 87
6.4.2.20 get_num_subsys() 87
6.4.2.21 kron2() 88
6.4.2.22 multiidx2n() 88
6.4.2.23 n2multiidx() 88
6.4.2.24 variadic_vector_emplace() [1/2] 88
6.4.2.25 variadic_vector_emplace() [2/2] 88

7 Class Documentation 89

7.1 qpp::experimental::Bit_circuit Class Reference 89
7.1.1 Member Function Documentation 90
7.1.1.1 CNOT() 91
7.1.1.2 FRED() 91
7.1.1.3 NOT() 91
7.1.1.4 reset() 91
7.1.1.5 SWAP() 91
7.1.1.6 TOF() 91
7.1.1.7 X() 91
7.1.2 Member Data Documentation 92
7.1.2.1 gate_count 92
7.2 qpp::Bit_circuit Class Reference 92
7.2.1 Detailed Description 92
7.3 qpp::Codes Class Reference 92
7.3.1 Detailed Description 93
7.3.2 Member Enumeration Documentation 93
7.3.2.1 Type 94
7.3.3 Constructor & Destructor Documentation 94
7.3.3.1 Codes() 94
7.3.3.2 ∼Codes() 94
7.3.4 Member Function Documentation 94
7.3.4.1 codeword() 94
7.3.5 Friends And Related Function Documentation 95
7.3.5.1 internal::Singleton< const Codes > 95
7.4 qpp::exception::CustomException Class Reference 95
7.4.1 Detailed Description 96
7.4.2 Constructor & Destructor Documentation 96
7.4.2.1 CustomException() 97
7.4.3 Member Function Documentation 97

7.4.3.1 type_description() 97
7.4.4 Member Data Documentation 97
7.4.4.1 what_ 97
7.5 qpp::exception::DimsInvalid Class Reference 98
7.5.1 Detailed Description 99
7.5.2 Member Function Documentation 99
7.5.2.1 type_description() 99
7.6 qpp::exception::DimsMismatchCvector Class Reference 99
7.6.1 Detailed Description 101
7.6.2 Member Function Documentation 101
7.6.2.1 type_description() 101
7.7 qpp::exception::DimsMismatchMatrix Class Reference 101
7.7.1 Detailed Description 102
7.7.2 Member Function Documentation 102
7.7.2.1 type_description() 103
7.8 qpp::exception::DimsMismatchRvector Class Reference 103
7.8.1 Detailed Description 104
7.8.2 Member Function Documentation 104
7.8.2.1 type_description() 105
7.9 qpp::exception::DimsMismatchVector Class Reference 105
7.9.1 Detailed Description 106
7.9.2 Member Function Documentation 106
7.9.2.1 type_description() 107
7.10 qpp::exception::DimsNotEqual Class Reference 107
7.10.1 Detailed Description 108
7.10.2 Member Function Documentation 108
7.10.2.1 type_description() 108
7.11 qpp::internal::Display_Impl_ Struct Reference 109
7.11.1 Member Function Documentation 109
7.11.1.1 display_impl_() 109

7.12 qpp::experimental::Dynamic_bitset Class Reference 110

7.12.1 Member Typedef Documentation 112

7.12.1.1 storage_type 112

7.12.1.2 value_type 112

7.12.2 Constructor & Destructor Documentation 112

7.12.2.1 Dynamic_bitset() 112

7.12.3 Member Function Documentation 112

7.12.3.1 all() 112

7.12.3.2 any() 113

7.12.3.3 count() 113

7.12.3.4 data() 113

7.12.3.5 flip() `[1/2]` 113

7.12.3.6 flip() `[2/2]` 114

7.12.3.7 get() 114

7.12.3.8 index_() 114

7.12.3.9 none() 114

7.12.3.10 offset_() 115

7.12.3.11 operator"!=() 115

7.12.3.12 operator==() 115

7.12.3.13 rand() `[1/2]` 116

7.12.3.14 rand() `[2/2]` 116

7.12.3.15 reset() `[1/2]` 116

7.12.3.16 reset() `[2/2]` 117

7.12.3.17 set() `[1/2]` 117

7.12.3.18 set() `[2/2]` 117

7.12.3.19 size() 117

7.12.3.20 storage_size() 118

7.12.3.21 to_string() 118

7.12.4 Friends And Related Function Documentation 118

7.12.4.1 operator<< 118

7.12.5 Member Data Documentation . . . 119
7.12.5.1 N_ . . . 119
7.12.5.2 storage_size_ . . . 119
7.12.5.3 v_ . . . 119
7.13 qpp::Dynamic_bitset Class Reference . . . 119
7.13.1 Detailed Description . . . 119
7.14 qpp::exception::Exception Class Reference . . . 120
7.14.1 Detailed Description . . . 121
7.14.2 Constructor & Destructor Documentation . . . 122
7.14.2.1 Exception() . . . 122
7.14.3 Member Function Documentation . . . 122
7.14.3.1 type_description() . . . 122
7.14.3.2 what() . . . 122
7.14.4 Member Data Documentation . . . 123
7.14.4.1 where_ . . . 123
7.15 qpp::experimental::Bit_circuit::Gate_count Struct Reference . . . 123
7.15.1 Member Data Documentation . . . 123
7.15.1.1 CNOT . . . 123
7.15.1.2 FRED . . . 123
7.15.1.3 NOT . . . 124
7.15.1.4 SWAP . . . 124
7.15.1.5 TOF . . . 124
7.15.1.6 X . . . 124
7.16 qpp::Gates Class Reference . . . 124
7.16.1 Detailed Description . . . 126
7.16.2 Constructor & Destructor Documentation . . . 126
7.16.2.1 Gates() . . . 127
7.16.2.2 ∼Gates() . . . 127
7.16.3 Member Function Documentation . . . 127
7.16.3.1 CTRL() . . . 127

7.16.3.2 expandout() [1/3] 128
7.16.3.3 expandout() [2/3] 128
7.16.3.4 expandout() [3/3] 129
7.16.3.5 Fd() 129
7.16.3.6 Id() 130
7.16.3.7 Rn() 130
7.16.3.8 Xd() 131
7.16.3.9 Zd() 131
7.16.4 Friends And Related Function Documentation 131
7.16.4.1 internal::Singleton< const Gates > 132
7.16.5 Member Data Documentation 132
7.16.5.1 CNOT 132
7.16.5.2 CNOTba 132
7.16.5.3 CZ 132
7.16.5.4 FRED 132
7.16.5.5 H 132
7.16.5.6 Id2 133
7.16.5.7 S 133
7.16.5.8 SWAP 133
7.16.5.9 T 133
7.16.5.10 TOF 133
7.16.5.11 X 133
7.16.5.12 Y 134
7.16.5.13 Z 134
7.17 qpp::IDisplay Class Reference 134
7.17.1 Detailed Description 135
7.17.2 Constructor & Destructor Documentation 135
7.17.2.1 IDisplay() [1/3] 135
7.17.2.2 IDisplay() [2/3] 136
7.17.2.3 IDisplay() [3/3] 136

7.17.2.4 ∼IDisplay() 136
7.17.3 Member Function Documentation 136
7.17.3.1 display() 136
7.17.3.2 operator=() [1/2] 136
7.17.3.3 operator=() [2/2] 137
7.17.4 Friends And Related Function Documentation 137
7.17.4.1 operator<< 137
7.18 qpp::Init Class Reference 137
7.18.1 Detailed Description 138
7.18.2 Constructor & Destructor Documentation 138
7.18.2.1 Init() 138
7.18.2.2 ∼Init() 139
7.18.3 Friends And Related Function Documentation 139
7.18.3.1 internal::Singleton< const Init > 139
7.19 qpp::internal::IOManipEigen Class Reference 139
7.19.1 Constructor & Destructor Documentation 140
7.19.1.1 IOManipEigen() [1/2] 140
7.19.1.2 IOManipEigen() [2/2] 140
7.19.2 Member Function Documentation 140
7.19.2.1 display() 140
7.19.3 Member Data Documentation 141
7.19.3.1 A_ 141
7.19.3.2 chop_ 141
7.20 qpp::internal::IOManipPointer< PointerType > Class Template Reference 141
7.20.1 Constructor & Destructor Documentation 142
7.20.1.1 IOManipPointer() [1/2] 143
7.20.1.2 IOManipPointer() [2/2] 143
7.20.2 Member Function Documentation 143
7.20.2.1 display() 143
7.20.2.2 operator=() 143

7.20.3 Member Data Documentation 143

7.20.3.1 end_ 144

7.20.3.2 N_ 144

7.20.3.3 p_ 144

7.20.3.4 separator_ 144

7.20.3.5 start_ 144

7.21 qpp::internal::IOManipRange< InputIterator > Class Template Reference 145

7.21.1 Constructor & Destructor Documentation 146

7.21.1.1 IOManipRange() [1/2] 146

7.21.1.2 IOManipRange() [2/2] 146

7.21.2 Member Function Documentation 146

7.21.2.1 display() 146

7.21.2.2 operator=() 147

7.21.3 Member Data Documentation 147

7.21.3.1 end_ 147

7.21.3.2 first_ 147

7.21.3.3 last_ 147

7.21.3.4 separator_ 147

7.21.3.5 start_ 147

7.22 qpp::is_complex< T > Struct Template Reference 148

7.22.1 Detailed Description 148

7.23 qpp::is_complex< std::complex< T > > Struct Template Reference 149

7.23.1 Detailed Description 149

7.24 qpp::is_iterable< T, typename > Struct Template Reference 150

7.24.1 Detailed Description 150

7.25 qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > > Struct Template Reference 151

7.25.1 Detailed Description 152

7.26 qpp::is_matrix_expression< Derived > Struct Template Reference 152

7.26.1 Detailed Description 153

7.27 qpp::make_void< Ts > Struct Template Reference 153

7.27.1 Detailed Description 153

7.27.2 Member Typedef Documentation 153

7.27.2.1 type 153

7.28 qpp::exception::MatrixMismatchSubsys Class Reference 154

7.28.1 Detailed Description 155

7.28.2 Member Function Documentation 155

7.28.2.1 type_description() 155

7.29 qpp::exception::MatrixNotCvector Class Reference 155

7.29.1 Detailed Description 157

7.29.2 Member Function Documentation 157

7.29.2.1 type_description() 157

7.30 qpp::exception::MatrixNotRvector Class Reference 157

7.30.1 Detailed Description 158

7.30.2 Member Function Documentation 158

7.30.2.1 type_description() 158

7.31 qpp::exception::MatrixNotSquare Class Reference 159

7.31.1 Detailed Description 160

7.31.2 Member Function Documentation 160

7.31.2.1 type_description() 160

7.32 qpp::exception::MatrixNotSquareNorCvector Class Reference 161

7.32.1 Detailed Description 162

7.32.2 Member Function Documentation 162

7.32.2.1 type_description() 162

7.33 qpp::exception::MatrixNotSquareNorRvector Class Reference 163

7.33.1 Detailed Description 164

7.33.2 Member Function Documentation 164

7.33.2.1 type_description() 164

7.34 qpp::exception::MatrixNotSquareNorVector Class Reference 165

7.34.1 Detailed Description 166

7.34.2 Member Function Documentation 166

7.34.2.1 type_description() 166

7.35 qpp::exception::MatrixNotVector Class Reference 167

7.35.1 Detailed Description 168

7.35.2 Member Function Documentation 168

7.35.2.1 type_description() 168

7.36 qpp::exception::NoCodeword Class Reference 169

7.36.1 Detailed Description 170

7.36.2 Member Function Documentation 170

7.36.2.1 type_description() 170

7.37 qpp::exception::NotBipartite Class Reference 171

7.37.1 Detailed Description 172

7.37.2 Member Function Documentation 172

7.37.2.1 type_description() 172

7.38 qpp::exception::NotQubitCvector Class Reference 172

7.38.1 Detailed Description 174

7.38.2 Member Function Documentation 174

7.38.2.1 type_description() 174

7.39 qpp::exception::NotQubitMatrix Class Reference 174

7.39.1 Detailed Description 175

7.39.2 Member Function Documentation 175

7.39.2.1 type_description() 175

7.40 qpp::exception::NotQubitRvector Class Reference 176

7.40.1 Detailed Description 177

7.40.2 Member Function Documentation 177

7.40.2.1 type_description() 177

7.41 qpp::exception::NotQubitSubsys Class Reference 178

7.41.1 Detailed Description 179

7.41.2 Member Function Documentation 179

7.41.2.1 type_description() 179

7.42 qpp::exception::NotQubitVector Class Reference 180

7.42.1 Detailed Description 181
7.42.2 Member Function Documentation 181
7.42.2.1 type_description() 181
7.43 qpp::exception::OutOfRange Class Reference 182
7.43.1 Detailed Description 183
7.43.2 Member Function Documentation 183
7.43.2.1 type_description() 183
7.44 qpp::exception::PermInvalid Class Reference 184
7.44.1 Detailed Description 185
7.44.2 Member Function Documentation 185
7.44.2.1 type_description() 185
7.45 qpp::exception::PermMismatchDims Class Reference 185
7.45.1 Detailed Description 187
7.45.2 Member Function Documentation 187
7.45.2.1 type_description() 187
7.46 qpp::RandomDevices Class Reference 187
7.46.1 Detailed Description 189
7.46.2 Constructor & Destructor Documentation 189
7.46.2.1 RandomDevices() 189
7.46.2.2 ∼RandomDevices() 189
7.46.3 Member Function Documentation 189
7.46.3.1 get_prng() 189
7.46.3.2 load() 189
7.46.3.3 save() 190
7.46.4 Friends And Related Function Documentation 190
7.46.4.1 internal::Singleton< RandomDevices > 190
7.46.5 Member Data Documentation 190
7.46.5.1 prng_ 190
7.46.5.2 rd_ 191
7.47 qpp::internal::Singleton< T > Class Template Reference 191

7.47.1 Detailed Description 191
7.47.2 Constructor & Destructor Documentation 192
7.47.2.1 Singleton() [1/2] 192
7.47.2.2 Singleton() [2/2] 192
7.47.2.3 ∼Singleton() 192
7.47.3 Member Function Documentation 192
7.47.3.1 get_instance() 193
7.47.3.2 get_thread_local_instance() 193
7.47.3.3 operator=() 193
7.48 qpp::exception::SizeMismatch Class Reference 193
7.48.1 Detailed Description 194
7.48.2 Member Function Documentation 194
7.48.2.1 type_description() 194
7.49 qpp::States Class Reference 195
7.49.1 Detailed Description 197
7.49.2 Constructor & Destructor Documentation 197
7.49.2.1 States() 197
7.49.2.2 ∼States() 197
7.49.3 Member Function Documentation 197
7.49.3.1 jn() 197
7.49.3.2 mes() 198
7.49.3.3 minus() 198
7.49.3.4 one() 199
7.49.3.5 plus() 199
7.49.3.6 zero() 199
7.49.4 Friends And Related Function Documentation 200
7.49.4.1 internal::Singleton< const States > 200
7.49.5 Member Data Documentation 200
7.49.5.1 b00 200
7.49.5.2 b01 200

7.49.5.3 b10 200

7.49.5.4 b11 200

7.49.5.5 GHZ 201

7.49.5.6 pb00 201

7.49.5.7 pb01 201

7.49.5.8 pb10 201

7.49.5.9 pb11 201

7.49.5.10 pGHZ 201

7.49.5.11 pW 202

7.49.5.12 px0 202

7.49.5.13 px1 202

7.49.5.14 py0 202

7.49.5.15 py1 202

7.49.5.16 pz0 202

7.49.5.17 pz1 203

7.49.5.18 W 203

7.49.5.19 x0 203

7.49.5.20 x1 203

7.49.5.21 y0 203

7.49.5.22 y1 203

7.49.5.23 z0 204

7.49.5.24 z1 204

7.50 qpp::exception::SubsysMismatchDims Class Reference 204

7.50.1 Detailed Description 205

7.50.2 Member Function Documentation 205

7.50.2.1 type_description() 205

7.51 qpp::Timer< T, CLOCK_T > Class Template Reference 206

7.51.1 Detailed Description 207

7.51.2 Constructor & Destructor Documentation 208

7.51.2.1 Timer() `[1/3]` 208

7.51.2.2 Timer() [2/3] 208
7.51.2.3 Timer() [3/3] 208
7.51.2.4 ∼Timer() 208
7.51.3 Member Function Documentation 208
7.51.3.1 display() 208
7.51.3.2 get_duration() 209
7.51.3.3 operator=() [1/2] 209
7.51.3.4 operator=() [2/2] 209
7.51.3.5 tic() 210
7.51.3.6 tics() 210
7.51.3.7 toc() 210
7.51.4 Member Data Documentation 210
7.51.4.1 end_ 210
7.51.4.2 start_ 211
7.52 qpp::exception::TypeMismatch Class Reference 211
7.52.1 Detailed Description 212
7.52.2 Member Function Documentation 212
7.52.2.1 type_description() 212
7.53 qpp::exception::UndefinedType Class Reference 213
7.53.1 Detailed Description 214
7.53.2 Member Function Documentation 214
7.53.2.1 type_description() 214
7.54 qpp::exception::Unknown Class Reference 214
7.54.1 Detailed Description 216
7.54.2 Member Function Documentation 216
7.54.2.1 type_description() 216
7.55 qpp::exception::ZeroSize Class Reference 216
7.55.1 Detailed Description 217
7.55.2 Member Function Documentation 217
7.55.2.1 type_description() 217

8 File Documentation 219

8.1 classes/codes.h File Reference 219
  8.1.1 Detailed Description 219
8.2 classes/exception.h File Reference 219
  8.2.1 Detailed Description 221
8.3 classes/gates.h File Reference 221
  8.3.1 Detailed Description 221
8.4 classes/idisplay.h File Reference 222
  8.4.1 Detailed Description 222
8.5 classes/init.h File Reference 222
  8.5.1 Detailed Description 223
8.6 classes/random_devices.h File Reference 223
  8.6.1 Detailed Description 223
8.7 classes/states.h File Reference 223
  8.7.1 Detailed Description 224
8.8 classes/timer.h File Reference 224
  8.8.1 Detailed Description 224
8.9 constants.h File Reference 224
  8.9.1 Detailed Description 225
8.10 entanglement.h File Reference 225
  8.10.1 Detailed Description 227
8.11 entropies.h File Reference 227
  8.11.1 Detailed Description 227
8.12 experimental/experimental.h File Reference 228
  8.12.1 Detailed Description 228
  8.12.2 Typedef Documentation 228
    8.12.2.1 idx 228
8.13 functions.h File Reference 229
  8.13.1 Detailed Description 229
8.14 input_output.h File Reference 229

8.14.1 Detailed Description 230
8.15 instruments.h File Reference 230
8.15.1 Detailed Description 232
8.16 internal/classes/iomanip.h File Reference 232
8.16.1 Detailed Description 232
8.17 internal/classes/singleton.h File Reference 233
8.17.1 Detailed Description 233
8.18 internal/util.h File Reference 233
8.18.1 Detailed Description 235
8.19 MATLAB/matlab.h File Reference 235
8.19.1 Detailed Description 236
8.20 number_theory.h File Reference 236
8.20.1 Detailed Description 237
8.21 operations.h File Reference 237
8.21.1 Detailed Description 239
8.22 qpp.h File Reference 239
8.22.1 Detailed Description 240
8.22.2 Macro Definition Documentation 240
8.22.2.1 QPP_UNUSED_ 240
8.23 random.h File Reference 240
8.23.1 Detailed Description 241
8.24 statistics.h File Reference 242
8.24.1 Detailed Description 242
8.25 traits.h File Reference 243
8.25.1 Detailed Description 243
8.26 types.h File Reference 244
8.26.1 Detailed Description 245
8.27 /Users/vlad/Dropbox/programming/cpp/qpp/README.md File Reference 245

Index **247**

Chapter 1

Quantum++

Version 1.0-rc2 - 6 September 2017

Build status:

Chat (questions/issues)

About

Quantum++ is a modern C++11 general purpose quantum computing library, composed solely of template header files. Quantum++ is written in standard C++11 and has very low external dependencies, using only the `Eigen 3` linear algebra header-only template library and, if available, the `OpenMP` multi-processing library.

Quantum++ is not restricted to qubit systems or specific quantum information processing tasks, being capable of simulating arbitrary quantum processes. The main design factors taken in consideration were the ease of use, high portability, and high performance. The library's simulation capabilities are only restricted by the amount of available physical memory. On a typical machine (Intel i5 8Gb RAM) Quantum++ can successfully simulate the evolution of 25 qubits in a pure state or of 12 qubits in a mixed state reasonably fast.

To report any bugs or ask for additional features/enhancements, please `submit an issue` with an appropriate label.

If you are interesting in contributing to this project, feel free to contact me. Alternatively, create a custom branch, add your contribution, then finally create a pull request. If I accept the pull request, I will merge your custom branch with the latest development branch. The latter will eventually be merged into a future release version. To contribute, you need to have a solid knowledge of C++ (preferably C++11), including templates and the standard library, a basic knowledge of quantum computing and linear algebra, and working experience with `Eigen 3`.

For additional `Eigen 3` documentation see `http://eigen.tuxfamily.org/dox/`. For a simple `Eigen 3` quick ASCII reference see `http://eigen.tuxfamily.org/dox/AsciiQuickReference.txt`.

License

`Quantum++` is distributed under the MIT license. Please see the `LICENSE` file for more details.

Installation instructions and further documentation

Please see the installation guide https://github.com/vsoftco/qpp/blob/master/INSTALL.md "'INSTALL.md'" and the comprehensive `Wiki` for further documentation and detailed examples.

The official API documentation is available in PDF and HTML formats in the `doc` folder.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

qpp
Quantum++ main namespace 13
qpp::exception
Quantum++ exception hierarchy namespace 81
qpp::experimental
Experimental/test functions/classes, do not use or modify 83
qpp::internal
Internal utility functions, do not use them directly or modify them 83

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

- qpp::Bit_circuit . . . 92
- qpp::internal::Display_Impl_ . . . 109
  - qpp::internal::IOManipEigen . . . 139
- qpp::experimental::Dynamic_bitset . . . 110
  - qpp::experimental::Bit_circuit . . . 89
- qpp::Dynamic_bitset . . . 119
- std::exception
  - qpp::exception::Exception . . . 120
    - qpp::exception::CustomException . . . 95
    - qpp::exception::DimsInvalid . . . 98
    - qpp::exception::DimsMismatchCvector . . . 99
    - qpp::exception::DimsMismatchMatrix . . . 101
    - qpp::exception::DimsMismatchRvector . . . 103
    - qpp::exception::DimsMismatchVector . . . 105
    - qpp::exception::DimsNotEqual . . . 107
    - qpp::exception::MatrixMismatchSubsys . . . 154
    - qpp::exception::MatrixNotCvector . . . 155
    - qpp::exception::MatrixNotRvector . . . 157
    - qpp::exception::MatrixNotSquare . . . 159
    - qpp::exception::MatrixNotSquareNorCvector . . . 161
    - qpp::exception::MatrixNotSquareNorRvector . . . 163
    - qpp::exception::MatrixNotSquareNorVector . . . 165
    - qpp::exception::MatrixNotVector . . . 167
    - qpp::exception::NoCodeword . . . 169
    - qpp::exception::NotBipartite . . . 171
    - qpp::exception::NotQubitCvector . . . 172
    - qpp::exception::NotQubitMatrix . . . 174
    - qpp::exception::NotQubitRvector . . . 176
    - qpp::exception::NotQubitSubsys . . . 178
    - qpp::exception::NotQubitVector . . . 180
    - qpp::exception::OutOfRange . . . 182
    - qpp::exception::PermInvalid . . . 184
    - qpp::exception::PermMismatchDims . . . 185
    - qpp::exception::SizeMismatch . . . 193
    - qpp::exception::SubsysMismatchDims . . . 204

        qpp::exception::TypeMismatch 211
        qpp::exception::UndefinedType 213
        qpp::exception::Unknown 214
        qpp::exception::ZeroSize 216
false_type
    qpp::is_complex< T > 148
    qpp::is_iterable< T, typename > 150
qpp::experimental::Bit_circuit::Gate_count 123
qpp::IDisplay 134
    qpp::internal::IOManipEigen 139
    qpp::internal::IOManipPointer< PointerType > 141
    qpp::internal::IOManipRange< InputIterator > 145
    qpp::Timer< T, CLOCK_T > 206
is_base_of
    qpp::is_matrix_expression< Derived > 152
qpp::make_void< Ts > 153
qpp::internal::Singleton< T > 191
qpp::internal::Singleton< const Codes > 191
    qpp::Codes 92
qpp::internal::Singleton< const Gates > 191
    qpp::Gates 124
qpp::internal::Singleton< const Init > 191
    qpp::Init 137
qpp::internal::Singleton< const States > 191
    qpp::States 195
qpp::internal::Singleton< RandomDevices > 191
    qpp::RandomDevices 187
true_type
    qpp::is_complex< std::complex< T > > 149
    qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > > 151

Chapter 4

Class Index

4.1 Class List

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

qpp::experimental::Bit_circuit 89

qpp::Bit_circuit

Classical reversible circuit simulator 92

qpp::Codes

Const Singleton class that defines quantum error correcting codes 92

qpp::exception::CustomException

Custom exception 95

qpp::exception::DimsInvalid

Invalid dimension(s) exception 98

qpp::exception::DimsMismatchCvector

Dimension(s) mismatch column vector size exception 99

qpp::exception::DimsMismatchMatrix

Dimension(s) mismatch matrix size exception 101

qpp::exception::DimsMismatchRvector

Dimension(s) mismatch row vector size exception 103

qpp::exception::DimsMismatchVector

Dimension(s) mismatch vector size exception 105

qpp::exception::DimsNotEqual

Dimensions not equal exception 107

qpp::internal::Display_Impl_ 109

qpp::experimental::Dynamic_bitset 110

qpp::Dynamic_bitset

Dynamic bitset class, allows the specification of the number of bits at runtime (unlike std::bitset<N>) 119

qpp::exception::Exception

Base class for generating Quantum++ custom exceptions 120

qpp::experimental::Bit_circuit::Gate_count 123

qpp::Gates

Const Singleton class that implements most commonly used gates 124

qpp::IDisplay

Abstract class (interface) that mandates the definition of virtual std::ostream& display(std::ostream& os) const 134

qpp::Init

Const Singleton class that performs additional initializations/cleanups 137

qpp::internal::IOManipEigen 139

qpp::internal::IOManipPointer< PointerType > 141
qpp::internal::IOManipRange< InputIterator > 145
qpp::is_complex< T >
Checks whether the type is a complex type 148
qpp::is_complex< std::complex< T > >
Checks whether the type is a complex number type, specialization for complex types 149
qpp::is_iterable< T, typename >
Checks whether *T* is compatible with an STL-like iterable container 150
qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > >
Checks whether *T* is compatible with an STL-like iterable container, specialization for STL-like iterable containers 151
qpp::is_matrix_expression< Derived >
Checks whether the type is an Eigen matrix expression 152
qpp::make_void< Ts >
Helper for qpp::to_void<> alias template 153
qpp::exception::MatrixMismatchSubsys
Matrix mismatch subsystems exception 154
qpp::exception::MatrixNotCvector
Matrix is not a column vector exception 155
qpp::exception::MatrixNotRvector
Matrix is not a row vector exception 157
qpp::exception::MatrixNotSquare
Matrix is not square exception 159
qpp::exception::MatrixNotSquareNorCvector
Matrix is not square nor column vector exception 161
qpp::exception::MatrixNotSquareNorRvector
Matrix is not square nor row vector exception 163
qpp::exception::MatrixNotSquareNorVector
Matrix is not square nor vector exception 165
qpp::exception::MatrixNotVector
Matrix is not a vector exception 167
qpp::exception::NoCodeword
Codeword does not exist exception 169
qpp::exception::NotBipartite
Not bi-partite exception 171
qpp::exception::NotQubitCvector
Column vector is not 2 x 1 exception 172
qpp::exception::NotQubitMatrix
Matrix is not 2 x 2 exception 174
qpp::exception::NotQubitRvector
Row vector is not 1 x 2 exception 176
qpp::exception::NotQubitSubsys
Subsystems are not qubits exception 178
qpp::exception::NotQubitVector
Vector is not 2 x 1 nor 1 x 2 exception 180
qpp::exception::OutOfRange
Parameter out of range exception 182
qpp::exception::PermInvalid
Invalid permutation exception 184
qpp::exception::PermMismatchDims
Permutation mismatch dimensions exception 185
qpp::RandomDevices
Singleton class that manages the source of randomness in the library 187
qpp::internal::Singleton< T >
Singleton policy class, used internally to implement the singleton pattern via CRTP (Curiously recurring template pattern) 191

qpp::exception::SizeMismatch
Size mismatch exception 193
qpp::States
Const Singleton class that implements most commonly used states 195
qpp::exception::SubsysMismatchDims
Subsystems mismatch dimensions exception 204
qpp::Timer< T, CLOCK_T >
Chronometer 206
qpp::exception::TypeMismatch
Type mismatch exception 211
qpp::exception::UndefinedType
Not defined for this type exception 213
qpp::exception::Unknown
Unknown exception 214
qpp::exception::ZeroSize
Object has zero size exception 216

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

constants.h
Constants 224
entanglement.h
Entanglement functions 225
entropies.h
Entropy functions 227
functions.h
Generic quantum computing functions 229
input_output.h
Input/output functions 229
instruments.h
Measurement functions 230
number_theory.h
Number theory functions 236
operations.h
Quantum operation functions 237
qpp.h
Quantum++ main header file, includes all other necessary headers 239
random.h
Randomness-related functions 240
statistics.h
Statistics functions 242
traits.h
Type traits 243
types.h
Type aliases 244
classes/codes.h
Quantum error correcting codes 219
classes/exception.h
Exceptions 219
classes/gates.h
Quantum gates 221
classes/idisplay.h
Display interface via the non-virtual interface (NVI) 222
classes/init.h
Initialization 222

classes/random_devices.h
Random devices 223
classes/states.h
Quantum states 223
classes/timer.h
Timing 224
experimental/experimental.h
Experimental/test functions/classes 228
internal/util.h
Internal utility functions 233
internal/classes/iomanip.h
Input/output manipulators 232
internal/classes/singleton.h
Singleton pattern via CRTP 233
MATLAB/matlab.h
Input/output interfacing with MATLAB 235

Chapter 6

Namespace Documentation

6.1 qpp Namespace Reference

Quantum++ main namespace.

Namespaces

- exception

  Quantum++ exception hierarchy namespace.
- experimental

  Experimental/test functions/classes, do not use or modify.
- internal

  Internal utility functions, do not use them directly or modify them.

Classes

- class Bit_circuit

  Classical reversible circuit simulator.
- class Codes

  const Singleton class that defines quantum error correcting codes
- class Dynamic_bitset

  Dynamic bitset class, allows the specification of the number of bits at runtime (unlike std::bitset< N >)
- class Gates

  const Singleton class that implements most commonly used gates
- class IDisplay

  Abstract class (interface) that mandates the definition of virtual std::ostream& display(std::ostream& os) const.
- class Init

  const Singleton class that performs additional initializations/cleanups
- struct is_complex

  Checks whether the type is a complex type.
- struct is_complex< std::complex< T > >

  Checks whether the type is a complex number type, specialization for complex types.
- struct is_iterable

  Checks whether T is compatible with an STL-like iterable container.

- struct is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > >

  Checks whether T is compatible with an STL-like iterable container, specialization for STL-like iterable containers.
- struct is_matrix_expression

  Checks whether the type is an Eigen matrix expression.
- struct make_void

  Helper for qpp::to_void<> alias template.
- class RandomDevices

  Singleton class that manages the source of randomness in the library.
- class States

  const Singleton class that implements most commonly used states
- class Timer

  Chronometer.

Typedefs

- template<typename... Ts>
  using to_void = typename make_void< Ts... >::type

  Alias template that implements the proposal for void_t.
- using idx = std::size_t

  Non-negative integer index.
- using bigint = long long int

  Big integer.
- using cplx = std::complex< double >

  Complex number in double precision.
- using ket = Eigen::VectorXcd

  Complex (double precision) dynamic Eigen column vector.
- using bra = Eigen::RowVectorXcd

  Complex (double precision) dynamic Eigen row vector.
- using cmat = Eigen::MatrixXcd

  Complex (double precision) dynamic Eigen matrix.
- using dmat = Eigen::MatrixXd

  Real (double precision) dynamic Eigen matrix.
- template<typename Scalar >
  using dyn_mat = Eigen::Matrix< Scalar, Eigen::Dynamic, Eigen::Dynamic >

  Dynamic Eigen matrix over the field specified by Scalar.
- template<typename Scalar >
  using dyn_col_vect = Eigen::Matrix< Scalar, Eigen::Dynamic, 1 >

  Dynamic Eigen column vector over the field specified by Scalar.
- template<typename Scalar >
  using dyn_row_vect = Eigen::Matrix< Scalar, 1, Eigen::Dynamic >

  Dynamic Eigen row vector over the field specified by Scalar.

Functions

- constexpr cplx operator"" _i (unsigned long long int x) noexcept

  User-defined literal for complex $i = \sqrt{-1}$ (integer overload)
- constexpr cplx operator"" _i (long double x) noexcept

  User-defined literal for complex $i = \sqrt{-1}$ (real overload)
- cplx omega (idx D)

  D-th root of unity.
- template<typename Derived >
  dyn_col_vect< double > schmidtcoeffs (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Schmidt coefficients of the bi-partite pure state A.
- template<typename Derived >
  dyn_col_vect< double > schmidtcoeffs (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Schmidt coefficients of the bi-partite pure state A.
- template<typename Derived >
  cmat schmidtA (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Schmidt basis on Alice side.
- template<typename Derived >
  cmat schmidtA (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Schmidt basis on Alice side.
- template<typename Derived >
  cmat schmidtB (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Schmidt basis on Bob side.
- template<typename Derived >
  cmat schmidtB (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Schmidt basis on Bob side.
- template<typename Derived >
  std::vector< double > schmidtprobs (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Schmidt probabilities of the bi-partite pure state A.
- template<typename Derived >
  std::vector< double > schmidtprobs (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Schmidt probabilities of the bi-partite pure state A.
- template<typename Derived >
  double entanglement (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Entanglement of the bi-partite pure state A.
- template<typename Derived >
  double entanglement (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Entanglement of the bi-partite pure state A.
- template<typename Derived >
  double gconcurrence (const Eigen::MatrixBase< Derived > &A)

  G-concurrence of the bi-partite pure state A.
- template<typename Derived >
  double negativity (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Negativity of the bi-partite mixed state A.
- template<typename Derived >
  double negativity (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Negativity of the bi-partite mixed state A.
- template<typename Derived >
  double lognegativity (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Logarithmic negativity of the bi-partite mixed state A.

- template< typename Derived >
  double lognegativity (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Logarithmic negativity of the bi-partite mixed state A.
- template< typename Derived >
  double concurrence (const Eigen::MatrixBase< Derived > &A)

  Wootters concurrence of the bi-partite qubit mixed state A.
- template< typename Derived >
  double entropy (const Eigen::MatrixBase< Derived > &A)

  von-Neumann entropy of the density matrix A
- double entropy (const std::vector< double > &prob)

  Shannon entropy of the probability distribution prob.
- template< typename Derived >
  double renyi (const Eigen::MatrixBase< Derived > &A, double alpha)

  Renyi- α *entropy of the density matrix A, for* $\alpha \geq 0$.
- double renyi (const std::vector< double > &prob, double alpha)

  Renyi- α *entropy of the probability distribution prob, for* $\alpha \geq 0$.
- template< typename Derived >
  double tsallis (const Eigen::MatrixBase< Derived > &A, double q)

  Tsallis- q *entropy of the density matrix A, for* $q \geq 0$.
- double tsallis (const std::vector< double > &prob, double q)

  Tsallis- q *entropy of the probability distribution prob, for* $q \geq 0$.
- template< typename Derived >
  double qmutualinfo (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsysA, const std::vector< idx > &subsysB, const std::vector< idx > &dims)

  Quantum mutual information between 2 subsystems of a composite system.
- template< typename Derived >
  double qmutualinfo (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsysA, const std::vector< idx > &subsysB, idx d=2)

  Quantum mutual information between 2 subsystems of a composite system.
- template< typename Derived >
  internal::IOManipEigen disp (const Eigen::MatrixBase< Derived > &A, double chop=qpp::chop)

  Eigen expression ostream manipulator.
- internal::IOManipEigen disp (cplx z, double chop=qpp::chop)

  Complex number ostream manipulator.
- template< typename InputIterator >
  internal::IOManipRange< InputIterator > disp (InputIterator first, InputIterator last, const std::string &separator, const std::string &start="[", const std::string &end="]")

  Range ostream manipulator.
- template< typename Container >
  internal::IOManipRange< typename Container::const_iterator > disp (const Container &c, const std::string &separator, const std::string &start="[", const std::string &end="]", typename std::enable_if< is_iterable< Container >::value >::type ∗=nullptr)

  Standard container ostream manipulator. The container must support std::begin(), std::end() and forward iteration.
- template< typename PointerType >
  internal::IOManipPointer< PointerType > disp (const PointerType ∗p, idx N, const std::string &separator, const std::string &start="[", const std::string &end="]")

  C-style pointer ostream manipulator.
- template< typename Derived >
  void save (const Eigen::MatrixBase< Derived > &A, const std::string &fname)

  Saves Eigen expression to a binary file (internal format) in double precision.
- template< typename Derived >
  dyn_mat< typename Derived::Scalar > load (const std::string &fname)

  Loads Eigen matrix from a binary file (internal format) in double precision.

- template<typename Derived >
  dyn_col_vect< typename Derived::Scalar > ip (const Eigen::MatrixBase< Derived > &phi, const Eigen::↩ MatrixBase< Derived > &psi, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Generalized inner product.
- template<typename Derived >
  dyn_col_vect< typename Derived::Scalar > ip (const Eigen::MatrixBase< Derived > &phi, const Eigen::↩ MatrixBase< Derived > &psi, const std::vector< idx > &subsys, idx d=2)

  Generalized inner product.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks)

  Measures the state A using the set of Kraus operators Ks.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const std::initializer_list< cmat > &Ks)

  Measures the state A using the set of Kraus operators Ks.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const cmat &U)

  Measures the state A in the orthonormal basis specified by the unitary matrix U.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Measures the part subsys of the multi-partite state vector or density matrix A using the set of Kraus operators Ks.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const std::initializer_list< cmat > &Ks, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Measures the part subsys of the multi-partite state vector or density matrix A using the set of Kraus operators Ks.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks, const std::vector< idx > &subsys, idx d=2)

  Measures the part subsys of the multi-partite state vector or density matrix A using the set of Kraus operators Ks.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const std::initializer_list< cmat > &Ks, const std::vector< idx > &subsys, idx d=2)

  Measures the part subsys of the multi-partite state vector or density matrix A using the set of Kraus operators Ks.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const cmat &V, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Measures the part subsys of the multi-partite state vector or density matrix A in the orthonormal basis or rank-1 POVM specified by the matrix V.
- template<typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > measure (const Eigen::MatrixBase< Derived > &A, const cmat &V, const std::vector< idx > &subsys, idx d=2)

  Measures the part subsys of the multi-partite state vector or density matrix A in the orthonormal basis or rank-1 POVM specified by the matrix V.
- template<typename Derived >
  std::tuple< std::vector< idx >, double, cmat > measure_seq (const Eigen::MatrixBase< Derived > &A, std::vector< idx > subsys, std::vector< idx > dims)

  Sequentially measures the part subsys of the multi-partite state vector or density matrix A in the computational basis.
- template<typename Derived >
  std::tuple< std::vector< idx >, double, cmat > measure_seq (const Eigen::MatrixBase< Derived > &A, std::vector< idx > subsys, idx d=2)

  Sequentially measures the part subsys of the multi-partite state vector or density matrix A in the computational basis.

- template<typename Derived >
std::enable_if< std::is_same< typename Derived::Scalar, cplx >::value, dyn_mat< cplx > >::type loadMATLAB (const std::string &mat_file, const std::string &var_name)

  Loads a complex Eigen dynamic matrix from a MATLAB .mat file,.
- template<typename Derived >
std::enable_if<!std::is_same< typename Derived::Scalar, cplx >::value, dyn_mat< typename Derived::Scalar > >::type loadMATLAB (const std::string &mat_file, const std::string &var_name)

  Loads a non-complex Eigen dynamic matrix from a MATLAB .mat file,.
- template<typename Derived >
std::enable_if< std::is_same< typename Derived::Scalar, cplx >::value >::type saveMATLAB (const Eigen::MatrixBase< Derived > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)

  Saves a complex Eigen dynamic matrix to a MATLAB .mat file,.
- template<typename Derived >
std::enable_if< !std::is_same< typename Derived::Scalar, cplx >::value >::type saveMATLAB (const Eigen::MatrixBase< Derived > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)

  Saves a non-complex Eigen dynamic matrix to a MATLAB .mat file,.
- std::vector< int > x2contfrac (double x, idx N, idx cut=1e5)

  Simple continued fraction expansion.
- double contfrac2x (const std::vector< int > &cf, idx N=idx(-1))

  Real representation of a simple continued fraction.
- bigint gcd (bigint a, bigint b)

  Greatest common divisor of two integers.
- bigint gcd (const std::vector< bigint > &as)

  Greatest common divisor of a list of integers.
- bigint lcm (bigint a, bigint b)

  Least common multiple of two integers.
- bigint lcm (const std::vector< bigint > &as)

  Least common multiple of a list of integers.
- std::vector< idx > invperm (const std::vector< idx > &perm)

  Inverse permutation.
- std::vector< idx > compperm (const std::vector< idx > &perm, const std::vector< idx > &sigma)

  Compose permutations.
- std::vector< bigint > factors (bigint a)

  Prime factor decomposition.
- bigint modmul (bigint a, bigint b, bigint p)

  Modular multiplication without overflow.
- bigint modpow (bigint a, bigint n, bigint p)

  Fast integer power modulo p based on the SQUARE-AND-MULTIPLY algorithm.
- std::tuple< bigint, bigint, bigint > egcd (bigint a, bigint b)

  Extended greatest common divisor of two integers.
- bigint modinv (bigint a, bigint p)

  Modular inverse of a mod p.
- bool isprime (bigint p, idx k=80)

  Primality test based on the Miller-Rabin's algorithm.
- bigint randprime (bigint a, bigint b, idx N=1000)

  Generates a random big prime uniformly distributed in the interval [a, b].
- template<typename Derived1 , typename Derived2 >
dyn_mat< typename Derived1::Scalar > applyCTRL (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< Derived2 > &A, const std::vector< idx > &ctrl, const std::vector< idx > &subsys, const std::vector< idx > &dims)

Applies the controlled-gate A to the part subsys of the multi-partite state vector or density matrix state.

- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > applyCTRL (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< Derived2 > &A, const std::vector< idx > &ctrl, const std::vector< idx > &subsys, idx d=2)

  Applies the controlled-gate A to the part subsys of the multi-partite state vector or density matrix state.
- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > apply (const Eigen::MatrixBase< Derived1 > &state, const Eigen↩ ::MatrixBase< Derived2 > &A, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Applies the gate A to the part subsys of the multi-partite state vector or density matrix state.
- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > apply (const Eigen::MatrixBase< Derived1 > &state, const Eigen↩ ::MatrixBase< Derived2 > &A, const std::vector< idx > &subsys, idx d=2)

  Applies the gate A to the part subsys of the multi-partite state vector or density matrix state.
- template<typename Derived >
  cmat apply (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks)

  Applies the channel specified by the set of Kraus operators Ks to the density matrix A.
- template<typename Derived >
  cmat apply (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Applies the channel specified by the set of Kraus operators Ks to the part subsys of the multi-partite density matrix A.
- template<typename Derived >
  cmat apply (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks, const std::vector< idx > &subsys, idx d=2)

  Applies the channel specified by the set of Kraus operators Ks to the part subsys of the multi-partite density matrix A.
- cmat kraus2super (const std::vector< cmat > &Ks)

  Superoperator matrix.
- cmat kraus2choi (const std::vector< cmat > &Ks)

  Choi matrix.
- std::vector< cmat > choi2kraus (const cmat &A)

  Orthogonal Kraus operators from Choi matrix.
- cmat choi2super (const cmat &A)

  Converts Choi matrix to superoperator matrix.
- cmat super2choi (const cmat &A)

  Converts superoperator matrix to Choi matrix.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > ptrace1 (const Eigen::MatrixBase< Derived > &A, const std↩ ::vector< idx > &dims)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > ptrace1 (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > ptrace2 (const Eigen::MatrixBase< Derived > &A, const std↩ ::vector< idx > &dims)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > ptrace2 (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > ptrace (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Partial trace.

- template<typename Derived >
  dyn_mat< typename Derived::Scalar > ptrace (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsys, idx d=2)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > ptranspose (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Partial transpose.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > ptranspose (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsys, idx d=2)

  Partial transpose.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > syspermute (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &perm, const std::vector< idx > &dims)

  Subsystem permutation.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > syspermute (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &perm, idx d=2)

  Subsystem permutation.
- double rand (double a, double b)

  Generates a random real number uniformly distributed in the interval [a, b)
- bigint rand (bigint a, bigint b)

  Generates a random big integer uniformly distributed in the interval [a, b].
- idx randidx (idx a=std::numeric_limits< idx >::min(), idx b=std::numeric_limits< idx >::max())

  Generates a random index (idx) uniformly distributed in the interval [a, b].
- template<typename Derived >
  Derived rand (idx rows, idx cols, double a=0, double b=1)

  Generates a random matrix with entries uniformly distributed in the interval [a, b)
- template<>
  dmat rand (idx rows, idx cols, double a, double b)

  Generates a random real matrix with entries uniformly distributed in the interval [a, b), specialization for double matrices (qpp::dmat)
- template<>
  cmat rand (idx rows, idx cols, double a, double b)

  Generates a random complex matrix with entries (both real and imaginary) uniformly distributed in the interval [a, b), specialization for complex matrices (qpp::cmat)
- template<typename Derived >
  Derived randn (idx rows, idx cols, double mean=0, double sigma=1)

  Generates a random matrix with entries normally distributed in N(mean, sigma)
- template<>
  dmat randn (idx rows, idx cols, double mean, double sigma)

  Generates a random real matrix with entries normally distributed in N(mean, sigma), specialization for double matrices (qpp::dmat)
- template<>
  cmat randn (idx rows, idx cols, double mean, double sigma)

  Generates a random complex matrix with entries (both real and imaginary) normally distributed in N(mean, sigma), specialization for complex matrices (qpp::cmat)
- double randn (double mean=0, double sigma=1)

  Generates a random real number (double) normally distributed in N(mean, sigma)
- cmat randU (idx D=2)

  Generates a random unitary matrix.
- cmat randV (idx Din, idx Dout)

Generates a random isometry matrix.
- std::vector< cmat > randkraus (idx N, idx D=2)

    Generates a set of random Kraus operators.
- cmat randH (idx D=2)

    Generates a random Hermitian matrix.
- ket randket (idx D=2)

    Generates a random normalized ket (pure state vector)
- cmat randrho (idx D=2)

    Generates a random density matrix.
- std::vector< idx > randperm (idx N)

    Generates a random uniformly distributed permutation.
- std::vector< double > randprob (idx N)

    Generates a random probability vector uniformly distributed over the probability simplex.
- std::vector< double > uniform (idx N)

    Uniform probability distribution vector.
- std::vector< double > marginalX (const dmat &probXY)

    Marginal distribution.
- std::vector< double > marginalY (const dmat &probXY)

    Marginal distribution.
- template<typename Container >
  double avg (const std::vector< double > &prob, const Container &X, typename std::enable_if< is_iterable< Container >::value >::type *=nullptr)

    Average.
- template<typename Container >
  double cov (const dmat &probXY, const Container &X, const Container &Y, typename std::enable_if< is_iterable< Container >::value >::type *=nullptr)

    Covariance.
- template<typename Container >
  double var (const std::vector< double > &prob, const Container &X, typename std::enable_if< is_iterable< Container >::value >::type *=nullptr)

    Variance.
- template<typename Container >
  double sigma (const std::vector< double > &prob, const Container &X, typename std::enable_if< is_iterable< Container >::value >::type *=nullptr)

    Standard deviation.
- template<typename Container >
  double cor (const dmat &probXY, const Container &X, const Container &Y, typename std::enable_if< is_iterable< Container >::value >::type *=nullptr)

    Correlation.

Variables

- constexpr double chop = 1e-10

    Used in qpp::disp() for setting to zero numbers that have their absolute value smaller than qpp::chop.
- constexpr double eps = 1e-12

    Used to decide whether a number or expression in double precision is zero or not.
- constexpr idx maxn = 64

    Maximum number of allowed qubits/qudits (subsystems)
- constexpr double pi = 3.141592653589793238462643383279502884

    π
- constexpr double ee = 2.718281828459045235360287471352662497

    Base of natural logarithm, e.
- constexpr double infty = std::numeric_limits<double>::max()

    Used to denote infinity in double precision.

6.1.1 Detailed Description

Quantum++ main namespace.

6.1.2 Typedef Documentation

6.1.2.1 bigint

`using qpp::bigint = typedef long long int`

Big integer.

6.1.2.2 bra

`using qpp::bra = typedef Eigen::RowVectorXcd`

Complex (double precision) dynamic Eigen row vector.

6.1.2.3 cmat

`using qpp::cmat = typedef Eigen::MatrixXcd`

Complex (double precision) dynamic Eigen matrix.

6.1.2.4 cplx

`using qpp::cplx = typedef std::complex<double>`

Complex number in double precision.

6.1.2.5 dmat

`using qpp::dmat = typedef Eigen::MatrixXd`

Real (double precision) dynamic Eigen matrix.

6.1.2.6 dyn_col_vect

```
template<typename Scalar >
using qpp::dyn_col_vect = typedef Eigen::Matrix<Scalar, Eigen::Dynamic, 1>
```

Dynamic Eigen column vector over the field specified by *Scalar*.

Example:

```
// type of colvect is Eigen::Matrix<float, Eigen::Dynamic, 1>
dyn_col_vect<float> colvect(2);
```

6.1.2.7 dyn_mat

```
template<typename Scalar >
using qpp::dyn_mat = typedef Eigen::Matrix<Scalar, Eigen::Dynamic, Eigen::Dynamic>
```

Dynamic Eigen matrix over the field specified by *Scalar*.

Example:

```
// type of mat is Eigen::Matrix<float, Eigen::Dynamic, Eigen::Dynamic>
dyn_mat<float> mat(2, 3);
```

6.1.2.8 dyn_row_vect

```
template<typename Scalar >
using qpp::dyn_row_vect = typedef Eigen::Matrix<Scalar, 1, Eigen::Dynamic>
```

Dynamic Eigen row vector over the field specified by *Scalar*.

Example:

```
// type of rowvect is Eigen::Matrix<float, 1, Eigen::Dynamic>
dyn_row_vect<float> rowvect(3);
```

6.1.2.9 idx

```
using qpp::idx = typedef std::size_t
```

Non-negative integer index.

6.1.2.10 ket

```
using qpp::ket = typedef Eigen::VectorXcd
```

Complex (double precision) dynamic Eigen column vector.

6.1.2.11 to_void

```
template<typename... Ts>
using qpp::to_void = typedef typename make_void<Ts...>::type
```

Alias template that implements the proposal for void_t.

See also

http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2014/n3911

6.1.3 Function Documentation

6.1.3.1 apply() [1/5]

```
template<typename Derived1 , typename Derived2 >
dyn_mat<typename Derived1::Scalar> qpp::apply (
             const Eigen::MatrixBase< Derived1 > & state,
             const Eigen::MatrixBase< Derived2 > & A,
             const std::vector< idx > & subsys,
             const std::vector< idx > & dims )
```

Applies the gate *A* to the part *subsys* of the multi-partite state vector or density matrix *state*.

Note

The dimension of the gate *A* must match the dimension of *subsys*

Parameters

state	Eigen expression
A	Eigen expression
subsys	Subsystem indexes where the gate *A* is applied
dims	Dimensions of the multi-partite system

Returns

Gate *A* applied to the part *subsys* of *state*

6.1.3.2 apply() [2/5]

```
template<typename Derived1 , typename Derived2 >
dyn_mat<typename Derived1::Scalar> qpp::apply (
            const Eigen::MatrixBase< Derived1 > & state,
            const Eigen::MatrixBase< Derived2 > & A,
            const std::vector< idx > & subsys,
            idx d = 2 )
```

Applies the gate *A* to the part *subsys* of the multi-partite state vector or density matrix *state*.

Note

The dimension of the gate *A* must match the dimension of *subsys*

Parameters

state	Eigen expression
A	Eigen expression
subsys	Subsystem indexes where the gate *A* is applied
d	Subsystem dimensions

Returns

Gate *A* applied to the part *subsys* of *state*

6.1.3.3 apply() [3/5]

```
template<typename Derived >
cmat qpp::apply (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< cmat > & Ks )
```

Applies the channel specified by the set of Kraus operators *Ks* to the density matrix *A*.

Parameters

A	Eigen expression
Ks	Set of Kraus operators

Returns

Output density matrix after the action of the channel

6.1.3.4 apply() [4/5]

```
template<typename Derived >
cmat qpp::apply (
             const Eigen::MatrixBase< Derived > & A,
             const std::vector< cmat > & Ks,
             const std::vector< idx > & subsys,
             const std::vector< idx > & dims )
```

Applies the channel specified by the set of Kraus operators *Ks* to the part *subsys* of the multi-partite density matrix *A*.

Parameters

A	Eigen expression
Ks	Set of Kraus operators
subsys	Subsystem indexes where the Kraus operators *Ks* are applied
dims	Dimensions of the multi-partite system

Returns

Output density matrix after the action of the channel

6.1.3.5 apply() [5/5]

```
template<typename Derived >
cmat qpp::apply (
             const Eigen::MatrixBase< Derived > & A,
             const std::vector< cmat > & Ks,
             const std::vector< idx > & subsys,
             idx d = 2 )
```

Applies the channel specified by the set of Kraus operators *Ks* to the part *subsys* of the multi-partite density matrix *A*.

Parameters

A	Eigen expression
Ks	Set of Kraus operators
subsys	Subsystem indexes where the Kraus operators *Ks* are applied
d	Subsystem dimensions

Returns

Output density matrix after the action of the channel

6.1.3.6 applyCTRL() [1/2]

```
template<typename Derived1 , typename Derived2 >
dyn_mat<typename Derived1::Scalar> qpp::applyCTRL (
            const Eigen::MatrixBase< Derived1 > & state,
            const Eigen::MatrixBase< Derived2 > & A,
            const std::vector< idx > & ctrl,
            const std::vector< idx > & subsys,
            const std::vector< idx > & dims )
```

Applies the controlled-gate *A* to the part *subsys* of the multi-partite state vector or density matrix *state*.

See also

qpp::Gates::CTRL()

Note

The dimension of the gate *A* must match the dimension of *subsys*. Also, all control subsystems in *ctrl* must have the same dimension.

Parameters

state	Eigen expression
A	Eigen expression
ctrl	Control subsystem indexes
subsys	Subsystem indexes where the gate *A* is applied
dims	Dimensions of the multi-partite system

Returns

CTRL-A gate applied to the part *subsys* of *state*

6.1.3.7 applyCTRL() [2/2]

```
template<typename Derived1 , typename Derived2 >
dyn_mat<typename Derived1::Scalar> qpp::applyCTRL (
            const Eigen::MatrixBase< Derived1 > & state,
            const Eigen::MatrixBase< Derived2 > & A,
            const std::vector< idx > & ctrl,
            const std::vector< idx > & subsys,
            idx d = 2 )
```

Applies the controlled-gate *A* to the part *subsys* of the multi-partite state vector or density matrix *state*.

See also

qpp::Gates::CTRL()

Note

The dimension of the gate *A* must match the dimension of *subsys*

Parameters

state	Eigen expression
A	Eigen expression
ctrl	Control subsystem indexes
subsys	Subsystem indexes where the gate *A* is applied
d	Subsystem dimensions

Returns

CTRL-A gate applied to the part *subsys* of *state*

6.1.3.8 avg()

```
template<typename Container >
double qpp::avg (
             const std::vector< double > & prob,
             const Container & X,
             typename std::enable_if< is_iterable< Container >::value >::type *  = nullptr )
```

Average.

Parameters

prob	Real probability vector representing the probability distribution of *X*
X	Real random variable values represented by an STL-like container

Returns

Average of *X*

6.1.3.9 choi2kraus()

```
std::vector<cmat> qpp::choi2kraus (
             const cmat & A )  [inline]
```

Orthogonal Kraus operators from Choi matrix.

See also

qpp::kraus2choi()

Extracts a set of orthogonal (under Hilbert-Schmidt operator norm) Kraus operators from the Choi matrix *A*

Note

The Kraus operators satisfy $Tr(K_i^\dagger K_j) = \delta_{ij}$ for all $i \neq j$

Parameters

A	Choi matrix

Returns

Set of orthogonal Kraus operators

6.1.3.10 choi2super()

```
cmat qpp::choi2super (
            const cmat & A )  [inline]
```

Converts Choi matrix to superoperator matrix.

See also

qpp::super2choi()

Parameters

A	Choi matrix

Returns

Superoperator matrix

6.1.3.11 compperm()

```
std::vector<idx> qpp::compperm (
            const std::vector< idx > & perm,
            const std::vector< idx > & sigma )  [inline]
```

Compose permutations.

Parameters

perm	Permutation
sigma	Permutation

Returns

Composition of the permutations $perm \circ sigma$ = perm(sigma)

6.1.3.12 concurrence()

```
template<typename Derived >
double qpp::concurrence (
             const Eigen::MatrixBase< Derived > & A )
```

Wootters concurrence of the bi-partite qubit mixed state *A*.

Parameters

A	Eigen expression

Returns

Wootters concurrence

6.1.3.13 contfrac2x()

```
double qpp::contfrac2x (
             const std::vector< int > & cf,
             idx N = idx(-1) )  [inline]
```

Real representation of a simple continued fraction.

See also

qpp::x2contfrac()

Note

If *N* is greater than the size of *cf* (by default it is), then all terms in *cf* are considered.

Parameters

cf	Integer vector containing the simple continued fraction expansion
N	Number of terms considered in the continued fraction expansion.

Returns

Real representation of the simple continued fraction

6.1.3.14 cor()

```
template<typename Container >
double qpp::cor (
             const dmat & probXY,
             const Container & X,
             const Container & Y,
             typename std::enable_if< is_iterable< Container >::value >::type *  = nullptr )
```

Correlation.

Parameters

probXY	Real matrix representing the joint probability distribution of *X* and *Y* in lexicographical order (*X* labels the rows, *Y* labels the columns)
X	Real random variable values represented by an STL-like container
Y	Real random variable values represented by an STL-like container

Returns

Correlation of *X* and *Y*

6.1.3.15 cov()

```
template<typename Container >
double qpp::cov (
            const dmat & probXY,
            const Container & X,
            const Container & Y,
            typename std::enable_if< is_iterable< Container >::value >::type *  = nullptr )
```

Covariance.

Parameters

probXY	Real matrix representing the joint probability distribution of *X* and *Y* in lexicographical order (*X* labels the rows, *Y* labels the columns)
X	Real random variable values represented by an STL-like container
Y	Real random variable values represented by an STL-like container

Returns

Covariance of *X* and *Y*

6.1.3.16 disp() [1/5]

```
template<typename Derived >
internal::IOManipEigen qpp::disp (
            const Eigen::MatrixBase< Derived > & A,
            double chop = qpp::chop )
```

Eigen expression ostream manipulator.

Parameters

A	Eigen expression
chop	Set to zero the elements smaller in absolute value than *chop*

Returns

Instance of qpp::internal::IOManipEigen

6.1.3.17 disp() [2/5]

```
internal::IOManipEigen qpp::disp (
             cplx z,
             double chop = qpp::chop )  [inline]
```

Complex number ostream manipulator.

Parameters

z	Complex number (or any other type implicitly cast-able to std::complex<double>)
chop	Set to zero the elements smaller in absolute value than *chop*

Returns

Instance of qpp::internal::IOManipEigen

6.1.3.18 disp() [3/5]

```
template<typename InputIterator >
internal::IOManipRange<InputIterator> qpp::disp (
             InputIterator first,
             InputIterator last,
             const std::string & separator,
             const std::string & start = "[",
             const std::string & end = "]" )
```

Range ostream manipulator.

Parameters

first	Iterator to the first element of the range
last	Iterator to the last element of the range
separator	Separator
start	Left marking
end	Right marking

Returns

Instance of qpp::internal::IOManipRange

6.1.3.19 disp() [4/5]

```
template<typename Container >
internal::IOManipRange<typename Container::const_iterator> qpp::disp (
             const Container & c,
             const std::string & separator,
             const std::string & start = "[",
             const std::string & end = "]",
             typename std::enable_if< is_iterable< Container >::value >::type *  = nullptr )
```

Standard container ostream manipulator. The container must support std::begin(), std::end() and forward iteration.

Parameters

c	Container
separator	Separator
start	Left marking
end	Right marking

Returns

Instance of qpp::internal::IOManipRange

6.1.3.20 disp() [5/5]

```
template<typename PointerType >
internal::IOManipPointer<PointerType> qpp::disp (
             const PointerType * p,
             idx N,
             const std::string & separator,
             const std::string & start = "[",
             const std::string & end = "]" )
```

C-style pointer ostream manipulator.

Parameters

p	Pointer to the first element
N	Number of elements to be displayed
separator	Separator
start	Left marking
end	Right marking

Returns

Instance of qpp::internal::IOManipPointer

6.1.3.21 egcd()

```
std::tuple<bigint, bigint, bigint> qpp::egcd (
            bigint a,
            bigint b )  [inline]
```

Extended greatest common divisor of two integers.

See also

qpp::gcd()

Parameters

a	Integer
b	Integer

Returns

Tuple of: 1. Integer m, 2. Integer n, and 3. Non-negative integer $gcd(a,b)$ such that $ma+nb = gcd(a,b)$

6.1.3.22 entanglement() [1/2]

```
template<typename Derived >
double qpp::entanglement (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & dims )
```

Entanglement of the bi-partite pure state *A*.

Defined as the von-Neumann entropy of the reduced density matrix of one of the subsystems

See also

qpp::entropy()

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Entanglement, with the logarithm in base 2

6.1.3.23 entanglement() [2/2]

```
template<typename Derived >
double qpp::entanglement (
            const Eigen::MatrixBase< Derived > & A,
            idx d = 2 )
```

Entanglement of the bi-partite pure state *A*.

Defined as the von-Neumann entropy of the reduced density matrix of one of the subsystems

See also

qpp::entropy()

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Entanglement, with the logarithm in base 2

6.1.3.24 entropy() [1/2]

```
template<typename Derived >
double qpp::entropy (
            const Eigen::MatrixBase< Derived > & A )
```

von-Neumann entropy of the density matrix *A*

Parameters

A	Eigen expression

Returns

von-Neumann entropy, with the logarithm in base 2

6.1.3.25 entropy() [2/2]

```
double qpp::entropy (
            const std::vector< double > & prob )  [inline]
```

Shannon entropy of the probability distribution *prob*.

Parameters

prob	Real probability vector

Returns

Shannon entropy, with the logarithm in base 2

6.1.3.26 factors()

```
std::vector<bigint> qpp::factors (
            bigint a )  [inline]
```

Prime factor decomposition.

Note

Runs in $\mathcal{O}(\sqrt{n})$ time complexity

Parameters

a	Integer different from 0, 1 or -1

Returns

Integer vector containing the factors

6.1.3.27 gcd() [1/2]

```
bigint qpp::gcd (
            bigint a,
            bigint b )  [inline]
```

Greatest common divisor of two integers.

See also

qpp::lcm()

Parameters

a	Integer
b	Integer

Returns

Greatest common divisor of *a* and *b*

6.1.3.28 gcd() [2/2]

```
bigint qpp::gcd (
             const std::vector< bigint > & as )  [inline]
```

Greatest common divisor of a list of integers.

See also

qpp::lcm()

Parameters

as	List of integers

Returns

Greatest common divisor of all numbers in *as*

6.1.3.29 gconcurrence()

```
template<typename Derived >
double qpp::gconcurrence (
             const Eigen::MatrixBase< Derived > & A )
```

G-concurrence of the bi-partite pure state *A*.

Note

Both local dimensions must be equal

Uses qpp::logdet() to avoid overflows

See also

qpp::logdet()

Parameters

A	Eigen expression

Returns

G-concurrence

6.1.3.30 invperm()

```
std::vector<idx> qpp::invperm (
            const std::vector< idx > & perm )  [inline]
```

Inverse permutation.

Parameters

perm	Permutation

Returns

Inverse of the permutation *perm*

6.1.3.31 ip() [1/2]

```
template<typename Derived >
dyn_col_vect<typename Derived::Scalar> qpp::ip (
            const Eigen::MatrixBase< Derived > & phi,
            const Eigen::MatrixBase< Derived > & psi,
            const std::vector< idx > & subsys,
            const std::vector< idx > & dims )
```

Generalized inner product.

Parameters

phi	Column vector Eigen expression
psi	Column vector Eigen expression
subsys	Subsystem indexes over which *phi* is defined
dims	Dimensions of the multi-partite system

Returns

Inner product $\langle \phi_{subsys} | \psi \rangle$, as a scalar or column vector over the remaining Hilbert space

6.1.3.32 ip() [2/2]

```
template<typename Derived >
dyn_col_vect<typename Derived::Scalar> qpp::ip (
             const Eigen::MatrixBase< Derived > & phi,
             const Eigen::MatrixBase< Derived > & psi,
             const std::vector< idx > & subsys,
             idx d = 2 )
```

Generalized inner product.

Parameters

phi	Column vector Eigen expression
psi	Column vector Eigen expression
subsys	Subsystem indexes over which *phi* is defined
d	Subsystem dimensions

Returns

Inner product $\langle \phi_{subsys} | \psi \rangle$, as a scalar or column vector over the remaining Hilbert space

6.1.3.33 isprime()

```
bool qpp::isprime (
             bigint p,
             idx k = 80 )  [inline]
```

Primality test based on the Miller-Rabin's algorithm.

Parameters

p	Integer different from 0, 1 or -1
k	Number of iterations. The probability of a false positive is 2^{-k}.

Returns

True if the number is (most-likely) prime, false otherwise

6.1.3.34 kraus2choi()

```
cmat qpp::kraus2choi (
             const std::vector< cmat > & Ks )  [inline]
```

Choi matrix.

See also

qpp::choi2kraus()

Constructs the Choi matrix of the channel specified by the set of Kraus operators *Ks* in the standard operator basis $\{|i\rangle\langle j|\}$ ordered in lexicographical order, i.e. $|0\rangle\langle 0|$, $|0\rangle\langle 1|$ etc.

Note

The superoperator matrix S and the Choi matrix C are related by $S_{ab,mn} = C_{ma,nb}$

Parameters

Ks	Set of Kraus operators

Returns

Choi matrix

6.1.3.35 kraus2super()

```
cmat qpp::kraus2super (
             const std::vector< cmat > & Ks )  [inline]
```

Superoperator matrix.

Constructs the superoperator matrix of the channel specified by the set of Kraus operators *Ks* in the standard operator basis $\{|i\rangle\langle j|\}$ ordered in lexicographical order, i.e. $|0\rangle\langle 0|$, $|0\rangle\langle 1|$ etc.

Parameters

Ks	Set of Kraus operators

Returns

Superoperator matrix

6.1.3.36 lcm() [1/2]

```
bigint qpp::lcm (
             bigint a,
             bigint b )  [inline]
```

Least common multiple of two integers.

See also

qpp::gcd()

Parameters

a	Integer
b	Integer

Returns

Least common multiple of *a* and *b*

6.1.3.37 lcm() [2/2]

```
bigint qpp::lcm (
            const std::vector< bigint > & as )  [inline]
```

Least common multiple of a list of integers.

See also

qpp::gcd()

Parameters

as	List of integers

Returns

Least common multiple of all numbers in *as*

6.1.3.38 load()

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::load (
            const std::string & fname )
```

Loads Eigen matrix from a binary file (internal format) in double precision.

See also

qpp::save()

The template parameter cannot be automatically deduced and must be explicitly provided, depending on the scalar field of the matrix that is being loaded.

Example:

```
// loads a previously saved Eigen dynamic complex matrix from "input.bin"
cmat mat = load<cmat>("input.bin");
```

Parameters

fname	Output file name

6.1.3.39 loadMATLAB() [1/2]

```
template<typename Derived >
std::enable_if<std::is_same<typename Derived::Scalar, cplx>::value, dyn_mat<cplx> >::type
qpp::loadMATLAB (
            const std::string & mat_file,
            const std::string & var_name )
```

Loads a complex Eigen dynamic matrix from a MATLAB .mat file,.

See also

qpp::saveMATLAB()

The template parameter cannot be automatically deduced and must be explicitly provided

Example:

```
// loads a previously saved Eigen ket
// from the MATLAB file "input.mat"
ket psi = loadMATLAB<ket>("input.mat");
```

Template Parameters

Derived	Complex Eigen type

Parameters

mat_file	MATALB .mat file
var_name	Variable name in the .mat file representing the matrix to be loaded

Returns

Eigen dynamic matrix

6.1.3.40 loadMATLAB() [2/2]

```
template<typename Derived >
std::enable_if<!std::is_same<typename Derived::Scalar, cplx>::value, dyn_mat<typename Derived↩
::Scalar> >::type qpp::loadMATLAB (
            const std::string & mat_file,
            const std::string & var_name )
```

Loads a non-complex Eigen dynamic matrix from a MATLAB .mat file,.

See also

qpp::saveMATLAB()

The template parameter cannot be automatically deduced and must be explicitly provided

Example:

```
// loads a previously saved Eigen dynamic double matrix
// from the MATLAB file "input.mat"
dmat mat = loadMATLAB<dmat>("input.mat");
```

Template Parameters

Derived	Non-complex Eigen type

Parameters

mat_file	MATALB .mat file
var_name	Variable name in the .mat file representing the matrix to be loaded

Returns

Eigen dynamic matrix

6.1.3.41 lognegativity() [1/2]

```
template<typename Derived >
double qpp::lognegativity (
             const Eigen::MatrixBase< Derived > & A,
             const std::vector< idx > & dims )
```

Logarithmic negativity of the bi-partite mixed state *A*.

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Logarithmic negativity, with the logarithm in base 2

6.1.3.42 lognegativity() [2/2]

```
template<typename Derived >
double qpp::lognegativity (
            const Eigen::MatrixBase< Derived > & A,
            idx d = 2 )
```

Logarithmic negativity of the bi-partite mixed state *A*.

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Logarithmic negativity, with the logarithm in base 2

6.1.3.43 marginalX()

```
std::vector<double> qpp::marginalX (
            const dmat & probXY )  [inline]
```

Marginal distribution.

Parameters

probXY	Real matrix representing the joint probability distribution of *X* and *Y* in lexicographical order (*X* labels the rows, *Y* labels the columns)

Returns

Real vector consisting of the marginal distribution of *X*

6.1.3.44 marginalY()

```
std::vector<double> qpp::marginalY (
            const dmat & probXY )  [inline]
```

Marginal distribution.

Parameters

probXY	Real matrix representing the joint probability distribution of *X* and *Y* in lexicographical order (*X* labels the rows, *Y* labels the columns)

Returns

Real vector consisting of the marginal distribution of *Y*

6.1.3.45 measure() [1/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< cmat > & Ks )
```

Measures the state *A* using the set of Kraus operators *Ks*.

Parameters

A	Eigen expression
Ks	Set of Kraus operators

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.46 measure() [2/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
            const Eigen::MatrixBase< Derived > & A,
            const std::initializer_list< cmat > & Ks )
```

Measures the state *A* using the set of Kraus operators *Ks*.

Parameters

A	Eigen expression
Ks	Set of Kraus operators

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.47 measure() [3/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
            const Eigen::MatrixBase< Derived > & A,
            const cmat & U )
```

Measures the state *A* in the orthonormal basis specified by the unitary matrix *U*.

Parameters

A	Eigen expression
U	Unitary matrix whose columns represent the measurement basis vectors

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.48 measure() [4/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< cmat > & Ks,
            const std::vector< idx > & subsys,
            const std::vector< idx > & dims )
```

Measures the part *subsys* of the multi-partite state vector or density matrix *A* using the set of Kraus operators *Ks*.

See also

qpp::measure_seq()

Note

The dimension of all *Ks* must match the dimension of *subsys*. The measurement is destructive, i.e. the measured subsystems are traced away.

Parameters

A	Eigen expression
Ks	Set of Kraus operators
subsys	Subsystem indexes that are measured
dims	Dimensions of the multi-partite system

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.49 measure() [5/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
              const Eigen::MatrixBase< Derived > & A,
              const std::initializer_list< cmat > & Ks,
              const std::vector< idx > & subsys,
              const std::vector< idx > & dims )
```

Measures the part *subsys* of the multi-partite state vector or density matrix *A* using the set of Kraus operators *Ks*.

See also

qpp::measure_seq()

Note

The dimension of all *Ks* must match the dimension of *subsys*. The measurement is destructive, i.e. the measured subsystems are traced away.

Parameters

A	Eigen expression
Ks	Set of Kraus operators
subsys	Subsystem indexes that are measured
dims	Dimensions of the multi-partite system

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.50 measure() [6/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
              const Eigen::MatrixBase< Derived > & A,
              const std::vector< cmat > & Ks,
              const std::vector< idx > & subsys,
              idx d = 2 )
```

Measures the part *subsys* of the multi-partite state vector or density matrix *A* using the set of Kraus operators *Ks*.

See also

qpp::measure_seq()

Note

The dimension of all *Ks* must match the dimension of *subsys*. The measurement is destructive, i.e. the measured subsystems are traced away.

Parameters

A	Eigen expression
Ks	Set of Kraus operators
subsys	Subsystem indexes that are measured
d	Subsystem dimensions

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.51 measure() [7/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
             const Eigen::MatrixBase< Derived > & A,
             const std::initializer_list< cmat > & Ks,
             const std::vector< idx > & subsys,
             idx d = 2 )
```

Measures the part *subsys* of the multi-partite state vector or density matrix *A* using the set of Kraus operators *Ks*.

See also

qpp::measure_seq()

Note

The dimension of all *Ks* must match the dimension of *subsys*. The measurement is destructive, i.e. the measured subsystems are traced away.

Parameters

A	Eigen expression
Ks	Set of Kraus operators
subsys	Subsystem indexes that are measured
d	Subsystem dimensions

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.52 measure() [8/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
            const Eigen::MatrixBase< Derived > & A,
            const cmat & V,
            const std::vector< idx > & subsys,
            const std::vector< idx > & dims )
```

Measures the part *subsys* of the multi-partite state vector or density matrix *A* in the orthonormal basis or rank-1 POVM specified by the matrix *V*.

See also

qpp::measure_seq()

Note

The dimension of *V* must match the dimension of *subsys*. The measurement is destructive, i.e. the measured subsystems are traced away.

Parameters

A	Eigen expression
V	Matrix whose columns represent the measurement basis vectors or the bra parts of the rank-1 POVM
subsys	Subsystem indexes that are measured
dims	Dimensions of the multi-partite system

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.53 measure() [9/9]

```
template<typename Derived >
std::tuple<idx, std::vector<double>, std::vector<cmat> > qpp::measure (
            const Eigen::MatrixBase< Derived > & A,
            const cmat & V,
            const std::vector< idx > & subsys,
            idx d = 2 )
```

Measures the part *subsys* of the multi-partite state vector or density matrix *A* in the orthonormal basis or rank-1 POVM specified by the matrix *V*.

See also

qpp::measure_seq()

Note

The dimension of *V* must match the dimension of *subsys*. The measurement is destructive, i.e. the measured subsystems are traced away.

Parameters

A	Eigen expression
V	Matrix whose columns represent the measurement basis vectors or the bra parts of the rank-1 POVM
subsys	Subsystem indexes that are measured
d	Subsystem dimensions

Returns

Tuple of: 1. Result of the measurement, 2. Vector of outcome probabilities, and 3. Vector of post-measurement normalized states

6.1.3.54 measure_seq() [1/2]

```
template<typename Derived >
std::tuple<std::vector<idx>, double, cmat> qpp::measure_seq (
            const Eigen::MatrixBase< Derived > & A,
            std::vector< idx > subsys,
            std::vector< idx > dims )
```

Sequentially measures the part *subsys* of the multi-partite state vector or density matrix *A* in the computational basis.

See also

qpp::measure()

Parameters

A	Eigen expression
subsys	Subsystem indexes that are measured
dims	Dimensions of the multi-partite system

Returns

Tuple of: 1. Vector of outcome results of the measurement (ordered in increasing order with respect to *subsys*, i.e. first measurement result corresponds to the subsystem with the smallest index), 2. Outcome probability, and 3. Post-measurement normalized state

6.1.3.55 measure_seq() [2/2]

```
template<typename Derived >
std::tuple<std::vector<idx>, double, cmat> qpp::measure_seq (
            const Eigen::MatrixBase< Derived > & A,
            std::vector< idx > subsys,
            idx d = 2 )
```

Sequentially measures the part *subsys* of the multi-partite state vector or density matrix *A* in the computational basis.

See also

qpp::measure()

Parameters

A	Eigen expression
subsys	Subsystem indexes that are measured
d	Subsystem dimensions

Returns

Tuple of: 1. Vector of outcome results of the measurement (ordered in increasing order with respect to *subsys*, i.e. first measurement result corresponds to the subsystem with the smallest index), 2. Outcome probability, and 3. Post-measurement normalized state

6.1.3.56 modinv()

```
bigint qpp::modinv (
            bigint a,
            bigint p )  [inline]
```

Modular inverse of *a* mod *p*.

See also

qpp::egcd()

Note

a and *p* must be co-prime

Parameters

a	Non-negative integer
p	Non-negative integer

Returns

Modular inverse $a^{-1} \mod p$

6.1.3.57 modmul()

```
bigint qpp::modmul (
            bigint a,
            bigint b,
            bigint p )  [inline]
```

Modular multiplication without overflow.

Computes $ab \bmod p$ without overflow

Parameters

a	Integer
b	Integer
p	Positive integer

Returns

$ab \bmod p$ avoiding overflow

6.1.3.58 modpow()

```
bigint qpp::modpow (
            bigint a,
            bigint n,
            bigint p )  [inline]
```

Fast integer power modulo *p* based on the SQUARE-AND-MULTIPLY algorithm.

Note

Uses qpp::modmul() that avoids overflows

Computes $a^n \bmod p$

Parameters

a	Non-negative integer
n	Non-negative integer
p	Strictly positive integer

Returns

$a^n \bmod p$

6.1.3.59 negativity() [1/2]

```
template<typename Derived >
double qpp::negativity (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & dims )
```

Negativity of the bi-partite mixed state *A*.

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Negativity

6.1.3.60 negativity() [2/2]

```
template<typename Derived >
double qpp::negativity (
            const Eigen::MatrixBase< Derived > & A,
            idx d = 2 )
```

Negativity of the bi-partite mixed state *A*.

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Negativity

6.1.3.61 omega()

```
cplx qpp::omega (
            idx D )  [inline]
```

D-th root of unity.

Parameters

D	Non-negative integer

Returns

D-th root of unity $\exp(2\pi i/D)$

6.1.3.62 operator"""" _i() [1/2]

```
constexpr cplx qpp::operator"" _i (
            unsigned long long int x )  [inline], [noexcept]
```

User-defined literal for complex $i = \sqrt{-1}$ (integer overload)

Example:

```
cplx z = 4_i; // type of z is std::complex<double>
```

6.1.3.63 operator"""" _i() [2/2]

```
constexpr cplx qpp::operator"" _i (
            long double x )  [inline], [noexcept]
```

User-defined literal for complex $i = \sqrt{-1}$ (real overload)

Example:

```
cplx z = 4.5_i; // type of z is std::complex<double>
```

6.1.3.64 ptrace() [1/2]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::ptrace (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & subsys,
            const std::vector< idx > & dims )
```

Partial trace.

See also

qpp::ptrace1(), qpp::ptrace2()

Partial trace of the multi-partite state vector or density matrix over a list of subsystems

Parameters

A	Eigen expression
subsys	Subsystem indexes
dims	Dimensions of the multi-partite system

Returns

Partial trace $Tr_{subsys}(\cdot)$ over the subsytems *subsys* in a multi-partite system, as a dynamic matrix over the same scalar field as *A*

6.1.3.65 ptrace() [2/2]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::ptrace (
             const Eigen::MatrixBase< Derived > & A,
             const std::vector< idx > & subsys,
             idx d = 2 )
```

Partial trace.

See also

qpp::ptrace1(), qpp::ptrace2()

Partial trace of the multi-partite state vector or density matrix over a list of subsystems

Parameters

A	Eigen expression
subsys	Subsystem indexes
d	Subsystem dimensions

Returns

Partial trace $Tr_{subsys}(\cdot)$ over the subsytems *subsys* in a multi-partite system, as a dynamic matrix over the same scalar field as *A*

6.1.3.66 ptrace1() [1/2]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::ptrace1 (
             const Eigen::MatrixBase< Derived > & A,
             const std::vector< idx > & dims )
```

Partial trace.

See also

qpp::ptrace2()

Partial trace over the first subsystem of bi-partite state vector or density matrix

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Partial trace $Tr_A(\cdot)$ over the first subsytem A in a bi-partite system $A \otimes B$, as a dynamic matrix over the same scalar field as *A*

6.1.3.67 ptrace1() [2/2]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::ptrace1 (
            const Eigen::MatrixBase< Derived > & A,
            idx d = 2 )
```

Partial trace.

See also

qpp::ptrace2()

Partial trace over the first subsystem of bi-partite state vector or density matrix

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Partial trace $Tr_A(\cdot)$ over the first subsytem A in a bi-partite system $A \otimes B$, as a dynamic matrix over the same scalar field as *A*

6.1.3.68 ptrace2() [1/2]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::ptrace2 (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & dims )
```

Partial trace.

See also

qpp::ptrace1()

Partial trace over the second subsystem of bi-partite state vector or density matrix

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Partial trace $Tr_B(\cdot)$ over the second subsytem B in a bi-partite system $A \otimes B$, as a dynamic matrix over the same scalar field as *A*

6.1.3.69 ptrace2() `[2/2]`

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::ptrace2 (
            const Eigen::MatrixBase< Derived > & A,
            idx d = 2 )
```

Partial trace.

See also

qpp::ptrace1()

Partial trace over the second subsystem of bi-partite state vector or density matrix

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Partial trace $Tr_B(\cdot)$ over the second subsytem B in a bi-partite system $A \otimes B$, as a dynamic matrix over the same scalar field as *A*

6.1.3.70 ptranspose() `[1/2]`

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::ptranspose (
             const Eigen::MatrixBase< Derived > & A,
             const std::vector< idx > & subsys,
             const std::vector< idx > & dims )
```

Partial transpose.

Partial transpose of the multi-partite state vector or density matrix over a list of subsystems

Parameters

A	Eigen expression
subsys	Subsystem indexes
dims	Dimensions of the multi-partite system

Returns

Partial transpose $(\cdot)^{T_{subsys}}$ over the subsytems *subsys* in a multi-partite system, as a dynamic matrix over the same scalar field as *A*

6.1.3.71 ptranspose() [2/2]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::ptranspose (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & subsys,
            idx d = 2 )
```

Partial transpose.

Partial transpose of the multi-partite state vector or density matrix over a list of subsystems

Parameters

A	Eigen expression
subsys	Subsystem indexes
d	Subsystem dimensions

Returns

Partial transpose $(\cdot)^{T_{subsys}}$ over the subsytems *subsys* in a multi-partite system, as a dynamic matrix over the same scalar field as *A*

6.1.3.72 qmutualinfo() [1/2]

```
template<typename Derived >
double qpp::qmutualinfo (
             const Eigen::MatrixBase< Derived > & A,
             const std::vector< idx > & subsysA,
             const std::vector< idx > & subsysB,
             const std::vector< idx > & dims )
```

Quantum mutual information between 2 subsystems of a composite system.

Parameters

A	Eigen expression
subsysA	Indexes of the first subsystem
subsysB	Indexes of the second subsystem
dims	Dimensions of the multi-partite system

Returns

Mutual information between the 2 subsystems

6.1.3.73 qmutualinfo() [2/2]

```
template<typename Derived >
double qpp::qmutualinfo (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & subsysA,
            const std::vector< idx > & subsysB,
            idx d = 2 )
```

Quantum mutual information between 2 subsystems of a composite system.

Parameters

A	Eigen expression
subsysA	Indexes of the first subsystem
subsysB	Indexes of the second subsystem
d	Subsystem dimensions

Returns

Mutual information between the 2 subsystems

6.1.3.74 rand() [1/5]

```
double qpp::rand (
            double a,
            double b )  [inline]
```

Generates a random real number uniformly distributed in the interval [a, b)

Parameters

a	Beginning of the interval, belongs to it
b	End of the interval, does not belong to it

Returns

Random real number (double) uniformly distributed in the interval [a, b)

6.1.3.75 rand() [2/5]

```
bigint qpp::rand (
            bigint a,
            bigint b )  [inline]
```

Generates a random big integer uniformly distributed in the interval [a, b].

Note

To avoid ambiguity with double qpp::rand(double, double) cast at least one of the arguments to qpp::bigint

Parameters

a	Beginning of the interval, belongs to it
b	End of the interval, belongs to it

Returns

Random big integer uniformly distributed in the interval [a, b]

6.1.3.76 rand() [3/5]

```
template<typename Derived >
Derived qpp::rand (
            idx rows,
            idx cols,
            double a = 0,
            double b = 1 )
```

Generates a random matrix with entries uniformly distributed in the interval [a, b)

If complex, then both real and imaginary parts are uniformly distributed in [a, b)

This is the generic version that always throws qpp::Exception::Type::UNDEFINED_TYPE. It is specialized only for qpp::dmat and qpp::cmat

6.1.3.77 rand() [4/5]

```
template<>
dmat qpp::rand (
              idx rows,
              idx cols,
              double a,
              double b )  [inline]
```

Generates a random real matrix with entries uniformly distributed in the interval [a, b), specialization for double matrices (qpp::dmat)

The template parameter cannot be automatically deduced and must be explicitly provided

Example:

```
// generates a 3 x 3 random Eigen::MatrixXd,
// with entries uniformly distributed in [-1,1)
dmat mat = rand<dmat>(3, 3, -1, 1);
```

Parameters

rows	Number of rows of the random generated matrix
cols	Number of columns of the random generated matrix
a	Beginning of the interval, belongs to it
b	End of the interval, does not belong to it

Returns

Random real matrix

6.1.3.78 rand() [5/5]

```
template<>
cmat qpp::rand (
              idx rows,
              idx cols,
              double a,
              double b )  [inline]
```

Generates a random complex matrix with entries (both real and imaginary) uniformly distributed in the interval [a, b), specialization for complex matrices (qpp::cmat)

The template parameter cannot be automatically deduced and must be explicitly provided

Example:

```
// generates a 3 x 3 random Eigen::MatrixXcd,
// with entries (both real and imaginary) uniformly distributed in [-1,1)
cmat mat = rand<cmat>(3, 3, -1, 1);
```

Parameters

rows	Number of rows of the random generated matrix
cols	Number of columns of the random generated matrix
a	Beginning of the interval, belongs to it
b	End of the interval, does not belong to it

Returns

Random complex matrix

6.1.3.79 randH()

```
cmat qpp::randH (
             idx D = 2 )  [inline]
```

Generates a random Hermitian matrix.

Parameters

D	Dimension of the Hilbert space

Returns

Random Hermitian matrix

6.1.3.80 randidx()

```
idx qpp::randidx (
             idx a = std::numeric_limits<idx>::min(),
             idx b = std::numeric_limits<idx>::max() )  [inline]
```

Generates a random index (idx) uniformly distributed in the interval [a, b].

Parameters

a	Beginning of the interval, belongs to it
b	End of the interval, belongs to it

Returns

Random index (idx) uniformly distributed in the interval [a, b]

6.1.3.81 randket()

```
ket qpp::randket (
            idx D = 2 )  [inline]
```

Generates a random normalized ket (pure state vector)

Parameters

D	Dimension of the Hilbert space

Returns

Random normalized ket

6.1.3.82 randkraus()

```
std::vector<cmat> qpp::randkraus (
            idx N,
            idx D = 2 )  [inline]
```

Generates a set of random Kraus operators.

Note

The set of Kraus operators satisfy the closure condition $\sum_i K_i^\dagger K_i = I$

Parameters

N	Number of Kraus operators
D	Dimension of the Hilbert space

Returns

Set of *N* Kraus operators satisfying the closure condition

6.1.3.83 randn() [1/4]

```
template<typename Derived >
Derived qpp::randn (
            idx rows,
            idx cols,
            double mean = 0,
            double sigma = 1 )
```

Generates a random matrix with entries normally distributed in N(mean, sigma)

If complex, then both real and imaginary parts are normally distributed in N(mean, sigma)

This is the generic version that always throws qpp::Exception::Type::UNDEFINED_TYPE. It is specialized only for qpp::dmat and qpp::cmat

6.1.3.84 randn() `[2/4]`

```
template<>
dmat qpp::randn (
            idx rows,
            idx cols,
            double mean,
            double sigma )  [inline]
```

Generates a random real matrix with entries normally distributed in N(mean, sigma), specialization for double matrices (qpp::dmat)

The template parameter cannot be automatically deduced and must be explicitly provided

Example:

```
// generates a 3 x 3 random Eigen::MatrixXd,
// with entries normally distributed in N(0,2)
dmat mat = randn<dmat>(3, 3, 0, 2);
```

Parameters

rows	Number of rows of the random generated matrix
cols	Number of columns of the random generated matrix
mean	Mean
sigma	Standard deviation

Returns

Random real matrix

6.1.3.85 randn() `[3/4]`

```
template<>
cmat qpp::randn (
            idx rows,
            idx cols,
            double mean,
            double sigma )  [inline]
```

Generates a random complex matrix with entries (both real and imaginary) normally distributed in N(mean, sigma), specialization for complex matrices (qpp::cmat)

The template parameter cannot be automatically deduced and must be explicitly provided

Example:

```
// generates a 3 x 3 random Eigen::MatrixXcd,
// with entries (both real and imaginary) normally distributed in N(0,2)
cmat mat = randn<cmat>(3, 3, 0, 2);
```

Parameters

rows	Number of rows of the random generated matrix
cols	Number of columns of the random generated matrix
mean	Mean
sigma	Standard deviation

Returns

Random complex matrix

6.1.3.86 randn() `[4/4]`

```
double qpp::randn (
            double mean = 0,
            double sigma = 1 )  [inline]
```

Generates a random real number (double) normally distributed in N(mean, sigma)

Parameters

mean	Mean
sigma	Standard deviation

Returns

Random real number normally distributed in N(mean, sigma)

6.1.3.87 randperm()

```
std::vector<idx> qpp::randperm (
            idx N )  [inline]
```

Generates a random uniformly distributed permutation.

Uses Knuth shuffle method (as implemented by std::shuffle), so that all permutations are equally probable

Parameters

N	Size of the permutation

Returns

Random permutation of size *N*

6.1.3.88 randprime()

```
bigint qpp::randprime (
            bigint a,
            bigint b,
            idx N = 1000 )  [inline]
```

Generates a random big prime uniformly distributed in the interval [a, b].

Parameters

a	Beginning of the interval, belongs to it
b	End of the interval, belongs to it
N	Maximum number of candidates

Returns

Random big integer uniformly distributed in the interval [a, b]

6.1.3.89 randprob()

```
std::vector<double> qpp::randprob (
            idx N )  [inline]
```

Generates a random probability vector uniformly distributed over the probability simplex.

Parameters

N	Size of the probability vector

Returns

Random probability vector

6.1.3.90 randrho()

```
cmat qpp::randrho (
            idx D = 2 )  [inline]
```

Generates a random density matrix.

Parameters

D	Dimension of the Hilbert space

Returns

Random density matrix

6.1.3.91 randU()

```
cmat qpp::randU (
             idx D = 2 )  [inline]
```

Generates a random unitary matrix.

Parameters

D	Dimension of the Hilbert space

Returns

Random unitary

6.1.3.92 randV()

```
cmat qpp::randV (
             idx Din,
             idx Dout )  [inline]
```

Generates a random isometry matrix.

Parameters

Din	Size of the input Hilbert space
Dout	Size of the output Hilbert space

Returns

Random isometry matrix

6.1.3.93 renyi() [1/2]

```
template<typename Derived >
double qpp::renyi (
            const Eigen::MatrixBase< Derived > & A,
            double alpha )
```

Renyi- α entropy of the density matrix *A*, for $\alpha \geq 0$.

Note

When $\alpha \to 1$ the Renyi entropy converges to the von-Neumann entropy, with the logarithm in base 2

Parameters

A	Eigen expression
alpha	Non-negative real number, use qpp::infty for $\alpha = \infty$

Returns

Renyi- α entropy, with the logarithm in base 2

6.1.3.94 renyi() [2/2]

```
double qpp::renyi (
            const std::vector< double > & prob,
            double alpha ) [inline]
```

Renyi- α entropy of the probability distribution *prob*, for $\alpha \geq 0$.

Note

When $\alpha \to 1$ the Renyi entropy converges to the Shannon entropy, with the logarithm in base 2

Parameters

prob	Real probability vector
alpha	Non-negative real number, use qpp::infty for $\alpha = \infty$

Returns

Renyi- α entropy, with the logarithm in base 2

6.1.3.95 save()

```
template<typename Derived >
void qpp::save (
            const Eigen::MatrixBase< Derived > & A,
            const std::string & fname )
```

Saves Eigen expression to a binary file (internal format) in double precision.

See also

qpp::load()

Parameters

A	Eigen expression
fname	Output file name

6.1.3.96 saveMATLAB() [1/2]

```
template<typename Derived >
std::enable_if< std::is_same<typename Derived::Scalar, cplx>::value>::type qpp::saveMATLAB (
            const Eigen::MatrixBase< Derived > & A,
            const std::string & mat_file,
            const std::string & var_name,
            const std::string & mode )
```

Saves a complex Eigen dynamic matrix to a MATLAB .mat file,.

See also

qpp::loadMATLAB()

Template Parameters

Complex	Eigen type

Parameters

A	Eigen expression over the complex field
mat_file	MATALB .mat file
var_name	Variable name in the .mat file representing the matrix to be saved
mode	Saving mode (append, overwrite etc.), see MATLAB *matOpen()* documentation for details

6.1.3.97 saveMATLAB() [2/2]

```
template<typename Derived >
std::enable_if< !std::is_same<typename Derived::Scalar, cplx>::value>::type qpp::saveMATLAB (
            const Eigen::MatrixBase< Derived > & A,
            const std::string & mat_file,
            const std::string & var_name,
            const std::string & mode )
```

Saves a non-complex Eigen dynamic matrix to a MATLAB .mat file,.

See also

qpp::loadMATLAB()

Template Parameters

Npn-complex	Eigen type

Parameters

A	Non-complex Eigen expression
mat_file	MATALB .mat file
var_name	Variable name in the .mat file representing the matrix to be saved
mode	Saving mode (append, overwrite etc.), see MATLAB *matOpen()* documentation for details

6.1.3.98 schmidtA() [1/2]

```
template<typename Derived >
cmat qpp::schmidtA (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & dims )
```

Schmidt basis on Alice side.

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Unitary matrix U whose columns represent the Schmidt basis vectors on Alice side.

6.1.3.99 schmidtA() [2/2]

```
template<typename Derived >
cmat qpp::schmidtA (
             const Eigen::MatrixBase< Derived > & A,
             idx d = 2 )
```

Schmidt basis on Alice side.

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Unitary matrix U whose columns represent the Schmidt basis vectors on Alice side.

6.1.3.100 schmidtB() [1/2]

```
template<typename Derived >
cmat qpp::schmidtB (
             const Eigen::MatrixBase< Derived > & A,
             const std::vector< idx > & dims )
```

Schmidt basis on Bob side.

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Unitary matrix V whose columns represent the Schmidt basis vectors on Bob side.

6.1.3.101 schmidtB() [2/2]

```
template<typename Derived >
cmat qpp::schmidtB (
             const Eigen::MatrixBase< Derived > & A,
             idx d = 2 )
```

Schmidt basis on Bob side.

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Unitary matrix V whose columns represent the Schmidt basis vectors on Bob side.

6.1.3.102 schmidtcoeffs() [1/2]

```
template<typename Derived >
dyn_col_vect<double> qpp::schmidtcoeffs (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & dims )
```

Schmidt coefficients of the bi-partite pure state *A*.

Note

The sum of the squares of the Schmidt coefficients equals 1

See also

qpp::schmidtprobs()

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Schmidt coefficients of *A*, ordered in decreasing order, as a real dynamic column vector

6.1.3.103 schmidtcoeffs() [2/2]

```
template<typename Derived >
dyn_col_vect<double> qpp::schmidtcoeffs (
            const Eigen::MatrixBase< Derived > & A,
            idx d = 2 )
```

Schmidt coefficients of the bi-partite pure state *A*.

Note

The sum of the squares of the Schmidt coefficients equals 1

See also

qpp::schmidtprobs()

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Schmidt coefficients of *A*, ordered in decreasing order, as a real dynamic column vector

6.1.3.104 schmidtprobs() [1/2]

```
template<typename Derived >
std::vector<double> qpp::schmidtprobs (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & dims )
```

Schmidt probabilities of the bi-partite pure state *A*.

Defined as the squares of the Schmidt coefficients. The sum of the Schmidt probabilities equals 1.

See also

qpp::schmidtcoeffs()

Parameters

A	Eigen expression
dims	Dimensions of the bi-partite system

Returns

Real vector consisting of the Schmidt probabilites of *A*, ordered in decreasing order

6.1.3.105 schmidtprobs() [2/2]

```
template<typename Derived >
std::vector<double> qpp::schmidtprobs (
             const Eigen::MatrixBase< Derived > & A,
             idx d = 2 )
```

Schmidt probabilities of the bi-partite pure state *A*.

Defined as the squares of the Schmidt coefficients. The sum of the Schmidt probabilities equals 1.

See also

qpp::schmidtcoeffs()

Parameters

A	Eigen expression
d	Subsystem dimensions

Returns

Real vector consisting of the Schmidt probabilites of *A*, ordered in decreasing order

6.1.3.106 sigma()

```
template<typename Container >
double qpp::sigma (
              const std::vector< double > & prob,
              const Container & X,
              typename std::enable_if< is_iterable< Container >::value >::type * = nullptr )
```

Standard deviation.

Parameters

prob	Real probability vector representing the probability distribution of *X*
X	Real random variable values represented by an STL-like container

Returns

Standard deviation of *X*

6.1.3.107 super2choi()

```
cmat qpp::super2choi (
              const cmat & A )  [inline]
```

Converts superoperator matrix to Choi matrix.

See also

qpp::choi2super()

Parameters

A	Superoperator matrix

Returns

Choi matrix

6.1.3.108 syspermute() [1/2]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::syspermute (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & perm,
            const std::vector< idx > & dims )
```

Subsystem permutation.

Permutes the subsystems of a state vector or density matrix. The qubit *perm*[*i*] is permuted to the location *i*.

Parameters

A	Eigen expression
perm	Permutation
dims	Dimensions of the multi-partite system

Returns

Permuted system, as a dynamic matrix over the same scalar field as *A*

6.1.3.109 syspermute() [2/2]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::syspermute (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & perm,
            idx d = 2 )
```

Subsystem permutation.

Permutes the subsystems of a state vector or density matrix. The qubit *perm*[*i*] is permuted to the location *i*.

Parameters

A	Eigen expression
perm	Permutation
d	Subsystem dimensions

Returns

Permuted system, as a dynamic matrix over the same scalar field as *A*

6.1.3.110 tsallis() `[1/2]`

```
template<typename Derived >
double qpp::tsallis (
            const Eigen::MatrixBase< Derived > & A,
            double q )
```

Tsallis- q entropy of the density matrix *A*, for $q \geq 0$.

Note

When $q \to 1$ the Tsallis entropy converges to the von-Neumann entropy, with the logarithm in base e

Parameters

A	Eigen expression
q	Non-negative real number

Returns

Tsallis- q entropy

6.1.3.111 tsallis() `[2/2]`

```
double qpp::tsallis (
            const std::vector< double > & prob,
            double q )  [inline]
```

Tsallis- q entropy of the probability distribution *prob*, for $q \geq 0$.

Note

When $q \to 1$ the Tsallis entropy converges to the Shannon entropy, with the logarithm in base e

Parameters

prob	Real probability vector
q	Non-negative real number

Returns

Tsallis- q entropy

6.1.3.112 uniform()

```
std::vector<double> qpp::uniform (
            idx N )  [inline]
```

Uniform probability distribution vector.

Parameters

N	Size of the alphabet

Returns

Real vector consisting of a uniform distribution of size *N*

6.1.3.113 var()

```
template<typename Container >
double qpp::var (
            const std::vector< double > & prob,
            const Container & X,
            typename std::enable_if< is_iterable< Container >::value >::type *  = nullptr )
```

Variance.

Parameters

prob	Real probability vector representing the probability distribution of *X*
X	Real random variable values represented by an STL-like container

Returns

Variance of *X*

6.1.3.114 x2contfrac()

```
std::vector<int> qpp::x2contfrac (
            double x,
            idx N,
            idx cut = 1e5 )  [inline]
```

Simple continued fraction expansion.

See also

qpp::contfrac2x()

Parameters

x	Real number
N	Maximum number of terms in the expansion
cut	Stop the expansion when the next term is greater than *cut*

Returns

Integer vector containing the simple continued fraction expansion of *x*. If there are *M* less than *N* terms in the expansion, a shorter vector with *M* components is returned.

6.1.4 Variable Documentation

6.1.4.1 chop

```
constexpr double qpp::chop = 1e-10
```

Used in qpp::disp() for setting to zero numbers that have their absolute value smaller than qpp::chop.

6.1.4.2 ee

```
constexpr double qpp::ee = 2.718281828459045235360287471352662497
```

Base of natural logarithm, e.

6.1.4.3 eps

```
constexpr double qpp::eps = 1e-12
```

Used to decide whether a number or expression in double precision is zero or not.

Example:

```
if(std::abs(x) < qpp::eps) // x is zero
```

6.1.4.4 infty

```
constexpr double qpp::infty = std::numeric_limits<double>::max()
```

Used to denote infinity in double precision.

6.1.4.5 maxn

```
constexpr idx qpp::maxn = 64
```

Maximum number of allowed qubits/qudits (subsystems)

Used internally to allocate arrays on the stack (for performance reasons):

6.1.4.6 pi

```
constexpr double qpp::pi = 3.141592653589793238462643383279502884
```

π

6.2 qpp::exception Namespace Reference

Quantum++ exception hierarchy namespace.

Classes

- class CustomException

  Custom exception.
- class DimsInvalid

  Invalid dimension(s) exception.
- class DimsMismatchCvector

  Dimension(s) mismatch column vector size exception.
- class DimsMismatchMatrix

  Dimension(s) mismatch matrix size exception.
- class DimsMismatchRvector

  Dimension(s) mismatch row vector size exception.
- class DimsMismatchVector

  Dimension(s) mismatch vector size exception.
- class DimsNotEqual

  Dimensions not equal exception.
- class Exception

  Base class for generating Quantum++ custom exceptions.
- class MatrixMismatchSubsys

  Matrix mismatch subsystems exception.
- class MatrixNotCvector

Matrix is not a column vector exception.
- class MatrixNotRvector

  Matrix is not a row vector exception.
- class MatrixNotSquare

  Matrix is not square exception.
- class MatrixNotSquareNorCvector

  Matrix is not square nor column vector exception.
- class MatrixNotSquareNorRvector

  Matrix is not square nor row vector exception.
- class MatrixNotSquareNorVector

  Matrix is not square nor vector exception.
- class MatrixNotVector

  Matrix is not a vector exception.
- class NoCodeword

  Codeword does not exist exception.
- class NotBipartite

  Not bi-partite exception.
- class NotQubitCvector

  Column vector is not 2 x 1 exception.
- class NotQubitMatrix

  Matrix is not 2 x 2 exception.
- class NotQubitRvector

  Row vector is not 1 x 2 exception.
- class NotQubitSubsys

  Subsystems are not qubits exception.
- class NotQubitVector

  Vector is not 2 x 1 nor 1 x 2 exception.
- class OutOfRange

  Parameter out of range exception.
- class PermInvalid

  Invalid permutation exception.
- class PermMismatchDims

  Permutation mismatch dimensions exception.
- class SizeMismatch

  Size mismatch exception.
- class SubsysMismatchDims

  Subsystems mismatch dimensions exception.
- class TypeMismatch

  Type mismatch exception.
- class UndefinedType

  Not defined for this type exception.
- class Unknown

  Unknown exception.
- class ZeroSize

  Object has zero size exception.

6.2.1 Detailed Description

Quantum++ exception hierarchy namespace.

6.3 qpp::experimental Namespace Reference

Experimental/test functions/classes, do not use or modify.

Classes

- class Bit_circuit
- class Dynamic_bitset

6.3.1 Detailed Description

Experimental/test functions/classes, do not use or modify.

6.4 qpp::internal Namespace Reference

Internal utility functions, do not use them directly or modify them.

Classes

- struct Display_Impl_
- class IOManipEigen
- class IOManipPointer
- class IOManipRange
- class Singleton

  Singleton policy class, used internally to implement the singleton pattern via CRTP (Curiously recurring template pattern)

Functions

- void n2multiidx (idx n, idx numdims, const idx ∗const dims, idx ∗result) noexcept
- idx multiidx2n (const idx ∗const midx, idx numdims, const idx ∗const dims) noexcept
- template< typename Derived >
  bool check_square_mat (const Eigen::MatrixBase< Derived > &A)
- template< typename Derived >
  bool check_vector (const Eigen::MatrixBase< Derived > &A)
- template< typename Derived >
  bool check_rvector (const Eigen::MatrixBase< Derived > &A)
- template< typename Derived >
  bool check_cvector (const Eigen::MatrixBase< Derived > &A)
- template< typename T >
  bool check_nonzero_size (const T &x) noexcept
- template< typename T1 , typename T2 >
  bool check_matching_sizes (const T1 &lhs, const T2 &rhs) noexcept
- bool check_dims (const std::vector< idx > &dims)
- template< typename Derived >
  bool check_dims_match_mat (const std::vector< idx > &dims, const Eigen::MatrixBase< Derived > &A)
- template< typename Derived >
  bool check_dims_match_cvect (const std::vector< idx > &dims, const Eigen::MatrixBase< Derived > &A)

- template<typename Derived >
  bool check_dims_match_rvect (const std::vector< idx > &dims, const Eigen::MatrixBase< Derived > &A)
- bool check_eq_dims (const std::vector< idx > &dims, idx dim) noexcept
- bool check_subsys_match_dims (const std::vector< idx > &subsys, const std::vector< idx > &dims)
- template<typename Derived >
  bool check_qubit_matrix (const Eigen::MatrixBase< Derived > &A) noexcept
- template<typename Derived >
  bool check_qubit_cvector (const Eigen::MatrixBase< Derived > &A) noexcept
- template<typename Derived >
  bool check_qubit_rvector (const Eigen::MatrixBase< Derived > &A) noexcept
- template<typename Derived >
  bool check_qubit_vector (const Eigen::MatrixBase< Derived > &A) noexcept
- bool check_perm (const std::vector< idx > &perm)
- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > kron2 (const Eigen::MatrixBase< Derived1 > &A, const Eigen::↩
  MatrixBase< Derived2 > &B)
- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > dirsum2 (const Eigen::MatrixBase< Derived1 > &A, const Eigen↩
  ::MatrixBase< Derived2 > &B)
- template<typename T >
  void variadic_vector_emplace (std::vector< T > &)
- template<typename T , typename First , typename... Args>
  void variadic_vector_emplace (std::vector< T > &v, First &&first, Args &&... args)
- idx get_num_subsys (idx sz, idx d)
- idx get_dim_subsys (idx sz, idx N)

6.4.1 Detailed Description

Internal utility functions, do not use them directly or modify them.

6.4.2 Function Documentation

6.4.2.1 check_cvector()

```
template<typename Derived >
bool qpp::internal::check_cvector (
            const Eigen::MatrixBase< Derived > & A )
```

6.4.2.2 check_dims()

```
bool qpp::internal::check_dims (
            const std::vector< idx > & dims )  [inline]
```

6.4.2.3 check_dims_match_cvect()

```
template<typename Derived >
bool qpp::internal::check_dims_match_cvect (
            const std::vector< idx > & dims,
            const Eigen::MatrixBase< Derived > & A )
```

6.4.2.4 check_dims_match_mat()

```
template<typename Derived >
bool qpp::internal::check_dims_match_mat (
            const std::vector< idx > & dims,
            const Eigen::MatrixBase< Derived > & A )
```

6.4.2.5 check_dims_match_rvect()

```
template<typename Derived >
bool qpp::internal::check_dims_match_rvect (
            const std::vector< idx > & dims,
            const Eigen::MatrixBase< Derived > & A )
```

6.4.2.6 check_eq_dims()

```
bool qpp::internal::check_eq_dims (
            const std::vector< idx > & dims,
            idx dim )  [inline], [noexcept]
```

6.4.2.7 check_matching_sizes()

```
template<typename T1 , typename T2 >
bool qpp::internal::check_matching_sizes (
            const T1 & lhs,
            const T2 & rhs )  [noexcept]
```

6.4.2.8 check_nonzero_size()

```
template<typename T >
bool qpp::internal::check_nonzero_size (
            const T & x )  [noexcept]
```

6.4.2.9 check_perm()

```
bool qpp::internal::check_perm (
            const std::vector< idx > & perm )  [inline]
```

6.4.2.10 check_qubit_cvector()

```
template<typename Derived >
bool qpp::internal::check_qubit_cvector (
            const Eigen::MatrixBase< Derived > & A )  [noexcept]
```

6.4.2.11 check_qubit_matrix()

```
template<typename Derived >
bool qpp::internal::check_qubit_matrix (
            const Eigen::MatrixBase< Derived > & A )  [noexcept]
```

6.4.2.12 check_qubit_rvector()

```
template<typename Derived >
bool qpp::internal::check_qubit_rvector (
            const Eigen::MatrixBase< Derived > & A )  [noexcept]
```

6.4.2.13 check_qubit_vector()

```
template<typename Derived >
bool qpp::internal::check_qubit_vector (
            const Eigen::MatrixBase< Derived > & A )  [noexcept]
```

6.4.2.14 check_rvector()

```
template<typename Derived >
bool qpp::internal::check_rvector (
            const Eigen::MatrixBase< Derived > & A )
```

6.4.2.15 check_square_mat()

```
template<typename Derived >
bool qpp::internal::check_square_mat (
             const Eigen::MatrixBase< Derived > & A )
```

6.4.2.16 check_subsys_match_dims()

```
bool qpp::internal::check_subsys_match_dims (
             const std::vector< idx > & subsys,
             const std::vector< idx > & dims )  [inline]
```

6.4.2.17 check_vector()

```
template<typename Derived >
bool qpp::internal::check_vector (
             const Eigen::MatrixBase< Derived > & A )
```

6.4.2.18 dirsum2()

```
template<typename Derived1 , typename Derived2 >
dyn_mat<typename Derived1::Scalar> qpp::internal::dirsum2 (
             const Eigen::MatrixBase< Derived1 > & A,
             const Eigen::MatrixBase< Derived2 > & B )
```

6.4.2.19 get_dim_subsys()

```
idx qpp::internal::get_dim_subsys (
             idx sz,
             idx N )  [inline]
```

6.4.2.20 get_num_subsys()

```
idx qpp::internal::get_num_subsys (
             idx sz,
             idx d )  [inline]
```

6.4.2.21 kron2()

```
template<typename Derived1 , typename Derived2 >
dyn_mat<typename Derived1::Scalar> qpp::internal::kron2 (
             const Eigen::MatrixBase< Derived1 > & A,
             const Eigen::MatrixBase< Derived2 > & B )
```

6.4.2.22 multiidx2n()

```
idx qpp::internal::multiidx2n (
             const idx *const midx,
             idx numdims,
             const idx *const dims )  [inline], [noexcept]
```

6.4.2.23 n2multiidx()

```
void qpp::internal::n2multiidx (
             idx n,
             idx numdims,
             const idx *const dims,
             idx * result )  [inline], [noexcept]
```

6.4.2.24 variadic_vector_emplace() [1/2]

```
template<typename T >
void qpp::internal::variadic_vector_emplace (
             std::vector< T > &  )
```

6.4.2.25 variadic_vector_emplace() [2/2]

```
template<typename T , typename First , typename...  Args>
void qpp::internal::variadic_vector_emplace (
             std::vector< T > & v,
             First && first,
             Args &&...  args )
```

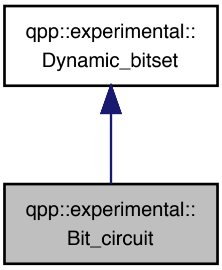
Chapter 7

Class Documentation

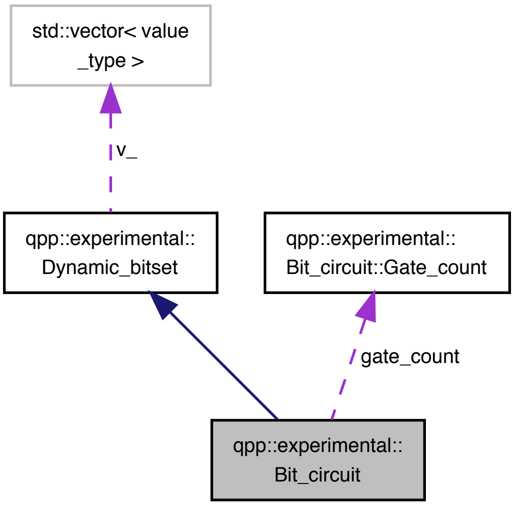
7.1 qpp::experimental::Bit_circuit Class Reference

`#include <experimental/experimental.h>`

Inheritance diagram for qpp::experimental::Bit_circuit:



Collaboration diagram for qpp::experimental::Bit_circuit:


std::vector< value _type >
v_
qpp::experimental:: Dynamic_bitset
qpp::experimental:: Bit_circuit::Gate_count
gate_count
qpp::experimental:: Bit_circuit

Classes

- struct Gate_count

Public Member Functions

- Bit_circuit & X (idx pos)
- Bit_circuit & NOT (idx pos)
- Bit_circuit & CNOT (const std::vector< idx > &pos)
- Bit_circuit & TOF (const std::vector< idx > &pos)
- Bit_circuit & SWAP (const std::vector< idx > &pos)
- Bit_circuit & FRED (const std::vector< idx > &pos)
- Bit_circuit & reset () noexcept

Public Attributes

- struct qpp::experimental::Bit_circuit::Gate_count gate_count

Additional Inherited Members

7.1.1 Member Function Documentation

7.1.1.1 CNOT()

```
Bit_circuit& qpp::experimental::Bit_circuit::CNOT (
             const std::vector< idx > & pos ) [inline]
```

7.1.1.2 FRED()

```
Bit_circuit& qpp::experimental::Bit_circuit::FRED (
             const std::vector< idx > & pos ) [inline]
```

7.1.1.3 NOT()

```
Bit_circuit& qpp::experimental::Bit_circuit::NOT (
             idx pos ) [inline]
```

7.1.1.4 reset()

```
Bit_circuit& qpp::experimental::Bit_circuit::reset ( ) [inline], [noexcept]
```

7.1.1.5 SWAP()

```
Bit_circuit& qpp::experimental::Bit_circuit::SWAP (
             const std::vector< idx > & pos ) [inline]
```

7.1.1.6 TOF()

```
Bit_circuit& qpp::experimental::Bit_circuit::TOF (
             const std::vector< idx > & pos ) [inline]
```

7.1.1.7 X()

```
Bit_circuit& qpp::experimental::Bit_circuit::X (
             idx pos ) [inline]
```

7.1.2 Member Data Documentation

7.1.2.1 gate_count

`struct qpp::experimental::Bit_circuit::Gate_count qpp::experimental::Bit_circuit::gate_count`

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

- experimental/experimental.h

7.2 qpp::Bit_circuit Class Reference

Classical reversible circuit simulator.

`#include <experimental/experimental.h>`

7.2.1 Detailed Description

Classical reversible circuit simulator.

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

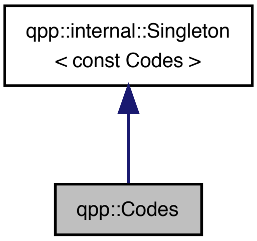
- experimental/experimental.h

7.3 qpp::Codes Class Reference

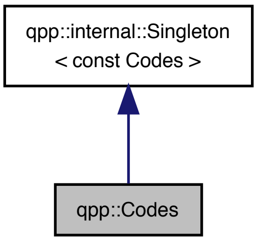
const Singleton class that defines quantum error correcting codes

`#include <classes/codes.h>`

Inheritance diagram for qpp::Codes:



Collaboration diagram for qpp::Codes:



Public Types

- enum Type { Type::FIVE_QUBIT = 1, Type::SEVEN_QUBIT_STEANE, Type::NINE_QUBIT_SHOR }

  Code types, add more codes here if needed.

Public Member Functions

- ket codeword (Type type, idx i) const

  Returns the codeword of the specified code type.

Private Member Functions

- Codes ()

  Default constructor.
- ∼Codes ()=default

  Default destructor.

Friends

- class internal::Singleton< const Codes >

Additional Inherited Members

7.3.1 Detailed Description

const Singleton class that defines quantum error correcting codes

7.3.2 Member Enumeration Documentation

7.3.2.1 Type

```
enum qpp::Codes::Type  [strong]
```

Code types, add more codes here if needed.

See also

qpp::Codes::codeword()

Enumerator

FIVE_QUBIT	[[5,1,3]] qubit code
SEVEN_QUBIT_STEANE	[[7,1,3]] Steane qubit code
NINE_QUBIT_SHOR	[[9,1,3]] Shor qubit code

7.3.3 Constructor & Destructor Documentation

7.3.3.1 Codes()

```
qpp::Codes::Codes ( )  [inline], [private]
```

Default constructor.

7.3.3.2 ∼Codes()

```
qpp::Codes::∼Codes ( )  [private], [default]
```

Default destructor.

7.3.4 Member Function Documentation

7.3.4.1 codeword()

```
ket qpp::Codes::codeword (
             Type type,
             idx i ) const  [inline]
```

Returns the codeword of the specified code type.

See also

qpp::Codes::Type

Parameters

type	Code type
i	Codeword index

Returns

i-th codeword of the code *type*

7.3.5 Friends And Related Function Documentation

7.3.5.1 internal::Singleton< const Codes >

```
friend class internal::Singleton< const Codes >  [friend]
```

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

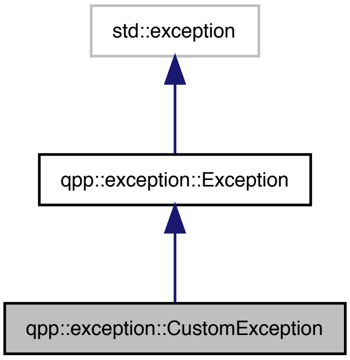
- classes/codes.h

7.4 qpp::exception::CustomException Class Reference

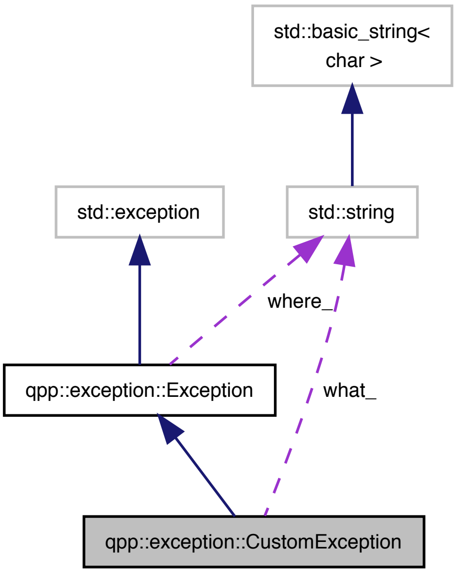
Custom exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::CustomException:



Collaboration diagram for qpp::exception::CustomException:



Public Member Functions

- CustomException (const std::string &where, const std::string &what)

Private Member Functions

- std::string type_description () const override

  Exception type description.

Private Attributes

- std::string what_ {}

7.4.1 Detailed Description

Custom exception.

Custom exception, the user must provide a custom message

7.4.2 Constructor & Destructor Documentation

7.4.2.1 CustomException()

```
qpp::exception::CustomException::CustomException (
            const std::string & where,
            const std::string & what )  [inline]
```

7.4.3 Member Function Documentation

7.4.3.1 type_description()

```
std::string qpp::exception::CustomException::type_description ( ) const  [inline], [override],
[private], [virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

7.4.4 Member Data Documentation

7.4.4.1 what_

```
std::string qpp::exception::CustomException::what_ {}  [private]
```

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

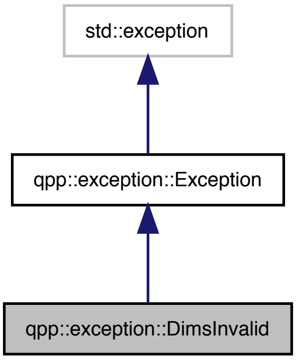
- classes/exception.h

7.5 qpp::exception::DimsInvalid Class Reference

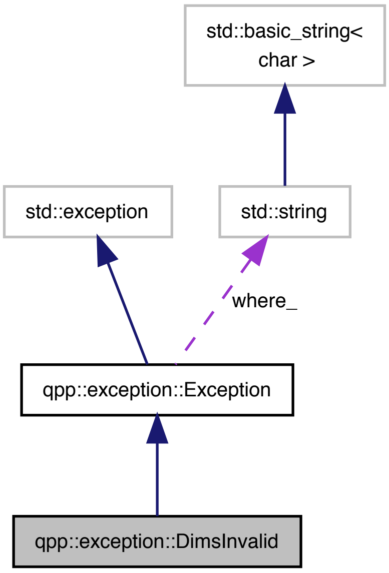
Invalid dimension(s) exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::DimsInvalid:



Collaboration diagram for qpp::exception::DimsInvalid:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.5.1 Detailed Description

Invalid dimension(s) exception.

std::vector<idx> of dimensions has zero size or contains zeros

7.5.2 Member Function Documentation

7.5.2.1 type_description()

```
std::string qpp::exception::DimsInvalid::type_description ( ) const  [inline], [override],
[virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

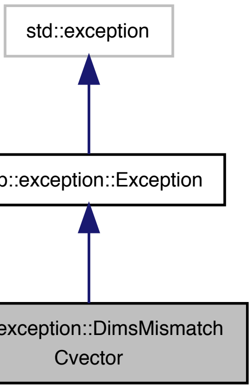
- classes/exception.h

7.6 qpp::exception::DimsMismatchCvector Class Reference

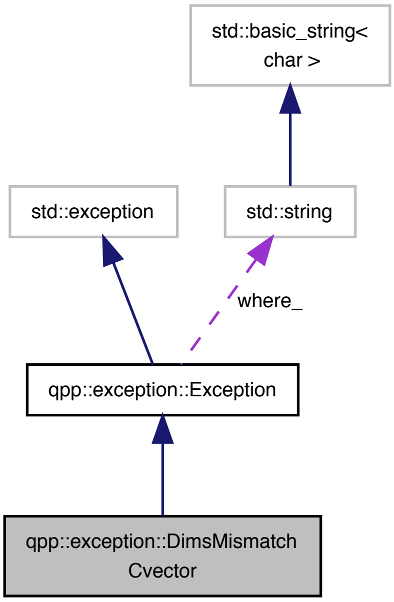
Dimension(s) mismatch column vector size exception.

```
#include <classes/exception.h>
```

Inheritance diagram for qpp::exception::DimsMismatchCvector:


std::exception
qpp::exception::Exception
qpp::exception::DimsMismatch Cvector

Collaboration diagram for qpp::exception::DimsMismatchCvector:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::DimsMismatch Cvector

Public Member Functions

- std::string type_description () const override
  Exception type description.

7.6.1 Detailed Description

Dimension(s) mismatch column vector size exception.

Product of the elements of std::vector<idx> of dimensions is not equal to the number of elements of the Eigen::↩ Matrix (assumed to be a column vector)

7.6.2 Member Function Documentation

7.6.2.1 type_description()

`std::string qpp::exception::DimsMismatchCvector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

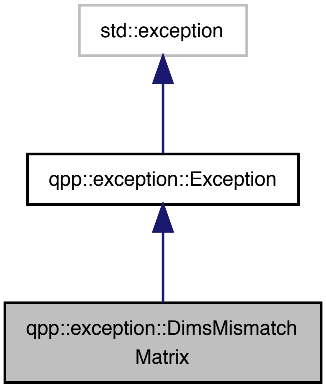
- classes/exception.h

7.7 qpp::exception::DimsMismatchMatrix Class Reference

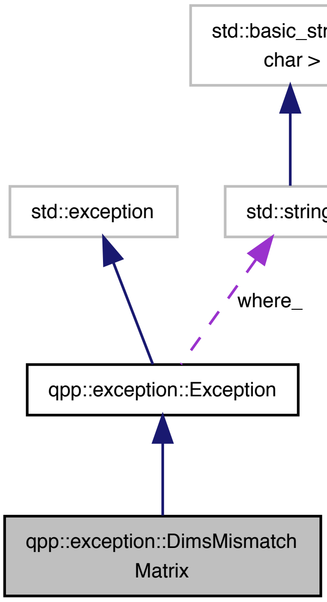
Dimension(s) mismatch matrix size exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::DimsMismatchMatrix:


std::exception
qpp::exception::Exception
qpp::exception::DimsMismatch Matrix

Collaboration diagram for qpp::exception::DimsMismatchMatrix:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.7.1 Detailed Description

Dimension(s) mismatch matrix size exception.

Product of the elements of std::vector<idx> of dimensions is not equal to the number of rows of the Eigen::Matrix (assumed to be a square matrix)

7.7.2 Member Function Documentation

7.7.2.1 type_description()

`std::string qpp::exception::DimsMismatchMatrix::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

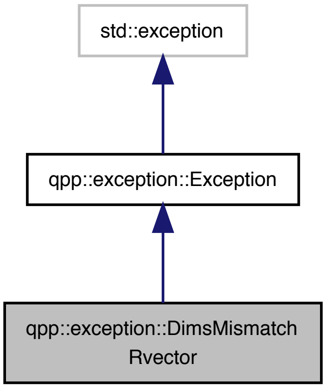
- classes/exception.h

7.8 qpp::exception::DimsMismatchRvector Class Reference

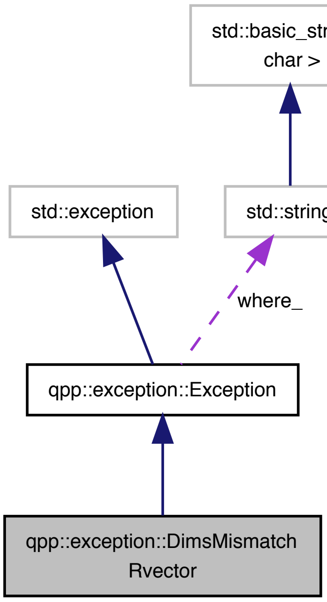
Dimension(s) mismatch row vector size exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::DimsMismatchRvector:


std::exception
qpp::exception::Exception
qpp::exception::DimsMismatch Rvector

Collaboration diagram for qpp::exception::DimsMismatchRvector:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::DimsMismatch Rvector

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.8.1 Detailed Description

Dimension(s) mismatch row vector size exception.

Product of the elements of std::vector<idx> of dimensions is not equal to the number of elements of the Eigen::↩ Matrix (assumed to be a row vector)

7.8.2 Member Function Documentation

7.8.2.1 type_description()

```
std::string qpp::exception::DimsMismatchRvector::type_description ( ) const  [inline], [override],
[virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

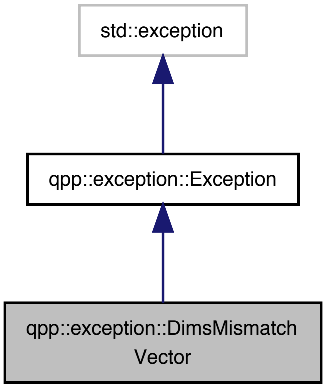
- classes/exception.h

7.9 qpp::exception::DimsMismatchVector Class Reference

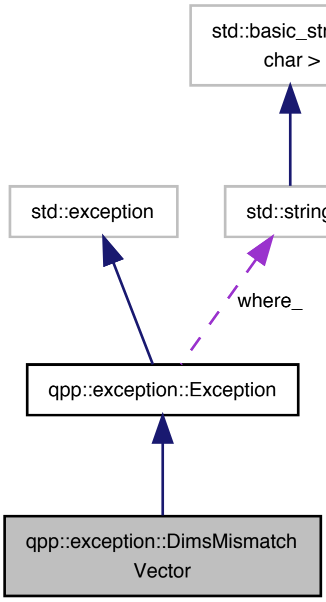
Dimension(s) mismatch vector size exception.

```
#include <classes/exception.h>
```

Inheritance diagram for qpp::exception::DimsMismatchVector:


std::exception
qpp::exception::Exception
qpp::exception::DimsMismatch Vector

Collaboration diagram for qpp::exception::DimsMismatchVector:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.9.1 Detailed Description

Dimension(s) mismatch vector size exception.

Product of the elements of std::vector<idx> of dimensions is not equal to the number of elements of the Eigen::↩ Matrix (assumed to be a row/column vector)

7.9.2 Member Function Documentation

7.9.2.1 type_description()

```
std::string qpp::exception::DimsMismatchVector::type_description ( ) const  [inline], [override],
[virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

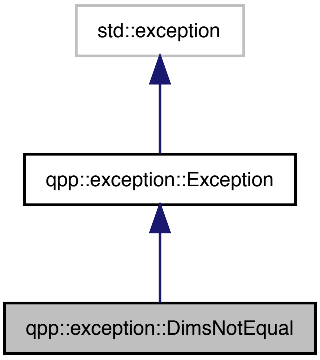
- classes/exception.h

7.10 qpp::exception::DimsNotEqual Class Reference

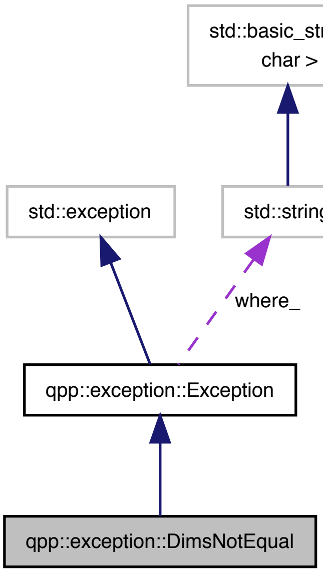
Dimensions not equal exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::DimsNotEqual:



Collaboration diagram for qpp::exception::DimsNotEqual:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.10.1 Detailed Description

Dimensions not equal exception.

Local/global dimensions are not equal

7.10.2 Member Function Documentation

7.10.2.1 type_description()

```
std::string qpp::exception::DimsNotEqual::type_description ( ) const  [inline], [override],
[virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

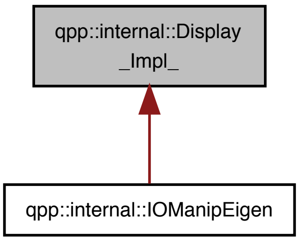
The documentation for this class was generated from the following file:

- classes/exception.h

7.11 qpp::internal::Display_Impl_ Struct Reference

`#include <internal/util.h>`

Inheritance diagram for qpp::internal::Display_Impl_:



Public Member Functions

- template<typename T >
  std::ostream & display_impl_ (const T &A, std::ostream &os, double chop=qpp::chop) const

7.11.1 Member Function Documentation

7.11.1.1 display_impl_()

```
template<typename T >
std::ostream& qpp::internal::Display_Impl_::display_impl_ (
             const T & A,
             std::ostream & os,
             double chop = qpp::chop ) const  [inline]
```

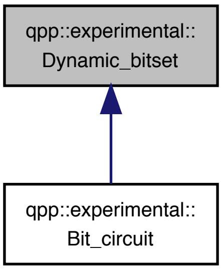
The documentation for this struct was generated from the following file:

- internal/util.h

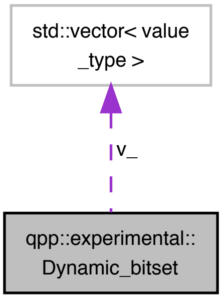
7.12 qpp::experimental::Dynamic_bitset Class Reference

`#include <experimental/experimental.h>`

Inheritance diagram for qpp::experimental::Dynamic_bitset:



Collaboration diagram for qpp::experimental::Dynamic_bitset:



Public Types

- using value_type = unsigned int

  Type of the storage elements.
- using storage_type = std::vector< value_type >

  Type of the storage.

Public Member Functions

- Dynamic_bitset (idx N)

  Constructor, initializes all bits to false (zero)
- const storage_type & data () const

  Raw storage space of the bitset.
- idx size () const

  Number of bits stored in the bitset.
- idx storage_size () const

  Size of the underlying storage space (in units of value_type, unsigned int by default)
- idx count () const noexcept
- bool get (idx pos) const
- bool none () const noexcept
- bool all () const noexcept
- bool any () const noexcept
- Dynamic_bitset & set (idx pos, bool value=true)
- Dynamic_bitset & set () noexcept
- Dynamic_bitset & rand (idx pos, double p=0.5)
- Dynamic_bitset & rand (double p=0.5)
- Dynamic_bitset & reset (idx pos)
- Dynamic_bitset & reset () noexcept
- Dynamic_bitset & flip (idx pos)
- Dynamic_bitset & flip () noexcept
- bool operator== (const Dynamic_bitset &rhs) const noexcept
- bool operator!= (const Dynamic_bitset &rhs) const noexcept
- template<class CharT = char, class Traits = std::char_traits<CharT>, class Allocator = std::allocator<CharT>>
  std::basic_string< CharT, Traits, Allocator > to_string (CharT zero=CharT('0'), CharT one=CharT('1')) const

Protected Member Functions

- idx index_ (idx pos) const

  Index of the pos bit in the storage space.
- idx offset_ (idx pos) const

  Offset of the pos bit in the storage space relative to its index.

Protected Attributes

- idx storage_size_

  Storage size.
- idx N_

  Number of bits.
- std::vector< value_type > v_

  Storage space.

Friends

- std::ostream & operator<< (std::ostream &os, const Dynamic_bitset &rhs)

7.12.1 Member Typedef Documentation

7.12.1.1 storage_type

```
using qpp::experimental::Dynamic_bitset::storage_type = std::vector<value_type>
```

Type of the storage.

7.12.1.2 value_type

```
using qpp::experimental::Dynamic_bitset::value_type = unsigned int
```

Type of the storage elements.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 Dynamic_bitset()

```
qpp::experimental::Dynamic_bitset::Dynamic_bitset (
             idx N )  [inline]
```

Constructor, initializes all bits to false (zero)

Parameters

N	Number of bits in the bitset

7.12.3 Member Function Documentation

7.12.3.1 all()

```
bool qpp::experimental::Dynamic_bitset::all ( ) const  [inline], [noexcept]
```

Returns

7.12.3.2 any()

```
bool qpp::experimental::Dynamic_bitset::any ( ) const  [inline], [noexcept]
```

Returns

7.12.3.3 count()

```
idx qpp::experimental::Dynamic_bitset::count ( ) const  [inline], [noexcept]
```

Returns

7.12.3.4 data()

```
const storage_type& qpp::experimental::Dynamic_bitset::data ( ) const  [inline]
```

Raw storage space of the bitset.

Returns

Const reference to the underlying storage space

7.12.3.5 flip() [1/2]

```
Dynamic_bitset& qpp::experimental::Dynamic_bitset::flip (
             idx pos )  [inline]
```

Parameters

pos	

Returns

7.12.3.6 flip() [2/2]

```
Dynamic_bitset& qpp::experimental::Dynamic_bitset::flip ( )  [inline], [noexcept]
```

Returns

7.12.3.7 get()

```
bool qpp::experimental::Dynamic_bitset::get (
            idx pos ) const  [inline]
```

Parameters

pos	

Returns

7.12.3.8 index_()

```
idx qpp::experimental::Dynamic_bitset::index_ (
            idx pos ) const  [inline], [protected]
```

Index of the *pos* bit in the storage space.

Parameters

pos	Bit location

Returns

Index of the *pos* bit in the storage space

7.12.3.9 none()

```
bool qpp::experimental::Dynamic_bitset::none ( ) const  [inline], [noexcept]
```

Returns

7.12.3.10 offset_()

```
idx qpp::experimental::Dynamic_bitset::offset_ (
            idx pos ) const  [inline], [protected]
```

Offset of the *pos* bit in the storage space relative to its index.

Parameters

pos	Bit location

Returns

Offset of the *pos* bit in the storage space relative to its index

7.12.3.11 operator"!=()

```
bool qpp::experimental::Dynamic_bitset::operator!= (
            const Dynamic_bitset & rhs ) const  [inline], [noexcept]
```

Parameters

rhs	

Returns

7.12.3.12 operator==()

```
bool qpp::experimental::Dynamic_bitset::operator== (
            const Dynamic_bitset & rhs ) const  [inline], [noexcept]
```

Parameters

rhs	

Returns

7.12.3.13 rand() [1/2]

```
Dynamic_bitset& qpp::experimental::Dynamic_bitset::rand (
             idx pos,
             double p = 0.5 )  [inline]
```

Parameters

pos	
p	

Returns

7.12.3.14 rand() [2/2]

```
Dynamic_bitset& qpp::experimental::Dynamic_bitset::rand (
             double p = 0.5 )  [inline]
```

Parameters

p	

Returns

7.12.3.15 reset() [1/2]

```
Dynamic_bitset& qpp::experimental::Dynamic_bitset::reset (
             idx pos )  [inline]
```

Parameters

pos	

Returns

7.12.3.16 reset() [2/2]

```
Dynamic_bitset& qpp::experimental::Dynamic_bitset::reset ( )  [inline], [noexcept]
```

Returns

7.12.3.17 set() [1/2]

```
Dynamic_bitset& qpp::experimental::Dynamic_bitset::set (
            idx pos,
            bool value = true )  [inline]
```

Parameters

pos	
value	

Returns

7.12.3.18 set() [2/2]

```
Dynamic_bitset& qpp::experimental::Dynamic_bitset::set ( )  [inline], [noexcept]
```

Returns

7.12.3.19 size()

```
idx qpp::experimental::Dynamic_bitset::size ( ) const  [inline]
```

Number of bits stored in the bitset.

Returns

Number of bits

7.12.3.20 storage_size()

```
idx qpp::experimental::Dynamic_bitset::storage_size ( ) const  [inline]
```

Size of the underlying storage space (in units of value_type, unsigned int by default)

Returns

Size of the underlying storage space

7.12.3.21 to_string()

```
template<class CharT = char, class Traits = std::char_traits<CharT>, class Allocator = std↩
::allocator<CharT>>
std::basic_string<CharT, Traits, Allocator> qpp::experimental::Dynamic_bitset::to_string (
             CharT zero = CharT('0'),
             CharT one = CharT('1') ) const  [inline]
```

Template Parameters

CharT	
Traits	
Allocator	

Parameters

zero	
one	

Returns

7.12.4 Friends And Related Function Documentation

7.12.4.1 operator<<

```
std::ostream& operator<< (
             std::ostream & os,
             const Dynamic_bitset & rhs )  [friend]
```

Parameters

os	
rhs	

Returns

7.12.5 Member Data Documentation

7.12.5.1 N_

`idx qpp::experimental::Dynamic_bitset::N_ [protected]`

Number of bits.

7.12.5.2 storage_size_

`idx qpp::experimental::Dynamic_bitset::storage_size_ [protected]`

Storage size.

7.12.5.3 v_

`std::vector<value_type> qpp::experimental::Dynamic_bitset::v_ [protected]`

Storage space.

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

- experimental/experimental.h

7.13 qpp::Dynamic_bitset Class Reference

Dynamic bitset class, allows the specification of the number of bits at runtime (unlike std::bitset<N>)

`#include <experimental/experimental.h>`

7.13.1 Detailed Description

Dynamic bitset class, allows the specification of the number of bits at runtime (unlike std::bitset<N>)

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

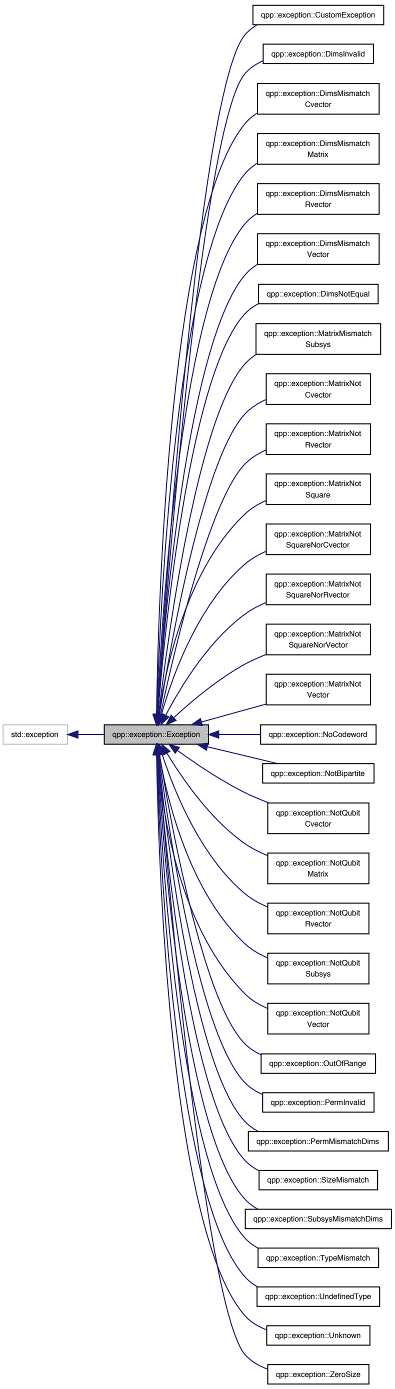
- experimental/experimental.h

7.14 qpp::exception::Exception Class Reference

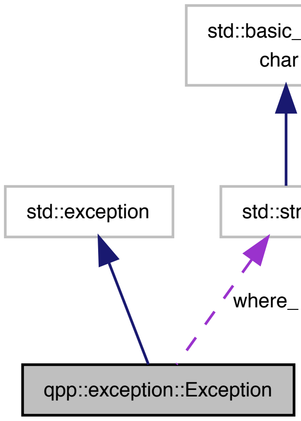
Base class for generating Quantum++ custom exceptions.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::Exception:


qpp::exception::CustomException
qpp::exception::DimsInvalid
qpp::exception::DimsMismatch Cvector
qpp::exception::DimsMismatch Matrix
qpp::exception::DimsMismatch Rvector
qpp::exception::DimsMismatch Vector
qpp::exception::DimsNotEqual
qpp::exception::MatrixMismatch Subsys
qpp::exception::MatrixNot Cvector
qpp::exception::MatrixNot Rvector
qpp::exception::MatrixNot Square
qpp::exception::MatrixNot SquareNorCvector
qpp::exception::MatrixNot SquareNorRvector
qpp::exception::MatrixNot SquareNorVector
qpp::exception::MatrixNot Vector
std::exception
qpp::exception::Exception
qpp::exception::NoCodeword
qpp::exception::NotBipartite
qpp::exception::NotQubit Cvector
qpp::exception::NotQubit Matrix
qpp::exception::NotQubit Rvector
qpp::exception::NotQubit Subsys
qpp::exception::NotQubit Vector
qpp::exception::OutOfRange
qpp::exception::PermInvalid
qpp::exception::PermMismatchDims
qpp::exception::SizeMismatch
qpp::exception::SubsysMismatchDims
qpp::exception::TypeMismatch
qpp::exception::UndefinedType
qpp::exception::Unknown
qpp::exception::ZeroSize

Collaboration diagram for qpp::exception::Exception:



Public Member Functions

- Exception (const std::string &where)

  Constructs an exception.
- virtual const char $*$ what () const noexcept override

  Overrides std::exception::what()
- virtual std::string type_description () const =0

  Exception type description.

Private Attributes

- std::string where_

7.14.1 Detailed Description

Base class for generating Quantum++ custom exceptions.

Derive from this class if more exceptions are needed, making sure to override qpp::exception::Exception::type_description() in the derived class and to inherit the constructor qpp::exception::Exception::Exception(). Preferably keep your newly defined exception classes in the namespace qpp::exception.

Example:

```
namespace qpp
{
namespace exception
{
    class ZeroSize : public Exception
    {
    public:
        std::string type_description() const override
        {
            return "Object has zero size";
        }

        // inherit the base class' qpp::exception::Exception constructor
        using Exception::Exception;
    };
} // namespace exception
} // namespace qpp
```

7.14.2 Constructor & Destructor Documentation

7.14.2.1 Exception()

```
qpp::exception::Exception::Exception (
            const std::string & where )  [inline]
```

Constructs an exception.

Parameters

where	Text representing where the exception occurred

7.14.3 Member Function Documentation

7.14.3.1 type_description()

```
std::string qpp::exception::Exception::type_description ( ) const  [inline], [pure virtual]
```

Exception type description.

Returns

Exception type description

Implemented in qpp::exception::CustomException, qpp::exception::UndefinedType, qpp::exception::SizeMismatch, qpp::exception::TypeMismatch, qpp::exception::OutOfRange, qpp::exception::NoCodeword, qpp::exception::NotBipartite, qpp::exception::NotQubitSubsys, qpp::exception::NotQubitVector, qpp::exception::NotQubitRvector, qpp::exception::NotQubitCvector, qpp::exception::NotQubitMatrix, qpp::exception::PermMismatchDims, qpp::exception::PermInvalid, qpp::exception::SubsysMismatchDims, qpp::exception::DimsMismatchVector, qpp::exception::DimsMismatchRvector, qpp::exception::DimsMismatchCvector, qpp::exception::DimsMismatchMatrix, qpp::exception::DimsNotEqual, qpp::exception::DimsInvalid, qpp::exception::MatrixMismatchSubsys, qpp::exception::MatrixNotSquareNorVector, qpp::exception::MatrixNotSquareNorRvector, qpp::exception::MatrixNotSquareNorCvector, qpp::exception::MatrixNotVector, qpp::exception::MatrixNotRvector, qpp::exception::MatrixNotCvector, qpp::exception::MatrixNotSquare, qpp::exception::ZeroSize, and qpp::exception::Unknown.

7.14.3.2 what()

```
virtual const char* qpp::exception::Exception::what ( ) const  [inline], [override], [virtual],
[noexcept]
```

Overrides std::exception::what()

Returns

Exception description

7.14.4 Member Data Documentation

7.14.4.1 where_

```
std::string qpp::exception::Exception::where_ [private]
```

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

- classes/exception.h

7.15 qpp::experimental::Bit_circuit::Gate_count Struct Reference

```
#include <experimental/experimental.h>
```

Public Attributes

- idx NOT = 0
- idx & X = NOT
- idx CNOT = 0
- idx SWAP = 0
- idx FRED = 0
- idx TOF = 0

7.15.1 Member Data Documentation

7.15.1.1 CNOT

```
idx qpp::experimental::Bit_circuit::Gate_count::CNOT = 0
```

7.15.1.2 FRED

```
idx qpp::experimental::Bit_circuit::Gate_count::FRED = 0
```

7.15.1.3 NOT

`idx qpp::experimental::Bit_circuit::Gate_count::NOT = 0`

7.15.1.4 SWAP

`idx qpp::experimental::Bit_circuit::Gate_count::SWAP = 0`

7.15.1.5 TOF

`idx qpp::experimental::Bit_circuit::Gate_count::TOF = 0`

7.15.1.6 X

`idx& qpp::experimental::Bit_circuit::Gate_count::X = NOT`

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

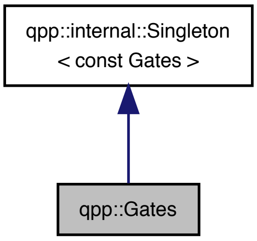
- experimental/experimental.h

7.16 qpp::Gates Class Reference

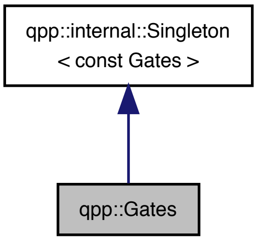
const Singleton class that implements most commonly used gates

`#include <classes/gates.h>`

Inheritance diagram for qpp::Gates:



Collaboration diagram for qpp::Gates:



Public Member Functions

- cmat Rn (double theta, const std::vector< double > &n) const

  Qubit rotation of theta about the 3-dimensional real (unit) vector n.
- cmat Zd (idx D=2) const

  Generalized Z gate for qudits.
- cmat Fd (idx D=2) const

  Fourier transform gate for qudits.
- cmat Xd (idx D=2) const

  Generalized X gate for qudits.
- template<typename Derived = Eigen::MatrixXcd>
  Derived Id (idx D=2) const

  Identity gate.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > CTRL (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &ctrl, const std::vector< idx > &subsys, idx N, idx d=2) const

  Generates the multi-partite multiple-controlled-A gate in matrix form.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > expandout (const Eigen::MatrixBase< Derived > &A, idx pos, const std::vector< idx > &dims) const

  Expands out.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > expandout (const Eigen::MatrixBase< Derived > &A, idx pos, const std::initializer_list< idx > &dims) const

  Expands out.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > expandout (const Eigen::MatrixBase< Derived > &A, idx pos, idx N, idx d=2) const

  Expands out.

Public Attributes

- cmat Id2 {cmat::Identity(2, 2)}

  Identity gate.
- cmat H {cmat::Zero(2, 2)}

Hadamard gate.
- cmat X {cmat::Zero(2, 2)}

  Pauli Sigma-X gate.
- cmat Y {cmat::Zero(2, 2)}

  Pauli Sigma-Y gate.
- cmat Z {cmat::Zero(2, 2)}

  Pauli Sigma-Z gate.
- cmat S {cmat::Zero(2, 2)}

  S gate.
- cmat T {cmat::Zero(2, 2)}

  T gate.
- cmat CNOT {cmat::Identity(4, 4)}

  Controlled-NOT control target gate.
- cmat CZ {cmat::Identity(4, 4)}

  Controlled-Phase gate.
- cmat CNOTba {cmat::Zero(4, 4)}

  Controlled-NOT target control gate.
- cmat SWAP {cmat::Identity(4, 4)}

  SWAP gate.
- cmat TOF {cmat::Identity(8, 8)}

  Toffoli gate.
- cmat FRED {cmat::Identity(8, 8)}

  Fredkin gate.

Private Member Functions

- Gates ()

  Initializes the gates.
- ∼Gates ()=default

  Default destructor.

Friends

- class internal::Singleton< const Gates >

Additional Inherited Members

7.16.1 Detailed Description

const Singleton class that implements most commonly used gates

7.16.2 Constructor & Destructor Documentation

7.16.2.1 Gates()

```
qpp::Gates::Gates ( )  [inline], [private]
```

Initializes the gates.

7.16.2.2 ∼Gates()

```
qpp::Gates::∼Gates ( )  [private], [default]
```

Default destructor.

7.16.3 Member Function Documentation

7.16.3.1 CTRL()

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::Gates::CTRL (
            const Eigen::MatrixBase< Derived > & A,
            const std::vector< idx > & ctrl,
            const std::vector< idx > & subsys,
            idx N,
            idx d = 2 ) const  [inline]
```

Generates the multi-partite multiple-controlled-*A* gate in matrix form.

See also

qpp::applyCTRL()

Note

The dimension of the gate *A* must match the dimension of *subsys*

Parameters

A	Eigen expression
ctrl	Control subsystem indexes
subsys	Subsystem indexes where the gate *A* is applied
N	Total number of subsystems
d	Subsystem dimensions

Returns

CTRL-A gate, as a matrix over the same scalar field as *A*

7.16.3.2 expandout() [1/3]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::Gates::expandout (
            const Eigen::MatrixBase< Derived > & A,
            idx pos,
            const std::vector< idx > & dims ) const  [inline]
```

Expands out.

See also

qpp::kron()

Expands out *A* as a matrix in a multi-partite system. Faster than using qpp::kron(I, I, ..., I, A, I, ..., I).

Parameters

A	Eigen expression
pos	Position
dims	Dimensions of the multi-partite system

Returns

Tensor product $I \otimes \cdots \otimes I \otimes A \otimes I \otimes \cdots \otimes I$, with *A* on position *pos*, as a dynamic matrix over the same scalar field as *A*

7.16.3.3 expandout() [2/3]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::Gates::expandout (
            const Eigen::MatrixBase< Derived > & A,
            idx pos,
            const std::initializer_list< idx > & dims ) const  [inline]
```

Expands out.

See also

qpp::kron()

Expands out *A* as a matrix in a multi-partite system. Faster than using qpp::kron(I, I, ..., I, A, I, ..., I).

Note

The std::initializer_list overload exists because otherwise, in the degenerate case when *dims* has only one element, the one element list is implicitly converted to the element's underlying type, i.e. qpp::idx, which has the net effect of picking the wrong (non-vector) qpp::expandout() overload

Parameters

A	Eigen expression
pos	Position
dims	Dimensions of the multi-partite system

Returns

Tensor product $I \otimes \cdots \otimes I \otimes A \otimes I \otimes \cdots \otimes I$, with *A* on position *pos*, as a dynamic matrix over the same scalar field as *A*

7.16.3.4 expandout() [3/3]

```
template<typename Derived >
dyn_mat<typename Derived::Scalar> qpp::Gates::expandout (
             const Eigen::MatrixBase< Derived > & A,
             idx pos,
             idx N,
             idx d = 2 ) const  [inline]
```

Expands out.

See also

qpp::kron()

Expands out *A* as a matrix in a multi-partite system. Faster than using qpp::kron(I, I, ..., I, A, I, ..., I).

Parameters

A	Eigen expression
pos	Position
N	Number of subsystems
d	Subsystem dimension

Returns

Tensor product $I \otimes \cdots \otimes I \otimes A \otimes I \otimes \cdots \otimes I$, with *A* on position *pos*, as a dynamic matrix over the same scalar field as *A*

7.16.3.5 Fd()

```
cmat qpp::Gates::Fd (
             idx D = 2 ) const  [inline]
```

Fourier transform gate for qudits.

Note

Defined as $F = \sum_{j,k=0}^{D-1} \exp(2\pi \mathrm{i} jk/D) |j\rangle\langle k|$

Parameters

D	Dimension of the Hilbert space

Returns

Fourier transform gate for qudits

7.16.3.6 Id()

```
template<typename Derived = Eigen::MatrixXcd>
Derived qpp::Gates::Id (
             idx D = 2 ) const  [inline]
```

Identity gate.

Note

Can change the return type from complex matrix (default) by explicitly specifying the template parameter

Parameters

D	Dimension of the Hilbert space

Returns

Identity gate on a Hilbert space of dimension *D*

7.16.3.7 Rn()

```
cmat qpp::Gates::Rn (
             double theta,
             const std::vector< double > & n ) const  [inline]
```

Qubit rotation of *theta* about the 3-dimensional real (unit) vector *n*.

Parameters

theta	Rotation angle
n	3-dimensional real (unit) vector

Returns

Rotation gate

7.16.3.8 Xd()

```
cmat qpp::Gates::Xd (
             idx D = 2 ) const  [inline]
```

Generalized X gate for qudits.

Note

Defined as $X = \sum_{j=0}^{D-1} |j \oplus 1\rangle\langle j|$, i.e. raising operator $X|j\rangle = |j \oplus 1\rangle$

Parameters

D	Dimension of the Hilbert space

Returns

Generalized X gate for qudits

7.16.3.9 Zd()

```
cmat qpp::Gates::Zd (
             idx D = 2 ) const  [inline]
```

Generalized Z gate for qudits.

Note

Defined as $Z = \sum_{j=0}^{D-1} \exp(2\pi \mathrm{i} j/D) |j\rangle\langle j|$

Parameters

D	Dimension of the Hilbert space

Returns

Generalized Z gate for qudits

7.16.4 Friends And Related Function Documentation

7.16.4.1 internal::Singleton< const Gates >

`friend class internal::Singleton< const Gates > [friend]`

7.16.5 Member Data Documentation

7.16.5.1 CNOT

`cmat qpp::Gates::CNOT {cmat::Identity(4, 4)}`

Controlled-NOT control target gate.

7.16.5.2 CNOTba

`cmat qpp::Gates::CNOTba {cmat::Zero(4, 4)}`

Controlled-NOT target control gate.

7.16.5.3 CZ

`cmat qpp::Gates::CZ {cmat::Identity(4, 4)}`

Controlled-Phase gate.

7.16.5.4 FRED

`cmat qpp::Gates::FRED {cmat::Identity(8, 8)}`

Fredkin gate.

7.16.5.5 H

`cmat qpp::Gates::H {cmat::Zero(2, 2)}`

Hadamard gate.

7.16.5.6 Id2

```
cmat qpp::Gates::Id2 {cmat::Identity(2, 2)}
```

Identity gate.

7.16.5.7 S

```
cmat qpp::Gates::S {cmat::Zero(2, 2)}
```

S gate.

7.16.5.8 SWAP

```
cmat qpp::Gates::SWAP {cmat::Identity(4, 4)}
```

SWAP gate.

7.16.5.9 T

```
cmat qpp::Gates::T {cmat::Zero(2, 2)}
```

T gate.

7.16.5.10 TOF

```
cmat qpp::Gates::TOF {cmat::Identity(8, 8)}
```

Toffoli gate.

7.16.5.11 X

```
cmat qpp::Gates::X {cmat::Zero(2, 2)}
```

Pauli Sigma-X gate.

7.16.5.12 Y

```
cmat qpp::Gates::Y {cmat::Zero(2, 2)}
```

Pauli Sigma-Y gate.

7.16.5.13 Z

```
cmat qpp::Gates::Z {cmat::Zero(2, 2)}
```

Pauli Sigma-Z gate.

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

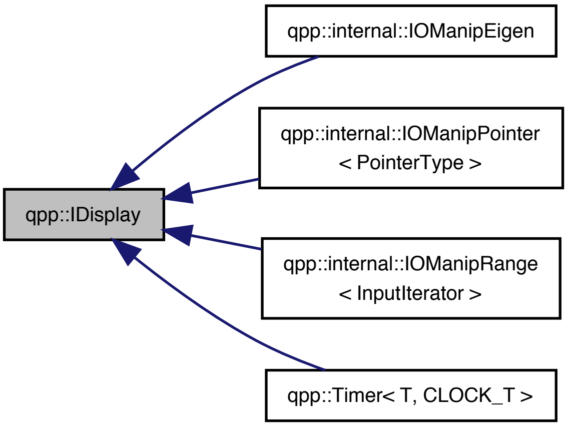
- classes/gates.h

7.17 qpp::IDisplay Class Reference

Abstract class (interface) that mandates the definition of virtual std::ostream& display(std::ostream& os) const.

`#include <classes/idisplay.h>`

Inheritance diagram for qpp::IDisplay:


qpp::internal::IOManipEigen
qpp::internal::IOManipPointer < PointerType >
qpp::IDisplay
qpp::internal::IOManipRange < InputIterator >
qpp::Timer< T, CLOCK_T >

Public Member Functions

- IDisplay ()=default

  Default constructor.
- IDisplay (const IDisplay &)=default

  Default copy constructor.
- IDisplay (IDisplay &&)=default

  Default move constructor.
- IDisplay & operator= (const IDisplay &)=default

  Default copy assignment operator.
- IDisplay & operator= (IDisplay &&)=default

  Default move assignment operator.
- virtual ∼IDisplay ()=default

  Default virtual destructor.

Private Member Functions

- virtual std::ostream & display (std::ostream &os) const =0

  Must be overridden by all derived classes.

Friends

- std::ostream & operator<< (std::ostream &os, const IDisplay &rhs)

  Overloads the extraction operator.

7.17.1 Detailed Description

Abstract class (interface) that mandates the definition of virtual std::ostream& display(std::ostream& os) const.

This class defines friend inline std::ostream& operator<< (std::ostream& os, const qpp::IDisplay& rhs). The latter delegates the work to the pure private virtual function qpp::IDisplay::display() which has to be overridden by all derived classes.

7.17.2 Constructor & Destructor Documentation

7.17.2.1 IDisplay() [1/3]

`qpp::IDisplay::IDisplay ( ) [default]`

Default constructor.

7.17.2.2 IDisplay() [2/3]

```
qpp::IDisplay::IDisplay (
            const IDisplay & ) [default]
```

Default copy constructor.

7.17.2.3 IDisplay() [3/3]

```
qpp::IDisplay::IDisplay (
            IDisplay && ) [default]
```

Default move constructor.

7.17.2.4 ∼IDisplay()

```
virtual qpp::IDisplay::∼IDisplay ( ) [virtual], [default]
```

Default virtual destructor.

7.17.3 Member Function Documentation

7.17.3.1 display()

```
virtual std::ostream& qpp::IDisplay::display (
            std::ostream & os ) const [private], [pure virtual]
```

Must be overridden by all derived classes.

The actual stream extraction processing is performed by the overriden member function in the derived class. This function is automatically invoked by friend inline std::ostream& operator<<(std::ostream& os, const IDisplay& rhs).

Implemented in qpp::internal::IOManipEigen, qpp::Timer< T, CLOCK_T >, qpp::internal::IOManipPointer< PointerType >, and qpp::internal::IOManipRange< InputIterator >.

7.17.3.2 operator=() [1/2]

```
IDisplay& qpp::IDisplay::operator= (
            const IDisplay & ) [default]
```

Default copy assignment operator.

7.17.3.3 operator=() [2/2]

```
IDisplay& qpp::IDisplay::operator= (
              IDisplay && )  [default]
```

Default move assignment operator.

7.17.4 Friends And Related Function Documentation

7.17.4.1 operator<<

```
std::ostream& operator<< (
              std::ostream & os,
              const IDisplay & rhs )  [friend]
```

Overloads the extraction operator.

Delegates the work to the virtual function qpp::IDisplay::display()

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

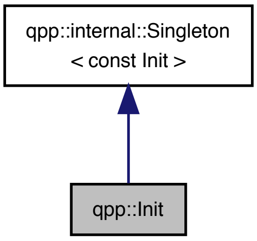
- classes/idisplay.h

7.18 qpp::Init Class Reference

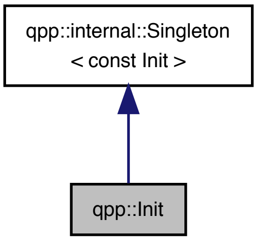
const Singleton class that performs additional initializations/cleanups

`#include <classes/init.h>`

Inheritance diagram for qpp::Init:



Collaboration diagram for qpp::Init:



Private Member Functions

- Init ()

  Additional initializations.
- ∼Init ()

  Cleanups.

Friends

- class internal::Singleton< const Init >

Additional Inherited Members

7.18.1 Detailed Description

const Singleton class that performs additional initializations/cleanups

7.18.2 Constructor & Destructor Documentation

7.18.2.1 Init()

`qpp::Init::Init ( ) [inline], [private]`

Additional initializations.

7.18.2.2 ∼Init()

`qpp::Init::∼Init ( ) [inline], [private]`

Cleanups.

7.18.3 Friends And Related Function Documentation

7.18.3.1 internal::Singleton< const Init >

`friend class internal::Singleton< const Init > [friend]`

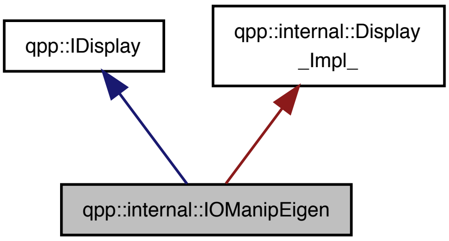
The documentation for this class was generated from the following file:

- classes/init.h

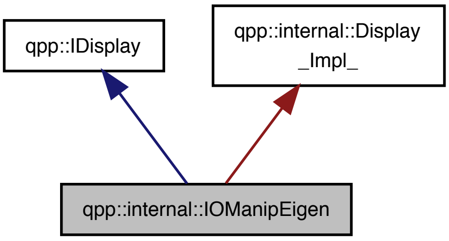
7.19 qpp::internal::IOManipEigen Class Reference

`#include <internal/classes/iomanip.h>`

Inheritance diagram for qpp::internal::IOManipEigen:



Collaboration diagram for qpp::internal::IOManipEigen:



Public Member Functions

- template<typename Derived >
  IOManipEigen (const Eigen::MatrixBase< Derived > &A, double chop=qpp::chop)
- IOManipEigen (const cplx z, double chop=qpp::chop)

Private Member Functions

- std::ostream & display (std::ostream &os) const override
  Must be overridden by all derived classes.

Private Attributes

- cmat A_
- double chop_

7.19.1 Constructor & Destructor Documentation

7.19.1.1 IOManipEigen() [1/2]

```
template<typename Derived >
qpp::internal::IOManipEigen::IOManipEigen (
            const Eigen::MatrixBase< Derived > & A,
            double chop = qpp::chop )  [inline], [explicit]
```

7.19.1.2 IOManipEigen() [2/2]

```
qpp::internal::IOManipEigen::IOManipEigen (
            const cplx z,
            double chop = qpp::chop )  [inline], [explicit]
```

7.19.2 Member Function Documentation

7.19.2.1 display()

```
std::ostream& qpp::internal::IOManipEigen::display (
            std::ostream & os ) const  [inline], [override], [private], [virtual]
```

Must be overridden by all derived classes.

The actual stream extraction processing is performed by the overriden member function in the derived class. This function is automatically invoked by friend inline std::ostream& operator<<(std::ostream& os, const IDisplay& rhs).

Implements qpp::IDisplay.

7.19.3 Member Data Documentation

7.19.3.1 A_

```
cmat qpp::internal::IOManipEigen::A_  [private]
```

7.19.3.2 chop_

```
double qpp::internal::IOManipEigen::chop_  [private]
```

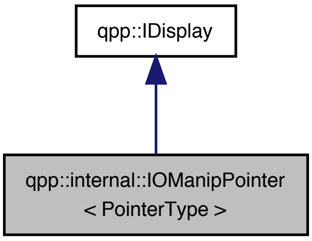
The documentation for this class was generated from the following file:

- internal/classes/iomanip.h

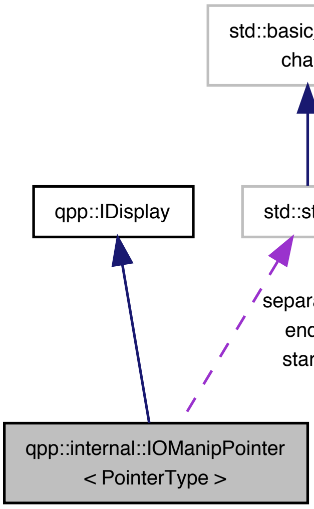
7.20 qpp::internal::IOManipPointer< PointerType > Class Template Reference

```
#include <internal/classes/iomanip.h>
```

Inheritance diagram for qpp::internal::IOManipPointer< PointerType >:



Collaboration diagram for qpp::internal::IOManipPointer< PointerType >:



Public Member Functions

- IOManipPointer (const PointerType ∗p, idx N, const std::string &separator, const std::string &start="[", const std::string &end="]")
- IOManipPointer (const IOManipPointer &)=default
- IOManipPointer & operator= (const IOManipPointer &)=default

Private Member Functions

- std::ostream & display (std::ostream &os) const override

  Must be overridden by all derived classes.

Private Attributes

- const PointerType ∗ p_
- idx N_
- std::string separator_
- std::string start_
- std::string end_

7.20.1 Constructor & Destructor Documentation

7.20.1.1 IOManipPointer() [1/2]

```
template<typename PointerType>
qpp::internal::IOManipPointer< PointerType >::IOManipPointer (
            const PointerType * p,
            idx N,
            const std::string & separator,
            const std::string & start = "[",
            const std::string & end = "]" )  [inline], [explicit]
```

7.20.1.2 IOManipPointer() [2/2]

```
template<typename PointerType>
qpp::internal::IOManipPointer< PointerType >::IOManipPointer (
            const IOManipPointer< PointerType > & )  [default]
```

7.20.2 Member Function Documentation

7.20.2.1 display()

```
template<typename PointerType>
std::ostream& qpp::internal::IOManipPointer< PointerType >::display (
            std::ostream & os ) const  [inline], [override], [private], [virtual]
```

Must be overridden by all derived classes.

The actual stream extraction processing is performed by the overriden member function in the derived class. This function is automatically invoked by friend inline std::ostream& operator<<(std::ostream& os, const IDisplay& rhs).

Implements qpp::IDisplay.

7.20.2.2 operator=()

```
template<typename PointerType>
IOManipPointer& qpp::internal::IOManipPointer< PointerType >::operator= (
            const IOManipPointer< PointerType > & )  [default]
```

7.20.3 Member Data Documentation

7.20.3.1 end_

```
template<typename PointerType>
std::string qpp::internal::IOManipPointer< PointerType >::end_  [private]
```

7.20.3.2 N_

```
template<typename PointerType>
idx qpp::internal::IOManipPointer< PointerType >::N_  [private]
```

7.20.3.3 p_

```
template<typename PointerType>
const PointerType* qpp::internal::IOManipPointer< PointerType >::p_  [private]
```

7.20.3.4 separator_

```
template<typename PointerType>
std::string qpp::internal::IOManipPointer< PointerType >::separator_  [private]
```

7.20.3.5 start_

```
template<typename PointerType>
std::string qpp::internal::IOManipPointer< PointerType >::start_  [private]
```

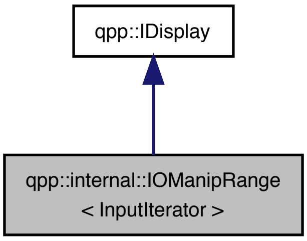
The documentation for this class was generated from the following file:

- internal/classes/iomanip.h

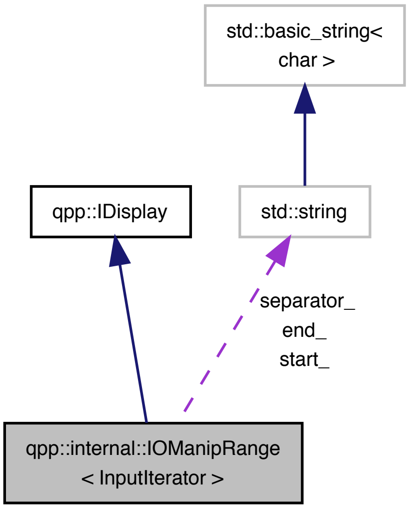
7.21 qpp::internal::IOManipRange< InputIterator > Class Template Reference

`#include <internal/classes/iomanip.h>`

Inheritance diagram for qpp::internal::IOManipRange< InputIterator >:



Collaboration diagram for qpp::internal::IOManipRange< InputIterator >:



Public Member Functions

- IOManipRange (InputIterator first, InputIterator last, const std::string &separator, const std::string &start="[", const std::string &end="]")
- IOManipRange (const IOManipRange &)=default
- IOManipRange & operator= (const IOManipRange &)=default

Private Member Functions

- std::ostream & display (std::ostream &os) const override
  Must be overridden by all derived classes.

Private Attributes

- InputIterator first_
- InputIterator last_
- std::string separator_
- std::string start_
- std::string end_

7.21.1 Constructor & Destructor Documentation

7.21.1.1 IOManipRange() [1/2]

```
template<typename InputIterator>
qpp::internal::IOManipRange< InputIterator >::IOManipRange (
            InputIterator first,
            InputIterator last,
            const std::string & separator,
            const std::string & start = "[",
            const std::string & end = "]" )  [inline], [explicit]
```

7.21.1.2 IOManipRange() [2/2]

```
template<typename InputIterator>
qpp::internal::IOManipRange< InputIterator >::IOManipRange (
            const IOManipRange< InputIterator > &  )  [default]
```

7.21.2 Member Function Documentation

7.21.2.1 display()

```
template<typename InputIterator>
std::ostream& qpp::internal::IOManipRange< InputIterator >::display (
            std::ostream & os ) const  [inline], [override], [private], [virtual]
```

Must be overridden by all derived classes.

The actual stream extraction processing is performed by the overriden member function in the derived class. This function is automatically invoked by friend inline std::ostream& operator<<(std::ostream& os, const IDisplay& rhs).

Implements qpp::IDisplay.

7.21.2.2 operator=()

```
template<typename InputIterator>
IOManipRange& qpp::internal::IOManipRange< InputIterator >::operator= (
             const IOManipRange< InputIterator > &  )  [default]
```

7.21.3 Member Data Documentation

7.21.3.1 end_

```
template<typename InputIterator>
std::string qpp::internal::IOManipRange< InputIterator >::end_  [private]
```

7.21.3.2 first_

```
template<typename InputIterator>
InputIterator qpp::internal::IOManipRange< InputIterator >::first_  [private]
```

7.21.3.3 last_

```
template<typename InputIterator>
InputIterator qpp::internal::IOManipRange< InputIterator >::last_  [private]
```

7.21.3.4 separator_

```
template<typename InputIterator>
std::string qpp::internal::IOManipRange< InputIterator >::separator_  [private]
```

7.21.3.5 start_

```
template<typename InputIterator>
std::string qpp::internal::IOManipRange< InputIterator >::start_  [private]
```

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

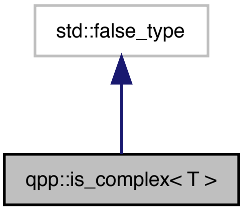
- internal/classes/iomanip.h

7.22 qpp::is_complex< T > Struct Template Reference

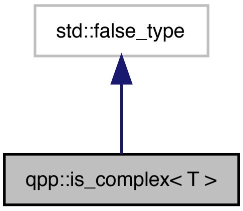
Checks whether the type is a complex type.

`#include <traits.h>`

Inheritance diagram for qpp::is_complex< T >:



Collaboration diagram for qpp::is_complex< T >:



7.22.1 Detailed Description

template<typename T>
struct qpp::is_complex< T >

Checks whether the type is a complex type.

Provides the constant member *value* which is equal to *true*, if the type is a complex type, i.e. *std::complex<T>*

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

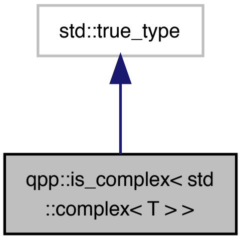
- traits.h

7.23 qpp::is_complex< std::complex< T > > Struct Template Reference

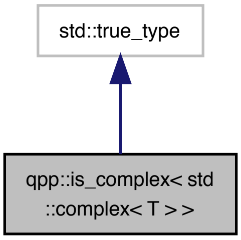
Checks whether the type is a complex number type, specialization for complex types.

`#include <traits.h>`

Inheritance diagram for qpp::is_complex< std::complex< T > >:



Collaboration diagram for qpp::is_complex< std::complex< T > >:



7.23.1 Detailed Description

template<typename T>
struct qpp::is_complex< std::complex< T > >

Checks whether the type is a complex number type, specialization for complex types.

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

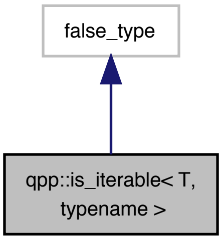
- traits.h

7.24 qpp::is_iterable< T, typename > Struct Template Reference

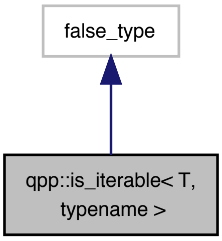
Checks whether *T* is compatible with an STL-like iterable container.

`#include <traits.h>`

Inheritance diagram for qpp::is_iterable< T, typename >:



Collaboration diagram for qpp::is_iterable< T, typename >:



7.24.1 Detailed Description

template<typename T, typename = void>
struct qpp::is_iterable< T, typename >

Checks whether *T* is compatible with an STL-like iterable container.

Provides the constant member *value* which is equal to *true*, if *T* is compatible with an iterable container, i.e. provides at least *begin()* and *end()* member functions. Otherwise, *value* is equal to *false*.

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

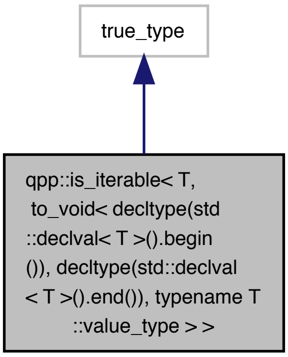
- traits.h

7.25 qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > > Struct Template Reference

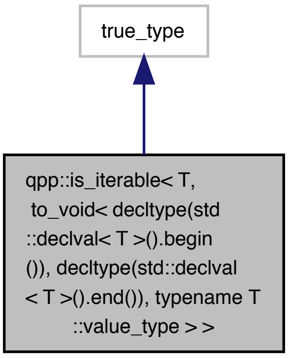
Checks whether *T* is compatible with an STL-like iterable container, specialization for STL-like iterable containers.

`#include <traits.h>`

Inheritance diagram for qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > >:


true_type
qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > >

Collaboration diagram for qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > >:


true_type
qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > >

7.25.1 Detailed Description

template<typename T>
struct qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T↩ ::value_type > >

Checks whether *T* is compatible with an STL-like iterable container, specialization for STL-like iterable containers.

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

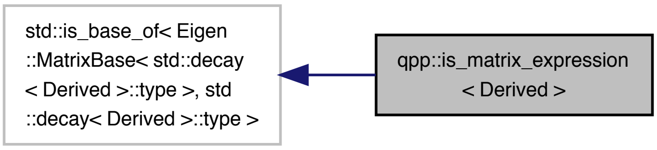
- traits.h

7.26 qpp::is_matrix_expression< Derived > Struct Template Reference

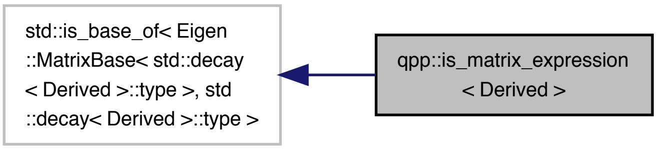
Checks whether the type is an Eigen matrix expression.

`#include <traits.h>`

Inheritance diagram for qpp::is_matrix_expression< Derived >:


std::is_base_of< Eigen ::MatrixBase< std::decay < Derived >::type >, std ::decay< Derived >::type >
qpp::is_matrix_expression < Derived >

Collaboration diagram for qpp::is_matrix_expression< Derived >:


std::is_base_of< Eigen ::MatrixBase< std::decay < Derived >::type >, std ::decay< Derived >::type >
qpp::is_matrix_expression < Derived >

7.26.1 Detailed Description

template<typename Derived>
struct qpp::is_matrix_expression< Derived >

Checks whether the type is an Eigen matrix expression.

Provides the constant member *value* which is equal to *true*, if the type is an Eigen matrix expression of type *Eigen::MatrixBase<Derived>*. Otherwise, *value* is equal to *false*.

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

- traits.h

7.27 qpp::make_void< Ts > Struct Template Reference

Helper for qpp::to_void<> alias template.

```
#include <traits.h>
```

Public Types

- typedef void type

7.27.1 Detailed Description

template<typename... Ts>
struct qpp::make_void< Ts >

Helper for qpp::to_void<> alias template.

See also

qpp::to_void<>

7.27.2 Member Typedef Documentation

7.27.2.1 type

```
template<typename... Ts>
typedef void qpp::make_void< Ts >::type
```

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

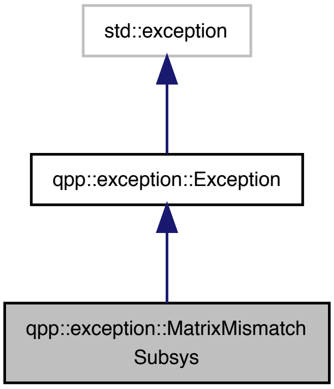
- traits.h

7.28 qpp::exception::MatrixMismatchSubsys Class Reference

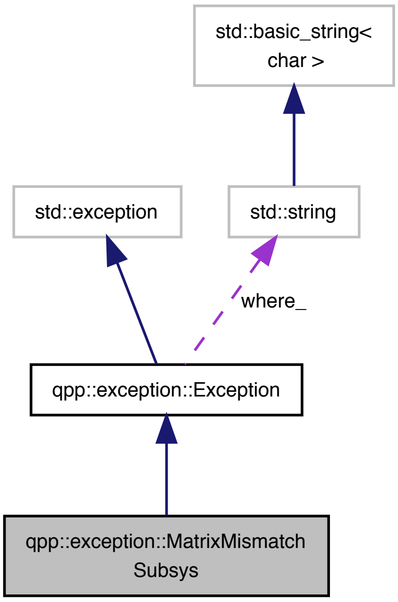
Matrix mismatch subsystems exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::MatrixMismatchSubsys:


std::exception
qpp::exception::Exception
qpp::exception::MatrixMismatch Subsys

Collaboration diagram for qpp::exception::MatrixMismatchSubsys:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::MatrixMismatch Subsys

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.28.1 Detailed Description

Matrix mismatch subsystems exception.

Matrix size mismatch subsystem sizes (e.g. in qpp::apply())

7.28.2 Member Function Documentation

7.28.2.1 type_description()

`std::string qpp::exception::MatrixMismatchSubsys::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

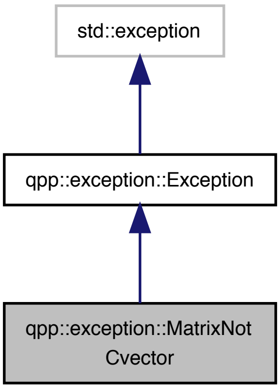
- classes/exception.h

7.29 qpp::exception::MatrixNotCvector Class Reference

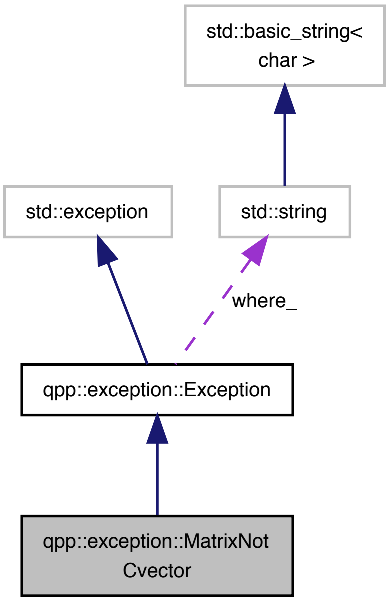
Matrix is not a column vector exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::MatrixNotCvector:



Collaboration diagram for qpp::exception::MatrixNotCvector:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::MatrixNot Cvector

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.29.1 Detailed Description

Matrix is not a column vector exception.

Eigen::Matrix is not a column vector

7.29.2 Member Function Documentation

7.29.2.1 type_description()

`std::string qpp::exception::MatrixNotCvector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

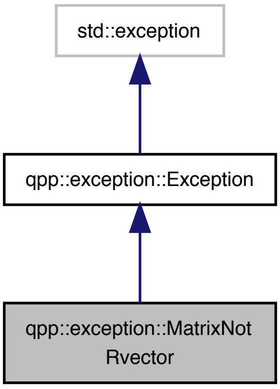
- classes/exception.h

7.30 qpp::exception::MatrixNotRvector Class Reference

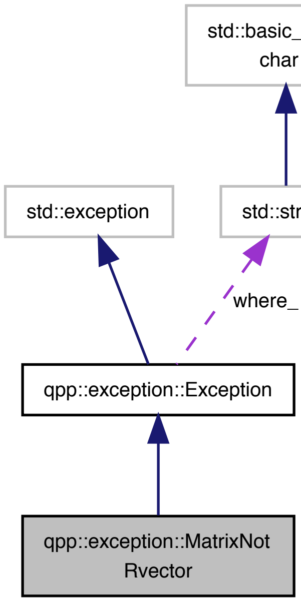
Matrix is not a row vector exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::MatrixNotRvector:



Collaboration diagram for qpp::exception::MatrixNotRvector:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.30.1 Detailed Description

Matrix is not a row vector exception.

Eigen::Matrix is not a row vector

7.30.2 Member Function Documentation

7.30.2.1 type_description()

`std::string qpp::exception::MatrixNotRvector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

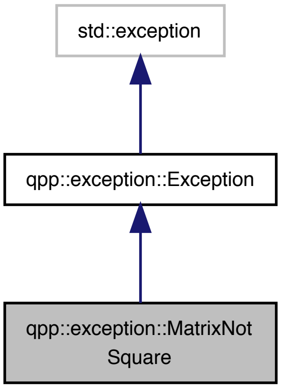
- classes/exception.h

7.31 qpp::exception::MatrixNotSquare Class Reference

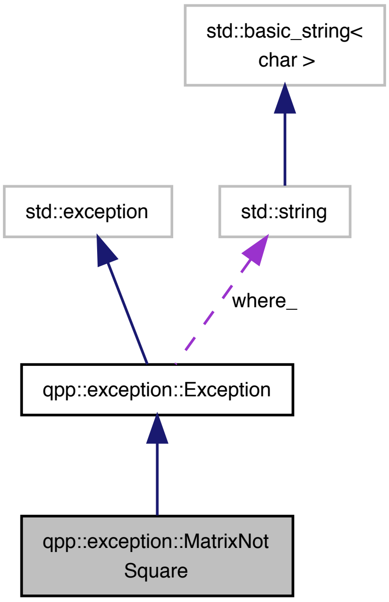
Matrix is not square exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::MatrixNotSquare:



Collaboration diagram for qpp::exception::MatrixNotSquare:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.31.1 Detailed Description

Matrix is not square exception.

Eigen::Matrix is not a square matrix

7.31.2 Member Function Documentation

7.31.2.1 type_description()

`std::string qpp::exception::MatrixNotSquare::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

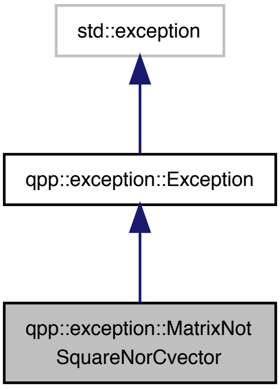
- classes/exception.h

7.32 qpp::exception::MatrixNotSquareNorCvector Class Reference

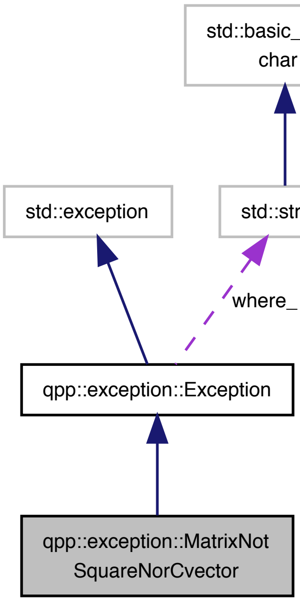
Matrix is not square nor column vector exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::MatrixNotSquareNorCvector:


std::exception
qpp::exception::Exception
qpp::exception::MatrixNot SquareNorCvector

Collaboration diagram for qpp::exception::MatrixNotSquareNorCvector:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.32.1 Detailed Description

Matrix is not square nor column vector exception.

Eigen::Matrix is not a square matrix nor a column vector

7.32.2 Member Function Documentation

7.32.2.1 type_description()

`std::string qpp::exception::MatrixNotSquareNorCvector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

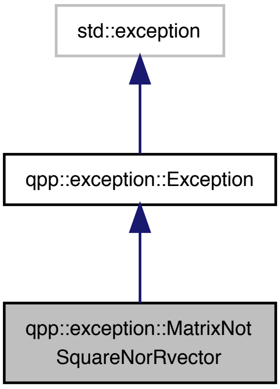
- classes/exception.h

7.33 qpp::exception::MatrixNotSquareNorRvector Class Reference

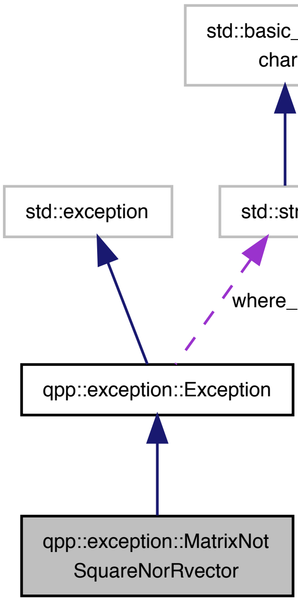
Matrix is not square nor row vector exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::MatrixNotSquareNorRvector:


std::exception
qpp::exception::Exception
qpp::exception::MatrixNot SquareNorRvector

Collaboration diagram for qpp::exception::MatrixNotSquareNorRvector:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.33.1 Detailed Description

Matrix is not square nor row vector exception.

Eigen::Matrix is not a square matrix nor a row vector

7.33.2 Member Function Documentation

7.33.2.1 type_description()

`std::string qpp::exception::MatrixNotSquareNorRvector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

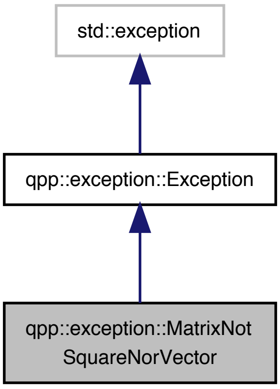
- classes/exception.h

7.34 qpp::exception::MatrixNotSquareNorVector Class Reference

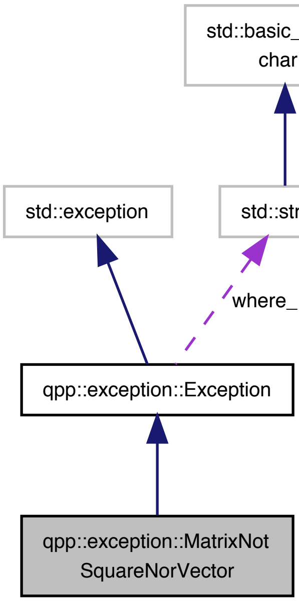
Matrix is not square nor vector exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::MatrixNotSquareNorVector:


std::exception
qpp::exception::Exception
qpp::exception::MatrixNot SquareNorVector

Collaboration diagram for qpp::exception::MatrixNotSquareNorVector:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.34.1 Detailed Description

Matrix is not square nor vector exception.

Eigen::Matrix is not a square matrix nor a row/column vector

7.34.2 Member Function Documentation

7.34.2.1 type_description()

`std::string qpp::exception::MatrixNotSquareNorVector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

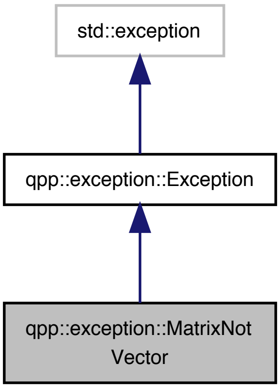
- classes/exception.h

7.35 qpp::exception::MatrixNotVector Class Reference

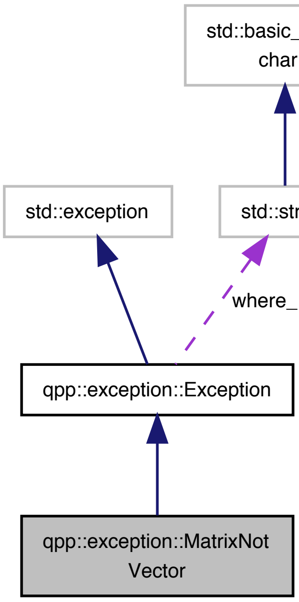
Matrix is not a vector exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::MatrixNotVector:



Collaboration diagram for qpp::exception::MatrixNotVector:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.35.1 Detailed Description

Matrix is not a vector exception.

Eigen::Matrix is not a row or column vector

7.35.2 Member Function Documentation

7.35.2.1 type_description()

`std::string qpp::exception::MatrixNotVector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

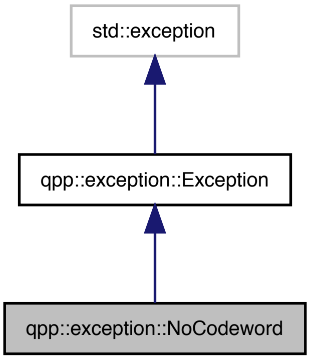
- classes/exception.h

7.36 qpp::exception::NoCodeword Class Reference

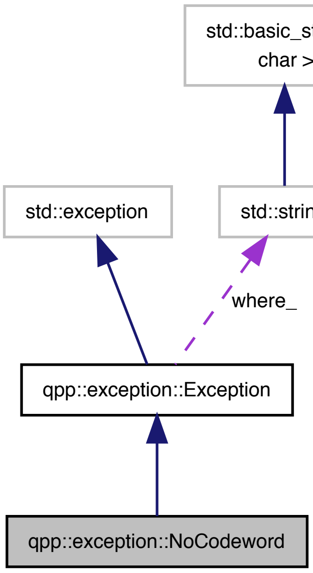
Codeword does not exist exception.

```
#include <classes/exception.h>
```

Inheritance diagram for qpp::exception::NoCodeword:



Collaboration diagram for qpp::exception::NoCodeword:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::NoCodeword

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.36.1 Detailed Description

Codeword does not exist exception.

Codeword does not exist, thrown when calling qpp::Codes::codeword() with an invalid index

7.36.2 Member Function Documentation

7.36.2.1 type_description()

`std::string qpp::exception::NoCodeword::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

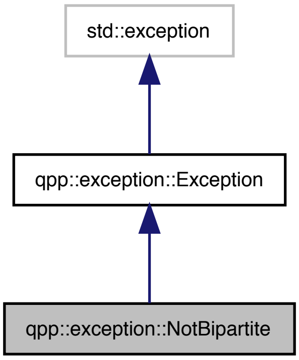
- classes/exception.h

7.37 qpp::exception::NotBipartite Class Reference

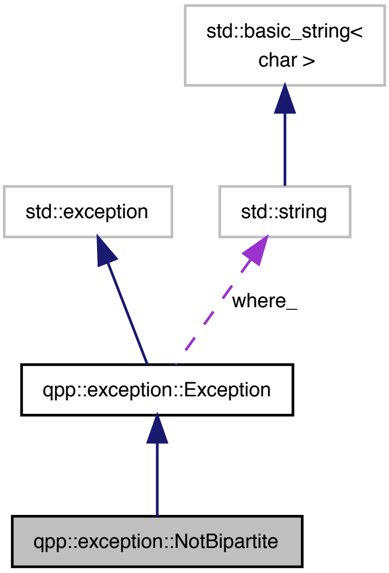
Not bi-partite exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::NotBipartite:



Collaboration diagram for qpp::exception::NotBipartite:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::NotBipartite

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.37.1 Detailed Description

Not bi-partite exception.

std::vector<idx> of dimensions has size different from 2

7.37.2 Member Function Documentation

7.37.2.1 type_description()

`std::string qpp::exception::NotBipartite::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

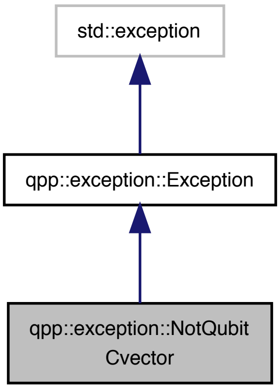
- classes/exception.h

7.38 qpp::exception::NotQubitCvector Class Reference

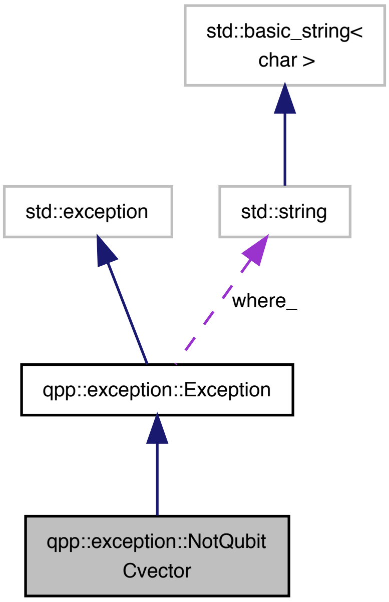
Column vector is not 2 x 1 exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::NotQubitCvector:



Collaboration diagram for qpp::exception::NotQubitCvector:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::NotQubit Cvector

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.38.1 Detailed Description

Column vector is not 2 x 1 exception.

Eigen::Matrix is not 2 x 1

7.38.2 Member Function Documentation

7.38.2.1 type_description()

`std::string qpp::exception::NotQubitCvector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

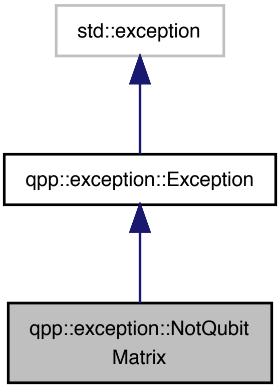
- classes/exception.h

7.39 qpp::exception::NotQubitMatrix Class Reference

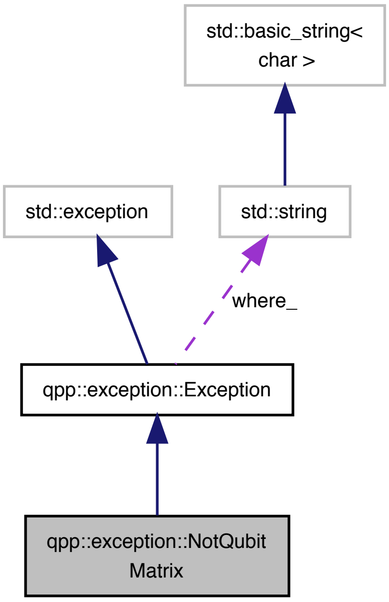
Matrix is not 2 x 2 exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::NotQubitMatrix:



Collaboration diagram for qpp::exception::NotQubitMatrix:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.39.1 Detailed Description

Matrix is not 2 x 2 exception.

Eigen::Matrix is not 2 x 2

7.39.2 Member Function Documentation

7.39.2.1 type_description()

`std::string qpp::exception::NotQubitMatrix::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

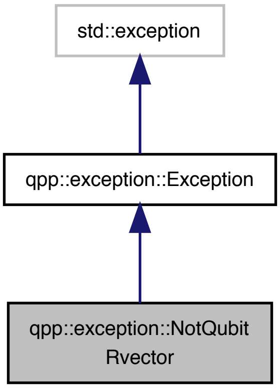
- classes/exception.h

7.40 qpp::exception::NotQubitRvector Class Reference

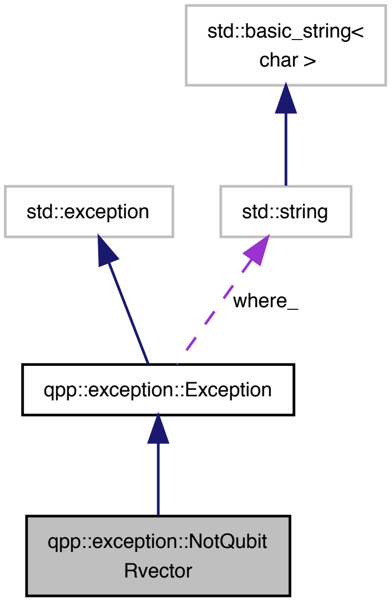
Row vector is not 1 x 2 exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::NotQubitRvector:



Collaboration diagram for qpp::exception::NotQubitRvector:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.40.1 Detailed Description

Row vector is not 1 x 2 exception.

Eigen::Matrix is not 1 x 2

7.40.2 Member Function Documentation

7.40.2.1 type_description()

```
std::string qpp::exception::NotQubitRvector::type_description ( ) const  [inline], [override],
[virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

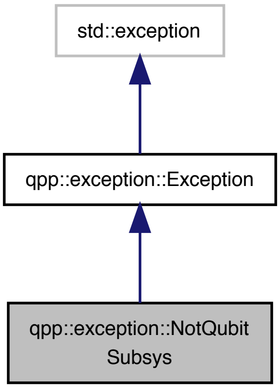
- classes/exception.h

7.41 qpp::exception::NotQubitSubsys Class Reference

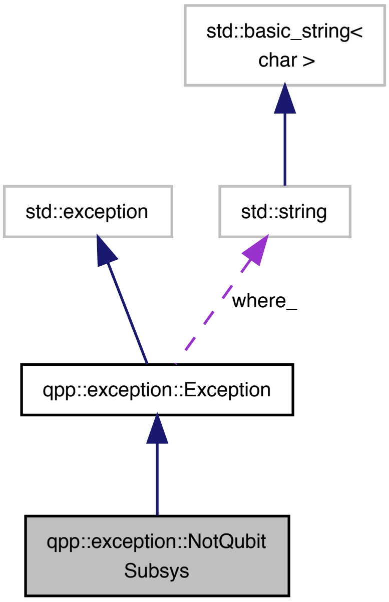
Subsystems are not qubits exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::NotQubitSubsys:



Collaboration diagram for qpp::exception::NotQubitSubsys:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.41.1 Detailed Description

Subsystems are not qubits exception.

Subsystems are not 2-dimensional (qubits)

7.41.2 Member Function Documentation

7.41.2.1 type_description()

`std::string qpp::exception::NotQubitSubsys::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

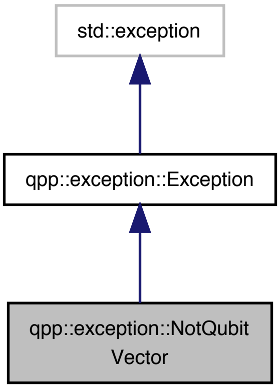
- classes/exception.h

7.42 qpp::exception::NotQubitVector Class Reference

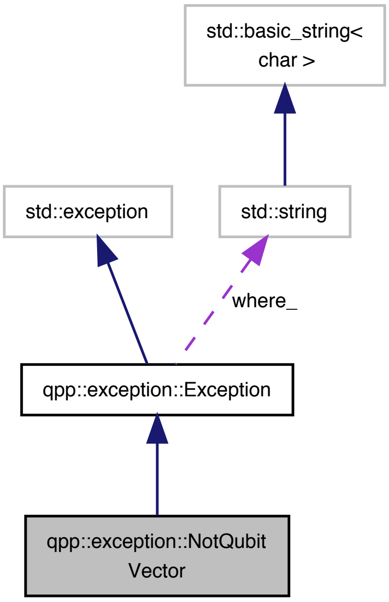
Vector is not 2 x 1 nor 1 x 2 exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::NotQubitVector:



Collaboration diagram for qpp::exception::NotQubitVector:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.42.1 Detailed Description

Vector is not 2 x 1 nor 1 x 2 exception.

Eigen::Matrix is not 2 x 1 nor 1 x 2

7.42.2 Member Function Documentation

7.42.2.1 type_description()

`std::string qpp::exception::NotQubitVector::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

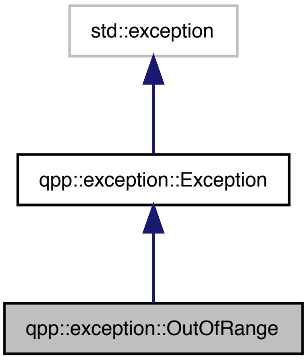
- classes/exception.h

7.43 qpp::exception::OutOfRange Class Reference

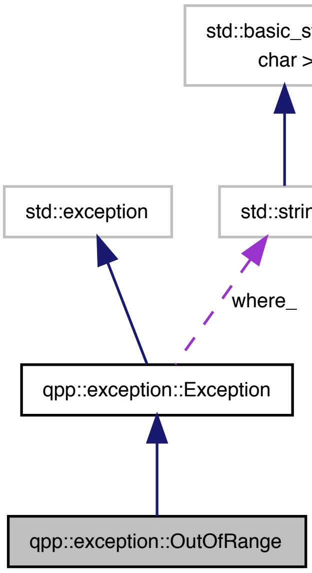
Parameter out of range exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::OutOfRange:



Collaboration diagram for qpp::exception::OutOfRange:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.43.1 Detailed Description

Parameter out of range exception.

Parameter out of range

7.43.2 Member Function Documentation

7.43.2.1 type_description()

```
std::string qpp::exception::OutOfRange::type_description ( ) const  [inline], [override],
[virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

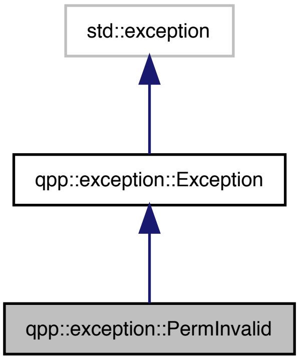
- classes/exception.h

7.44 qpp::exception::PermInvalid Class Reference

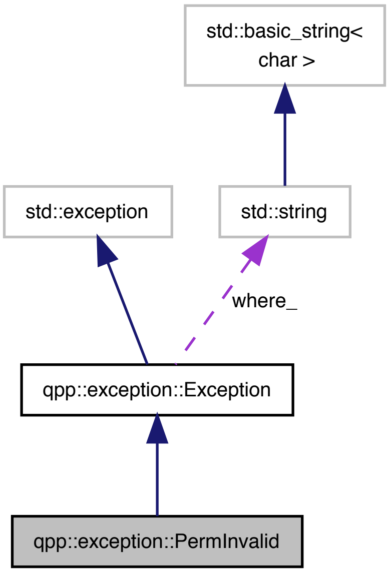
Invalid permutation exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::PermInvalid:



Collaboration diagram for qpp::exception::PermInvalid:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::PermInvalid

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.44.1 Detailed Description

Invalid permutation exception.

std::vector<idx> does note represent a valid permutation

7.44.2 Member Function Documentation

7.44.2.1 type_description()

`std::string qpp::exception::PermInvalid::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

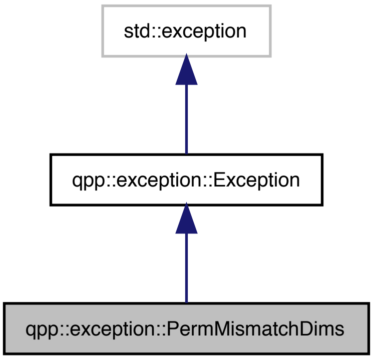
- classes/exception.h

7.45 qpp::exception::PermMismatchDims Class Reference

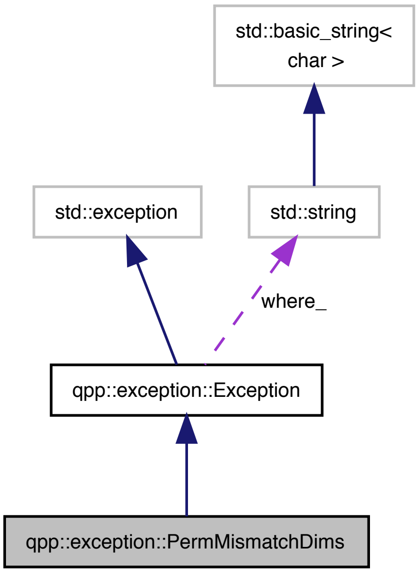
Permutation mismatch dimensions exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::PermMismatchDims:



Collaboration diagram for qpp::exception::PermMismatchDims:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::PermMismatchDims

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.45.1 Detailed Description

Permutation mismatch dimensions exception.

Size of the std::vector<idx> representing the permutation is different from the size of the std::vector<idx> of dimensions

7.45.2 Member Function Documentation

7.45.2.1 type_description()

`std::string qpp::exception::PermMismatchDims::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

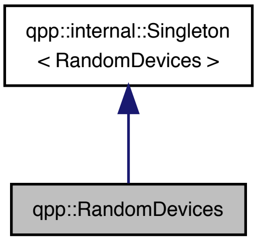
- classes/exception.h

7.46 qpp::RandomDevices Class Reference

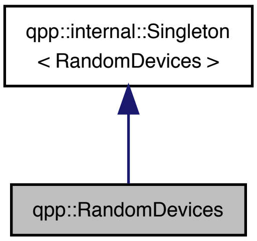
Singleton class that manages the source of randomness in the library.

`#include <classes/random_devices.h>`

Inheritance diagram for qpp::RandomDevices:



Collaboration diagram for qpp::RandomDevices:



Public Member Functions

- std::mt19937 & get_prng ()

  Returns a reference to the internal PRNG object.
- std::istream & load (std::istream &is)

  Loads the state of the PRNG from an input stream.
- std::ostream & save (std::ostream &os) const

  Saves the state of the PRNG to an output stream.

Private Member Functions

- RandomDevices ()

  Initializes and seeds the random number generators.
- ∼RandomDevices ()=default

  Default destructor.

Private Attributes

- std::random_device rd_

  *used to seed std::mt19937 prng_*
- std::mt19937 prng_

  Mersenne twister random number generator.

Friends

- class internal::Singleton< RandomDevices >

Additional Inherited Members

7.46.1 Detailed Description

Singleton class that manages the source of randomness in the library.

Consists of a wrapper around an std::mt19937 Mersenne twister random number generator engine and an std::random_device engine. The latter is used to seed the Mersenne twister.

Warning

> This class DOES NOT seed the standard C number generator used by Eigen::Matrix::Random(), since it is not thread safe. Do not use Eigen::Matrix::Random() or functions that depend on the C style random number engine, but use qpp::rand() instead!

7.46.2 Constructor & Destructor Documentation

7.46.2.1 RandomDevices()

`qpp::RandomDevices::RandomDevices ( ) [inline], [private]`

Initializes and seeds the random number generators.

7.46.2.2 ∼RandomDevices()

`qpp::RandomDevices::∼RandomDevices ( ) [private], [default]`

Default destructor.

7.46.3 Member Function Documentation

7.46.3.1 get_prng()

`std::mt19937& qpp::RandomDevices::get_prng ( ) [inline]`

Returns a reference to the internal PRNG object.

Returns

> Reference to the internal PRNG object

7.46.3.2 load()

```
std::istream& qpp::RandomDevices::load (
            std::istream & is ) [inline]
```

Loads the state of the PRNG from an input stream.

Parameters

is	Input stream

Returns

The input stream

7.46.3.3 save()

```
std::ostream& qpp::RandomDevices::save (
            std::ostream & os ) const  [inline]
```

Saves the state of the PRNG to an output stream.

Parameters

os	Output stream

Returns

The output stream

7.46.4 Friends And Related Function Documentation

7.46.4.1 internal::Singleton< RandomDevices >

```
friend class internal::Singleton< RandomDevices >  [friend]
```

7.46.5 Member Data Documentation

7.46.5.1 prng_

```
std::mt19937 qpp::RandomDevices::prng_  [private]
```

Mersenne twister random number generator.

7.46.5.2 rd_

```
std::random_device qpp::RandomDevices::rd_  [private]
```

used to seed std::mt19937 prng_

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

- classes/random_devices.h

7.47 qpp::internal::Singleton< T > Class Template Reference

Singleton policy class, used internally to implement the singleton pattern via CRTP (Curiously recurring template pattern)

`#include <internal/classes/singleton.h>`

Static Public Member Functions

- static T & get_instance () noexcept(std::is_nothrow_constructible< T >::value)
- static T & get_thread_local_instance () noexcept(std::is_nothrow_constructible< T >::value)

Protected Member Functions

- Singleton () noexcept=default
- Singleton (const Singleton &)=delete
- Singleton & operator= (const Singleton &)=delete
- virtual ∼Singleton ()=default

7.47.1 Detailed Description

template<typename T>
class qpp::internal::Singleton< T >

Singleton policy class, used internally to implement the singleton pattern via CRTP (Curiously recurring template pattern)

To implement a singleton, derive your class from qpp::internal::Singleton, make qpp::internal::Singleton a friend of your class, then declare the constructor and destructor of your class as private. To get an instance, use the static member function qpp::internal::Singleton::get_instance() (qpp::internal::Singleton::get_thread_local_instance()), which returns a reference (thread_local reference) to your newly created singleton (thread-safe in C++11).

Example:

```
class MySingleton: public qpp::internal::Singleton<MySingleton>
{
       friend class qpp::internal::Singleton<MySingleton>;
public:
    // Declare all public members here
private:
    MySingleton()
    {
        // Implement the constructor here
    }
    ~MySingleton()
    {
        // Implement the destructor here
    }
};

MySingleton& mySingleton = MySingleton::get_instance(); // Get an instance
thread_local MySingleton& tls = MySingleton::get_thread_local_instance();
// Get a thread_local instance
```

See also

Code of qpp::Codes, qpp::Gates, qpp::Init, qpp::RandomDevices, qpp::States or qpp.h for real world examples of usage.

7.47.2 Constructor & Destructor Documentation

7.47.2.1 Singleton() [1/2]

```
template<typename T>
qpp::internal::Singleton< T >::Singleton ( )  [protected], [default], [noexcept]
```

7.47.2.2 Singleton() [2/2]

```
template<typename T>
qpp::internal::Singleton< T >::Singleton (
              const Singleton< T > &  )  [protected], [delete]
```

7.47.2.3 ∼Singleton()

```
template<typename T>
virtual qpp::internal::Singleton< T >::∼Singleton ( )  [protected], [virtual], [default]
```

7.47.3 Member Function Documentation

7.47.3.1 get_instance()

```
template<typename T>
static T& qpp::internal::Singleton< T >::get_instance ( )  [inline], [static], [noexcept]
```

7.47.3.2 get_thread_local_instance()

```
template<typename T>
static T& qpp::internal::Singleton< T >::get_thread_local_instance ( )  [inline], [static],
[noexcept]
```

7.47.3.3 operator=()

```
template<typename T>
Singleton& qpp::internal::Singleton< T >::operator= (
             const Singleton< T > &  )  [protected], [delete]
```

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

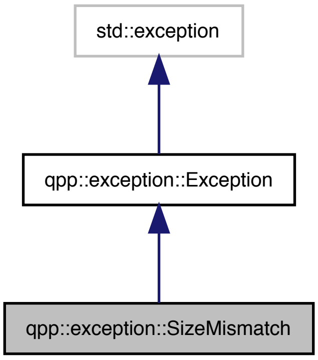
- internal/classes/singleton.h

7.48 qpp::exception::SizeMismatch Class Reference

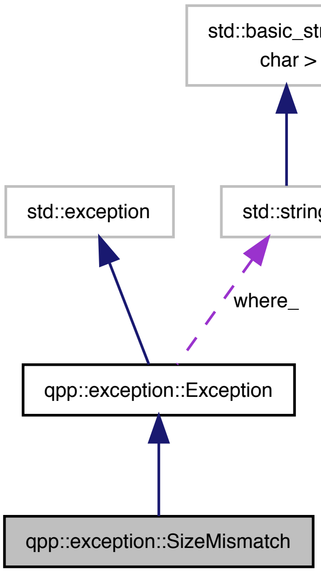
Size mismatch exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::SizeMismatch:



Collaboration diagram for qpp::exception::SizeMismatch:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::SizeMismatch

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.48.1 Detailed Description

Size mismatch exception.

Sizes do not match

7.48.2 Member Function Documentation

7.48.2.1 type_description()

`std::string qpp::exception::SizeMismatch::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

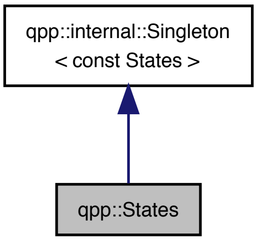
- classes/exception.h

7.49 qpp::States Class Reference

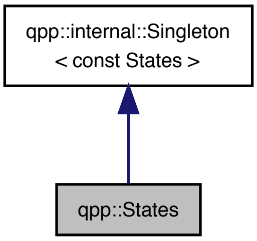
const Singleton class that implements most commonly used states

`#include <classes/states.h>`

Inheritance diagram for qpp::States:



Collaboration diagram for qpp::States:



Public Member Functions

- ket mes (idx d=2) const

  Maximally entangled state of 2 qudits.
- ket zero (idx n, idx d=2) const

  Zero state of n qudits.
- ket one (idx n, idx d=2) const

  One state of n qudits.
- ket jn (idx j, idx n, idx d=2) const

  $|j\rangle^{\otimes n}$ *state of n qudits*
- ket plus (idx n) const

  Plus state of n qubits.
- ket minus (idx n) const

  Minus state of n qubits.

Public Attributes

- ket x0 {ket::Zero(2)}

  Pauli Sigma-X 0-eigenstate |+>
- ket x1 {ket::Zero(2)}

  Pauli Sigma-X 1-eigenstate |->
- ket y0 {ket::Zero(2)}

  Pauli Sigma-Y 0-eigenstate |y+>
- ket y1 {ket::Zero(2)}

  Pauli Sigma-Y 1-eigenstate |y->
- ket z0 {ket::Zero(2)}

  Pauli Sigma-Z 0-eigenstate |0>
- ket z1 {ket::Zero(2)}

  Pauli Sigma-Z 1-eigenstate |1>
- cmat px0 {cmat::Zero(2, 2)}

  Projector onto the Pauli Sigma-X 0-eigenstate |+><+|.
- cmat px1 {cmat::Zero(2, 2)}

  Projector onto the Pauli Sigma-X 1-eigenstate |-><-|.
- cmat py0 {cmat::Zero(2, 2)}

  Projector onto the Pauli Sigma-Y 0-eigenstate |y+><y+|.
- cmat py1 {cmat::Zero(2, 2)}

  Projector onto the Pauli Sigma-Y 1-eigenstate |y-><y-|.
- cmat pz0 {cmat::Zero(2, 2)}

  Projector onto the Pauli Sigma-Z 0-eigenstate |0><0|.
- cmat pz1 {cmat::Zero(2, 2)}

  Projector onto the Pauli Sigma-Z 1-eigenstate |1><1|.
- ket b00 {ket::Zero(4)}

  Bell-00 state (following the convention in Nielsen and Chuang)
- ket b01 {ket::Zero(4)}

  Bell-01 state (following the convention in Nielsen and Chuang)
- ket b10 {ket::Zero(4)}

  Bell-10 state (following the convention in Nielsen and Chuang)
- ket b11 {ket::Zero(4)}

  Bell-11 state (following the convention in Nielsen and Chuang)
- cmat pb00 {cmat::Zero(4, 4)}

  Projector onto the Bell-00 state.
- cmat pb01 {cmat::Zero(4, 4)}

  Projector onto the Bell-01 state.
- cmat pb10 {cmat::Zero(4, 4)}

  Projector onto the Bell-10 state.
- cmat pb11 {cmat::Zero(4, 4)}

  Projector onto the Bell-11 state.
- ket GHZ {ket::Zero(8)}

  GHZ state.
- ket W {ket::Zero(8)}

  W state.
- cmat pGHZ {cmat::Zero(8, 8)}

  Projector onto the GHZ state.
- cmat pW {cmat::Zero(8, 8)}

  Projector onto the W state.

Private Member Functions

- States ()
- ∼States ()=default

  Default destructor.

Friends

- class internal::Singleton< const States >

Additional Inherited Members

7.49.1 Detailed Description

const Singleton class that implements most commonly used states

7.49.2 Constructor & Destructor Documentation

7.49.2.1 States()

```
qpp::States::States ( )  [inline], [private]
```

Initialize the states

7.49.2.2 ∼States()

```
qpp::States::∼States ( )  [private], [default]
```

Default destructor.

7.49.3 Member Function Documentation

7.49.3.1 jn()

```
ket qpp::States::jn (
            idx j,
            idx n,
            idx d = 2 ) const  [inline]
```

$|j\rangle^{\otimes n}$ state of *n* qudits

Parameters

j	Non-negative integer
n	Non-negative integer
d	Subsystem dimensions

Returns

$|j\rangle^{\otimes n}$ state of *n* qudits

7.49.3.2 mes()

```
ket qpp::States::mes (
            idx d = 2 ) const  [inline]
```

Maximally entangled state of 2 qudits.

Parameters

d	Subsystem dimensions

Returns

Maximally entangled state $\frac{1}{\sqrt{d}}\sum_{j=0}^{d-1}|jj\rangle$ of 2 qudits

7.49.3.3 minus()

```
ket qpp::States::minus (
            idx n ) const  [inline]
```

Minus state of *n* qubits.

Parameters

n	Non-negative integer

Returns

Minus state $|-\rangle^{\otimes n}$ of *n* qubits

7.49.3.4 one()

```
ket qpp::States::one (
            idx n,
            idx d = 2 ) const  [inline]
```

One state of *n* qudits.

Parameters

n	Non-negative integer
d	Subsystem dimensions

Returns

One state $|1\rangle^{\otimes n}$ of *n* qudits

7.49.3.5 plus()

```
ket qpp::States::plus (
            idx n ) const  [inline]
```

Plus state of *n* qubits.

Parameters

n	Non-negative integer

Returns

Plus state $|+\rangle^{\otimes n}$ of *n* qubits

7.49.3.6 zero()

```
ket qpp::States::zero (
            idx n,
            idx d = 2 ) const  [inline]
```

Zero state of *n* qudits.

Parameters

n	Non-negative integer
d	Subsystem dimensions

Returns

Zero state $|0\rangle^{\otimes n}$ of *n* qudits

7.49.4 Friends And Related Function Documentation

7.49.4.1 internal::Singleton< const States >

```
friend class internal::Singleton< const States >  [friend]
```

7.49.5 Member Data Documentation

7.49.5.1 b00

```
ket qpp::States::b00 {ket::Zero(4)}
```

Bell-00 state (following the convention in Nielsen and Chuang)

7.49.5.2 b01

```
ket qpp::States::b01 {ket::Zero(4)}
```

Bell-01 state (following the convention in Nielsen and Chuang)

7.49.5.3 b10

```
ket qpp::States::b10 {ket::Zero(4)}
```

Bell-10 state (following the convention in Nielsen and Chuang)

7.49.5.4 b11

```
ket qpp::States::b11 {ket::Zero(4)}
```

Bell-11 state (following the convention in Nielsen and Chuang)

7.49.5.5 GHZ

```
ket qpp::States::GHZ {ket::Zero(8)}
```

GHZ state.

7.49.5.6 pb00

```
cmat qpp::States::pb00 {cmat::Zero(4, 4)}
```

Projector onto the Bell-00 state.

7.49.5.7 pb01

```
cmat qpp::States::pb01 {cmat::Zero(4, 4)}
```

Projector onto the Bell-01 state.

7.49.5.8 pb10

```
cmat qpp::States::pb10 {cmat::Zero(4, 4)}
```

Projector onto the Bell-10 state.

7.49.5.9 pb11

```
cmat qpp::States::pb11 {cmat::Zero(4, 4)}
```

Projector onto the Bell-11 state.

7.49.5.10 pGHZ

```
cmat qpp::States::pGHZ {cmat::Zero(8, 8)}
```

Projector onto the GHZ state.

7.49.5.11 pW

```
cmat qpp::States::pW {cmat::Zero(8, 8)}
```

Projector onto the W state.

7.49.5.12 px0

```
cmat qpp::States::px0 {cmat::Zero(2, 2)}
```

Projector onto the Pauli Sigma-X 0-eigenstate |+><+|.

7.49.5.13 px1

```
cmat qpp::States::px1 {cmat::Zero(2, 2)}
```

Projector onto the Pauli Sigma-X 1-eigenstate |-><-|.

7.49.5.14 py0

```
cmat qpp::States::py0 {cmat::Zero(2, 2)}
```

Projector onto the Pauli Sigma-Y 0-eigenstate |y+><y+|.

7.49.5.15 py1

```
cmat qpp::States::py1 {cmat::Zero(2, 2)}
```

Projector onto the Pauli Sigma-Y 1-eigenstate |y-><y-|.

7.49.5.16 pz0

```
cmat qpp::States::pz0 {cmat::Zero(2, 2)}
```

Projector onto the Pauli Sigma-Z 0-eigenstate |0><0|.

7.49.5.17 pz1

```
cmat qpp::States::pz1 {cmat::Zero(2, 2)}
```

Projector onto the Pauli Sigma-Z 1-eigenstate |1><1|.

7.49.5.18 W

```
ket qpp::States::W {ket::Zero(8)}
```

W state.

7.49.5.19 x0

```
ket qpp::States::x0 {ket::Zero(2)}
```

Pauli Sigma-X 0-eigenstate |+>

7.49.5.20 x1

```
ket qpp::States::x1 {ket::Zero(2)}
```

Pauli Sigma-X 1-eigenstate |->

7.49.5.21 y0

```
ket qpp::States::y0 {ket::Zero(2)}
```

Pauli Sigma-Y 0-eigenstate |y+>

7.49.5.22 y1

```
ket qpp::States::y1 {ket::Zero(2)}
```

Pauli Sigma-Y 1-eigenstate |y->

7.49.5.23 z0

```
ket qpp::States::z0 {ket::Zero(2)}
```

Pauli Sigma-Z 0-eigenstate |0>

7.49.5.24 z1

```
ket qpp::States::z1 {ket::Zero(2)}
```

Pauli Sigma-Z 1-eigenstate |1>

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

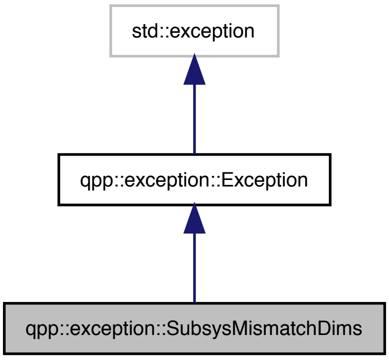
- classes/states.h

7.50 qpp::exception::SubsysMismatchDims Class Reference

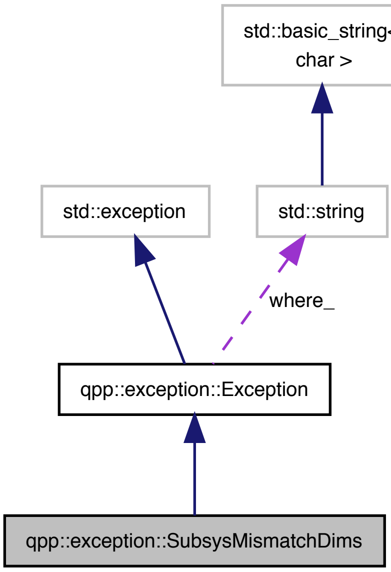
Subsystems mismatch dimensions exception.

```
#include <classes/exception.h>
```

Inheritance diagram for qpp::exception::SubsysMismatchDims:



Collaboration diagram for qpp::exception::SubsysMismatchDims:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::SubsysMismatchDims

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.50.1 Detailed Description

Subsystems mismatch dimensions exception.

std::vector<idx> of subsystem labels has duplicates, or has entries that are larger than the size of the std::vector<idx> of dimensions

7.50.2 Member Function Documentation

7.50.2.1 type_description()

`std::string qpp::exception::SubsysMismatchDims::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

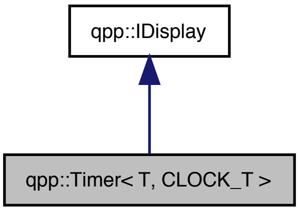
- classes/exception.h

7.51 qpp::Timer< T, CLOCK_T > Class Template Reference

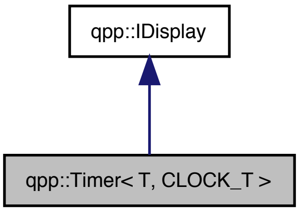
Chronometer.

`#include <classes/timer.h>`

Inheritance diagram for qpp::Timer< T, CLOCK_T >:



Collaboration diagram for qpp::Timer< T, CLOCK_T >:



Public Member Functions

- Timer () noexcept

    Constructs an instance with the current time as the starting point.
- void tic () noexcept

    Resets the chronometer.
- const Timer & toc () noexcept

    Stops the chronometer.
- double tics () const noexcept

    Time passed in the duration specified by T.
- template<typename U = T>
  U get_duration () const noexcept

    Duration specified by U.
- Timer (const Timer &)=default

    Default copy constructor.
- Timer (Timer &&)=default

    Default move constructor.
- Timer & operator= (const Timer &)=default

    Default copy assignment operator.
- Timer & operator= (Timer &&)=default

    Default move assignment operator.
- virtual ∼Timer ()=default

    Default virtual destructor.

Protected Attributes

- CLOCK_T::time_point start_
- CLOCK_T::time_point end_

Private Member Functions

- std::ostream & display (std::ostream &os) const override

    qpp::IDisplay::display() override

7.51.1 Detailed Description

**template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady_clock>
class qpp::Timer< T, CLOCK_T >**

Chronometer.

Template Parameters

T	Tics duration, default is std::chrono::duration<double, 1>, i.e. seconds in double precision
CLOCK_T	Clock's type, default is std::chrono::steady_clock, not affected by wall clock changes during runtime

7.51.2 Constructor & Destructor Documentation

7.51.2.1 Timer() [1/3]

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
qpp::Timer< T, CLOCK_T >::Timer ( )  [inline], [noexcept]
```

Constructs an instance with the current time as the starting point.

7.51.2.2 Timer() [2/3]

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
qpp::Timer< T, CLOCK_T >::Timer (
             const Timer< T, CLOCK_T > &  )  [default]
```

Default copy constructor.

7.51.2.3 Timer() [3/3]

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
qpp::Timer< T, CLOCK_T >::Timer (
             Timer< T, CLOCK_T > &&  )  [default]
```

Default move constructor.

7.51.2.4 ∼Timer()

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
virtual qpp::Timer< T, CLOCK_T >::∼Timer ( )  [virtual], [default]
```

Default virtual destructor.

7.51.3 Member Function Documentation

7.51.3.1 display()

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
std::ostream& qpp::Timer< T, CLOCK_T >::display (
             std::ostream & os ) const  [inline], [override], [private], [virtual]
```

qpp::IDisplay::display() override

Parameters

os	Output stream

Returns

Writes to the output stream the number of tics (specified by T) that passed between the instantiation/reset and invocation of qpp::Timer::toc().

Implements qpp::IDisplay.

7.51.3.2 get_duration()

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
template<typename U = T>
U qpp::Timer< T, CLOCK_T >::get_duration ( ) const  [inline], [noexcept]
```

Duration specified by U.

Template Parameters

U	Duration, default is T, which defaults to std::chrono::duration<double, 1>, i.e. seconds in double precision

Returns

Duration that passed between the instantiation/reset and invocation of qpp::Timer::toc()

7.51.3.3 operator=() [1/2]

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
Timer& qpp::Timer< T, CLOCK_T >::operator= (
             const Timer< T, CLOCK_T > &  )  [default]
```

Default copy assignment operator.

7.51.3.4 operator=() [2/2]

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
Timer& qpp::Timer< T, CLOCK_T >::operator= (
             Timer< T, CLOCK_T > &&  )  [default]
```

Default move assignment operator.

7.51.3.5 tic()

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
void qpp::Timer< T, CLOCK_T >::tic ( )  [inline], [noexcept]
```

Resets the chronometer.

Resets the starting/ending point to the current time

7.51.3.6 tics()

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
double qpp::Timer< T, CLOCK_T >::tics ( ) const  [inline], [noexcept]
```

Time passed in the duration specified by T.

Returns

Number of tics (specified by T) that passed between the instantiation/reset and invocation of qpp::Timer::toc()

7.51.3.7 toc()

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
const Timer& qpp::Timer< T, CLOCK_T >::toc ( )  [inline], [noexcept]
```

Stops the chronometer.

Set the current time as the ending point

Returns

Current instance

7.51.4 Member Data Documentation

7.51.4.1 end_

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
CLOCK_T::time_point qpp::Timer< T, CLOCK_T >::end_  [protected]
```

7.51.4.2 start_

```
template<typename T = std::chrono::duration<double>, typename CLOCK_T = std::chrono::steady↩
_clock>
CLOCK_T::time_point qpp::Timer< T, CLOCK_T >::start_  [protected]
```

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

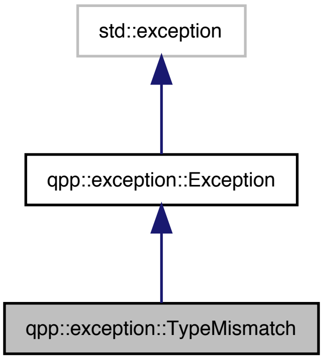
- classes/timer.h

7.52 qpp::exception::TypeMismatch Class Reference

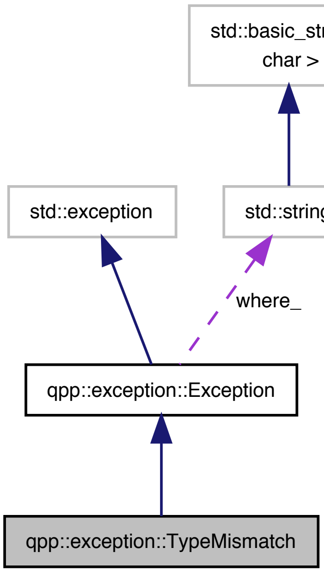
Type mismatch exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::TypeMismatch:



Collaboration diagram for qpp::exception::TypeMismatch:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.52.1 Detailed Description

Type mismatch exception.

Scalar types do not match

7.52.2 Member Function Documentation

7.52.2.1 type_description()

`std::string qpp::exception::TypeMismatch::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

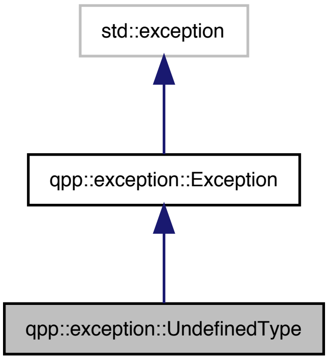
- classes/exception.h

7.53 qpp::exception::UndefinedType Class Reference

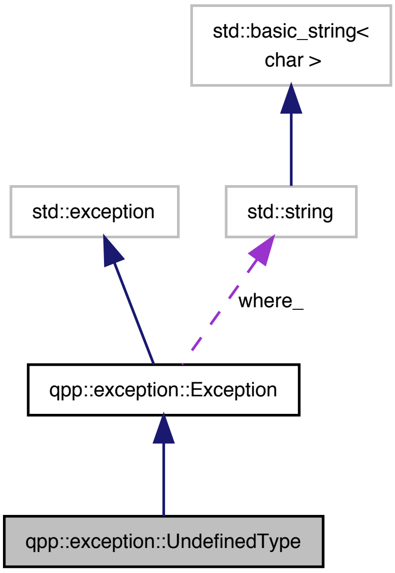
Not defined for this type exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::UndefinedType:



Collaboration diagram for qpp::exception::UndefinedType:


std::basic_string< char >
std::exception
std::string
where_
qpp::exception::Exception
qpp::exception::UndefinedType

Public Member Functions

- std::string type_description () const override

  Exception type description.

7.53.1 Detailed Description

Not defined for this type exception.

Templated specialization is not defined for this type

7.53.2 Member Function Documentation

7.53.2.1 type_description()

```
std::string qpp::exception::UndefinedType::type_description ( ) const  [inline], [override],
[virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

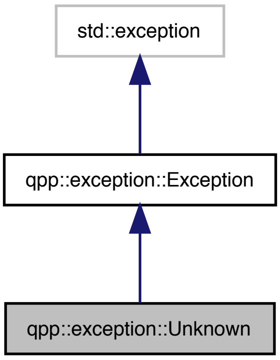
- classes/exception.h

7.54 qpp::exception::Unknown Class Reference

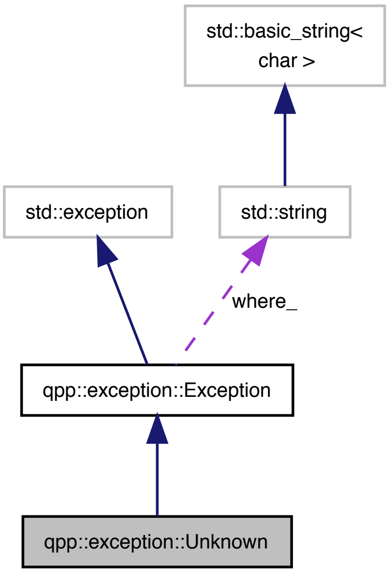
Unknown exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::Unknown:



Collaboration diagram for qpp::exception::Unknown:



Public Member Functions

- std::string type_description () const override

  Exception type description.

7.54.1 Detailed Description

Unknown exception.

Thrown when no other exception is suitable (not recommended, it is better to define another suitable exception type)

7.54.2 Member Function Documentation

7.54.2.1 type_description()

`std::string qpp::exception::Unknown::type_description ( ) const [inline], [override], [virtual]`

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

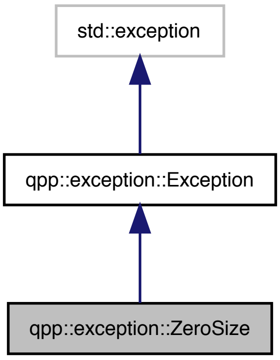
- classes/exception.h

7.55 qpp::exception::ZeroSize Class Reference

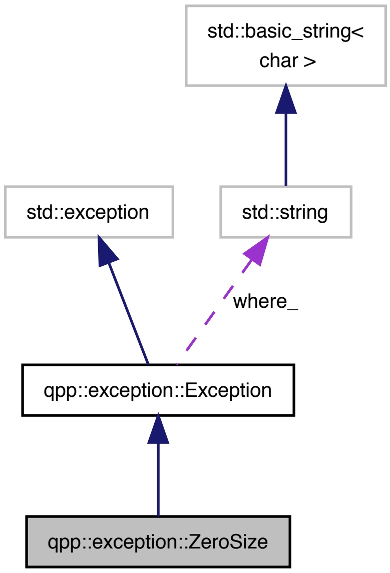
Object has zero size exception.

`#include <classes/exception.h>`

Inheritance diagram for qpp::exception::ZeroSize:



Collaboration diagram for qpp::exception::ZeroSize:



Public Member Functions

- std::string type_description () const override

    Exception type description.

7.55.1 Detailed Description

Object has zero size exception.

Zero sized object, e.g. empty Eigen::Matrix or std::vector with no elements

7.55.2 Member Function Documentation

7.55.2.1 type_description()

```
std::string qpp::exception::ZeroSize::type_description ( ) const  [inline], [override], [virtual]
```

Exception type description.

Returns

Exception type description

Implements qpp::exception::Exception.

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

- classes/exception.h

Chapter 8

File Documentation

8.1 classes/codes.h File Reference

Quantum error correcting codes.

Classes

- class qpp::Codes

    const Singleton class that defines quantum error correcting codes

Namespaces

- qpp

    Quantum++ main namespace.

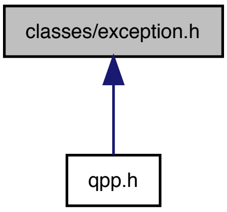
8.1.1 Detailed Description

Quantum error correcting codes.

8.2 classes/exception.h File Reference

Exceptions.

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



Classes

- class qpp::exception::Exception

  Base class for generating Quantum++ custom exceptions.
- class qpp::exception::Unknown

  Unknown exception.
- class qpp::exception::ZeroSize

  Object has zero size exception.
- class qpp::exception::MatrixNotSquare

  Matrix is not square exception.
- class qpp::exception::MatrixNotCvector

  Matrix is not a column vector exception.
- class qpp::exception::MatrixNotRvector

  Matrix is not a row vector exception.
- class qpp::exception::MatrixNotVector

  Matrix is not a vector exception.
- class qpp::exception::MatrixNotSquareNorCvector

  Matrix is not square nor column vector exception.
- class qpp::exception::MatrixNotSquareNorRvector

  Matrix is not square nor row vector exception.
- class qpp::exception::MatrixNotSquareNorVector

  Matrix is not square nor vector exception.
- class qpp::exception::MatrixMismatchSubsys

  Matrix mismatch subsystems exception.
- class qpp::exception::DimsInvalid

  Invalid dimension(s) exception.
- class qpp::exception::DimsNotEqual

  Dimensions not equal exception.
- class qpp::exception::DimsMismatchMatrix

  Dimension(s) mismatch matrix size exception.
- class qpp::exception::DimsMismatchCvector

  Dimension(s) mismatch column vector size exception.
- class qpp::exception::DimsMismatchRvector

  Dimension(s) mismatch row vector size exception.
- class qpp::exception::DimsMismatchVector

  Dimension(s) mismatch vector size exception.
- class qpp::exception::SubsysMismatchDims

  Subsystems mismatch dimensions exception.
- class qpp::exception::PermInvalid

  Invalid permutation exception.
- class qpp::exception::PermMismatchDims

  Permutation mismatch dimensions exception.
- class qpp::exception::NotQubitMatrix

  Matrix is not 2 x 2 exception.
- class qpp::exception::NotQubitCvector

  Column vector is not 2 x 1 exception.
- class qpp::exception::NotQubitRvector

  Row vector is not 1 x 2 exception.
- class qpp::exception::NotQubitVector

  Vector is not 2 x 1 nor 1 x 2 exception.
- class qpp::exception::NotQubitSubsys

  Subsystems are not qubits exception.
- class qpp::exception::NotBipartite

  Not bi-partite exception.
- class qpp::exception::NoCodeword

  Codeword does not exist exception.
- class qpp::exception::OutOfRange

  Parameter out of range exception.
- class qpp::exception::TypeMismatch

  Type mismatch exception.
- class qpp::exception::SizeMismatch

  Size mismatch exception.
- class qpp::exception::UndefinedType

  Not defined for this type exception.
- class qpp::exception::CustomException

  Custom exception.

Namespaces

- qpp

  Quantum++ main namespace.
- qpp::exception

  Quantum++ exception hierarchy namespace.

8.2.1 Detailed Description

Exceptions.

8.3 classes/gates.h File Reference

Quantum gates.

Classes

- class qpp::Gates

  const Singleton class that implements most commonly used gates

Namespaces

- qpp

  Quantum++ main namespace.

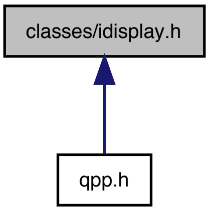
8.3.1 Detailed Description

Quantum gates.

8.4 classes/idisplay.h File Reference

Display interface via the non-virtual interface (NVI)

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



Classes

- class qpp::IDisplay

  Abstract class (interface) that mandates the definition of virtual std::ostream& display(std::ostream& os) const.

Namespaces

- qpp

  Quantum++ main namespace.

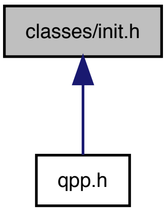
8.4.1 Detailed Description

Display interface via the non-virtual interface (NVI)

8.5 classes/init.h File Reference

Initialization.

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



Classes

- class qpp::Init

  const Singleton class that performs additional initializations/cleanups

Namespaces

- qpp

  Quantum++ main namespace.

8.5.1 Detailed Description

Initialization.

8.6 classes/random_devices.h File Reference

Random devices.

Classes

- class qpp::RandomDevices

  Singleton class that manages the source of randomness in the library.

Namespaces

- qpp

  Quantum++ main namespace.

8.6.1 Detailed Description

Random devices.

8.7 classes/states.h File Reference

Quantum states.

Classes

- class qpp::States

  const Singleton class that implements most commonly used states

Namespaces

- qpp

  Quantum++ main namespace.

8.7.1 Detailed Description

Quantum states.

8.8 classes/timer.h File Reference

Timing.

Classes

- class qpp::Timer< T, CLOCK_T >

  Chronometer.

Namespaces

- qpp

  Quantum++ main namespace.

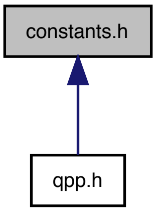
8.8.1 Detailed Description

Timing.

8.9 constants.h File Reference

Constants.

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



Namespaces

- qpp

  Quantum++ main namespace.

Functions

- constexpr cplx qpp::operator"" _i (unsigned long long int x) noexcept

  User-defined literal for complex $i = \sqrt{-1}$ (integer overload)
- constexpr cplx qpp::operator"" _i (long double x) noexcept

  User-defined literal for complex $i = \sqrt{-1}$ (real overload)
- cplx qpp::omega (idx D)

  D-th root of unity.

Variables

- constexpr double qpp::chop = 1e-10

  Used in qpp::disp() for setting to zero numbers that have their absolute value smaller than qpp::chop.
- constexpr double qpp::eps = 1e-12

  Used to decide whether a number or expression in double precision is zero or not.
- constexpr idx qpp::maxn = 64

  Maximum number of allowed qubits/qudits (subsystems)
- constexpr double qpp::pi = 3.141592653589793238462643383279502884

  π
- constexpr double qpp::ee = 2.718281828459045235360287471352662497

  Base of natural logarithm, e.
- constexpr double qpp::infty = std::numeric_limits<double>::max()

  Used to denote infinity in double precision.

8.9.1 Detailed Description

Constants.

8.10 entanglement.h File Reference

Entanglement functions.

Namespaces

- qpp

  Quantum++ main namespace.

Functions

- template<typename Derived >
  dyn_col_vect< double > qpp::schmidtcoeffs (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Schmidt coefficients of the bi-partite pure state A.
- template<typename Derived >
  dyn_col_vect< double > qpp::schmidtcoeffs (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Schmidt coefficients of the bi-partite pure state A.
- template<typename Derived >
  cmat qpp::schmidtA (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Schmidt basis on Alice side.
- template<typename Derived >
  cmat qpp::schmidtA (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Schmidt basis on Alice side.
- template<typename Derived >
  cmat qpp::schmidtB (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Schmidt basis on Bob side.
- template<typename Derived >
  cmat qpp::schmidtB (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Schmidt basis on Bob side.
- template<typename Derived >
  std::vector< double > qpp::schmidtprobs (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Schmidt probabilities of the bi-partite pure state A.
- template<typename Derived >
  std::vector< double > qpp::schmidtprobs (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Schmidt probabilities of the bi-partite pure state A.
- template<typename Derived >
  double qpp::entanglement (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Entanglement of the bi-partite pure state A.
- template<typename Derived >
  double qpp::entanglement (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Entanglement of the bi-partite pure state A.
- template<typename Derived >
  double qpp::gconcurrence (const Eigen::MatrixBase< Derived > &A)

  G-concurrence of the bi-partite pure state A.
- template<typename Derived >
  double qpp::negativity (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Negativity of the bi-partite mixed state A.
- template<typename Derived >
  double qpp::negativity (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Negativity of the bi-partite mixed state A.
- template<typename Derived >
  double qpp::lognegativity (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &dims)

  Logarithmic negativity of the bi-partite mixed state A.
- template<typename Derived >
  double qpp::lognegativity (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Logarithmic negativity of the bi-partite mixed state A.
- template<typename Derived >
  double qpp::concurrence (const Eigen::MatrixBase< Derived > &A)

  Wootters concurrence of the bi-partite qubit mixed state A.

8.10.1 Detailed Description

Entanglement functions.

8.11 entropies.h File Reference

Entropy functions.

Namespaces

- qpp

  Quantum++ main namespace.

Functions

- template< typename Derived >
  double qpp::entropy (const Eigen::MatrixBase< Derived > &A)

  von-Neumann entropy of the density matrix A
- double qpp::entropy (const std::vector< double > &prob)

  Shannon entropy of the probability distribution prob.
- template< typename Derived >
  double qpp::renyi (const Eigen::MatrixBase< Derived > &A, double alpha)

  Renyi- α entropy of the density matrix A, for $\alpha \geq 0$.
- double qpp::renyi (const std::vector< double > &prob, double alpha)

  Renyi- α entropy of the probability distribution prob, for $\alpha \geq 0$.
- template< typename Derived >
  double qpp::tsallis (const Eigen::MatrixBase< Derived > &A, double q)

  Tsallis- q entropy of the density matrix A, for $q \geq 0$.
- double qpp::tsallis (const std::vector< double > &prob, double q)

  Tsallis- q entropy of the probability distribution prob, for $q \geq 0$.
- template< typename Derived >
  double qpp::qmutualinfo (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsysA, const std::vector< idx > &subsysB, const std::vector< idx > &dims)

  Quantum mutual information between 2 subsystems of a composite system.
- template< typename Derived >
  double qpp::qmutualinfo (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsysA, const std::vector< idx > &subsysB, idx d=2)

  Quantum mutual information between 2 subsystems of a composite system.

8.11.1 Detailed Description

Entropy functions.

8.12 experimental/experimental.h File Reference

Experimental/test functions/classes.

```
#include <algorithm>
#include <cassert>
#include <climits>
#include <cstddef>
#include <random>
#include <utility>
#include <vector>
```

Classes

- class qpp::experimental::Dynamic_bitset
- class qpp::experimental::Bit_circuit
- struct qpp::experimental::Bit_circuit::Gate_count

Namespaces

- qpp

  Quantum++ main namespace.
- qpp::experimental

  Experimental/test functions/classes, do not use or modify.

Typedefs

- using idx = std::size_t

8.12.1 Detailed Description

Experimental/test functions/classes.

8.12.2 Typedef Documentation

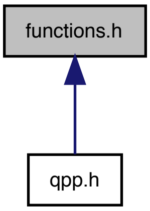
8.12.2.1 idx

```
using idx = std::size_t
```

8.13 functions.h File Reference

Generic quantum computing functions.

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



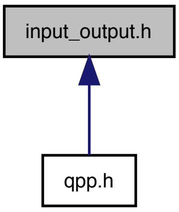
8.13.1 Detailed Description

Generic quantum computing functions.

8.14 input_output.h File Reference

Input/output functions.

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



Namespaces

- qpp

  Quantum++ main namespace.

Functions

- template<typename Derived >
  internal::IOManipEigen qpp::disp (const Eigen::MatrixBase< Derived > &A, double chop=qpp::chop)

  Eigen expression ostream manipulator.
- internal::IOManipEigen qpp::disp (cplx z, double chop=qpp::chop)

  Complex number ostream manipulator.
- template<typename InputIterator >
  internal::IOManipRange< InputIterator > qpp::disp (InputIterator first, InputIterator last, const std::string &separator, const std::string &start="[", const std::string &end="]")

  Range ostream manipulator.
- template<typename Container >
  internal::IOManipRange< typename Container::const_iterator > qpp::disp (const Container &c, const std::string &separator, const std::string &start="[", const std::string &end="]", typename std::enable_if< is_iterable< Container >::value >::type *=nullptr)

  Standard container ostream manipulator. The container must support std::begin(), std::end() and forward iteration.
- template<typename PointerType >
  internal::IOManipPointer< PointerType > qpp::disp (const PointerType *p, idx N, const std::string &separator, const std::string &start="[", const std::string &end="]")

  C-style pointer ostream manipulator.
- template<typename Derived >
  void qpp::save (const Eigen::MatrixBase< Derived > &A, const std::string &fname)

  Saves Eigen expression to a binary file (internal format) in double precision.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::load (const std::string &fname)

  Loads Eigen matrix from a binary file (internal format) in double precision.

8.14.1 Detailed Description

Input/output functions.

8.15 instruments.h File Reference

Measurement functions.

Namespaces

- qpp

  Quantum++ main namespace.

Functions

- template< typename Derived >
  dyn_col_vect< typename Derived::Scalar > qpp::ip (const Eigen::MatrixBase< Derived > &phi, const Eigen::MatrixBase< Derived > &psi, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Generalized inner product.
- template< typename Derived >
  dyn_col_vect< typename Derived::Scalar > qpp::ip (const Eigen::MatrixBase< Derived > &phi, const Eigen::MatrixBase< Derived > &psi, const std::vector< idx > &subsys, idx d=2)

  Generalized inner product.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks)

  Measures the state A using the set of Kraus operators Ks.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const std::initializer_list< cmat > &Ks)

  Measures the state A using the set of Kraus operators Ks.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const cmat &U)

  Measures the state A in the orthonormal basis specified by the unitary matrix U.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Measures the part subsys of the multi-partite state vector or density matrix A using the set of Kraus operators Ks.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const std::initializer_list< cmat > &Ks, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Measures the part subsys of the multi-partite state vector or density matrix A using the set of Kraus operators Ks.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks, const std::vector< idx > &subsys, idx d=2)

  Measures the part subsys of the multi-partite state vector or density matrix A using the set of Kraus operators Ks.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const std::initializer_list< cmat > &Ks, const std::vector< idx > &subsys, idx d=2)

  Measures the part subsys of the multi-partite state vector or density matrix A using the set of Kraus operators Ks.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const cmat &V, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Measures the part subsys of the multi-partite state vector or density matrix A in the orthonormal basis or rank-1 POVM specified by the matrix V.
- template< typename Derived >
  std::tuple< idx, std::vector< double >, std::vector< cmat > > qpp::measure (const Eigen::MatrixBase< Derived > &A, const cmat &V, const std::vector< idx > &subsys, idx d=2)

  Measures the part subsys of the multi-partite state vector or density matrix A in the orthonormal basis or rank-1 POVM specified by the matrix V.
- template< typename Derived >
  std::tuple< std::vector< idx >, double, cmat > qpp::measure_seq (const Eigen::MatrixBase< Derived > &A, std::vector< idx > subsys, std::vector< idx > dims)

  Sequentially measures the part subsys of the multi-partite state vector or density matrix A in the computational basis.

- template<typename Derived >
  std::tuple< std::vector< idx >, double, cmat > qpp::measure_seq (const Eigen::MatrixBase< Derived > &A, std::vector< idx > subsys, idx d=2)

  Sequentially measures the part subsys of the multi-partite state vector or density matrix A in the computational basis.

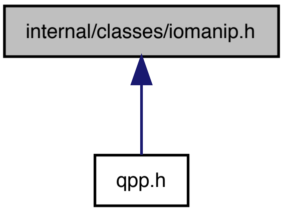
8.15.1 Detailed Description

Measurement functions.

8.16 internal/classes/iomanip.h File Reference

Input/output manipulators.

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



Classes

- class qpp::internal::IOManipRange< InputIterator >
- class qpp::internal::IOManipPointer< PointerType >
- class qpp::internal::IOManipEigen

Namespaces

- qpp

  Quantum++ main namespace.
- qpp::internal

  Internal utility functions, do not use them directly or modify them.

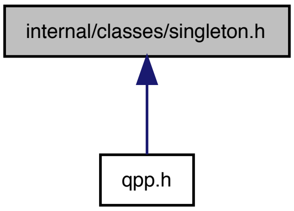
8.16.1 Detailed Description

Input/output manipulators.

8.17 internal/classes/singleton.h File Reference

Singleton pattern via CRTP.

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



Classes

- class qpp::internal::Singleton< T >

  Singleton policy class, used internally to implement the singleton pattern via CRTP (Curiously recurring template pattern)

Namespaces

- qpp

  Quantum++ main namespace.
- qpp::internal

  Internal utility functions, do not use them directly or modify them.

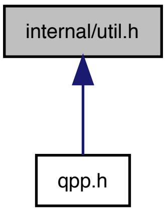
8.17.1 Detailed Description

Singleton pattern via CRTP.

8.18 internal/util.h File Reference

Internal utility functions.

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



Classes

- struct qpp::internal::Display_Impl_

Namespaces

- qpp

  Quantum++ main namespace.
- qpp::internal

  Internal utility functions, do not use them directly or modify them.

Functions

- void qpp::internal::n2multiidx (idx n, idx numdims, const idx ∗const dims, idx ∗result) noexcept
- idx qpp::internal::multiidx2n (const idx ∗const midx, idx numdims, const idx ∗const dims) noexcept
- template< typename Derived >
  bool qpp::internal::check_square_mat (const Eigen::MatrixBase< Derived > &A)
- template< typename Derived >
  bool qpp::internal::check_vector (const Eigen::MatrixBase< Derived > &A)
- template< typename Derived >
  bool qpp::internal::check_rvector (const Eigen::MatrixBase< Derived > &A)
- template< typename Derived >
  bool qpp::internal::check_cvector (const Eigen::MatrixBase< Derived > &A)
- template< typename T >
  bool qpp::internal::check_nonzero_size (const T &x) noexcept
- template< typename T1 , typename T2 >
  bool qpp::internal::check_matching_sizes (const T1 &lhs, const T2 &rhs) noexcept
- bool qpp::internal::check_dims (const std::vector< idx > &dims)
- template< typename Derived >
  bool qpp::internal::check_dims_match_mat (const std::vector< idx > &dims, const Eigen::MatrixBase< Derived > &A)
- template< typename Derived >
  bool qpp::internal::check_dims_match_cvect (const std::vector< idx > &dims, const Eigen::MatrixBase< Derived > &A)

- template<typename Derived >
  bool qpp::internal::check_dims_match_rvect (const std::vector< idx > &dims, const Eigen::MatrixBase< Derived > &A)
- bool qpp::internal::check_eq_dims (const std::vector< idx > &dims, idx dim) noexcept
- bool qpp::internal::check_subsys_match_dims (const std::vector< idx > &subsys, const std::vector< idx > &dims)
- template<typename Derived >
  bool qpp::internal::check_qubit_matrix (const Eigen::MatrixBase< Derived > &A) noexcept
- template<typename Derived >
  bool qpp::internal::check_qubit_cvector (const Eigen::MatrixBase< Derived > &A) noexcept
- template<typename Derived >
  bool qpp::internal::check_qubit_rvector (const Eigen::MatrixBase< Derived > &A) noexcept
- template<typename Derived >
  bool qpp::internal::check_qubit_vector (const Eigen::MatrixBase< Derived > &A) noexcept
- bool qpp::internal::check_perm (const std::vector< idx > &perm)
- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > qpp::internal::kron2 (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)
- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > qpp::internal::dirsum2 (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)
- template<typename T >
  void qpp::internal::variadic_vector_emplace (std::vector< T > &)
- template<typename T , typename First , typename... Args>
  void qpp::internal::variadic_vector_emplace (std::vector< T > &v, First &&first, Args &&... args)
- idx qpp::internal::get_num_subsys (idx sz, idx d)
- idx qpp::internal::get_dim_subsys (idx sz, idx N)

8.18.1 Detailed Description

Internal utility functions.

8.19 MATLAB/matlab.h File Reference

Input/output interfacing with MATLAB.

```
#include "mat.h"
#include "mex.h"
```

Namespaces

- qpp

  Quantum++ main namespace.

Functions

- template<typename Derived >
  std::enable_if< std::is_same< typename Derived::Scalar, cplx >::value, dyn_mat< cplx > >::type qpp::loadMATLAB (const std::string &mat_file, const std::string &var_name)

  Loads a complex Eigen dynamic matrix from a MATLAB .mat file,.
- template<typename Derived >
  std::enable_if<!std::is_same< typename Derived::Scalar, cplx >::value, dyn_mat< typename Derived::Scalar > >::type qpp::loadMATLAB (const std::string &mat_file, const std::string &var_name)

  Loads a non-complex Eigen dynamic matrix from a MATLAB .mat file,.
- template<typename Derived >
  std::enable_if< std::is_same< typename Derived::Scalar, cplx >::value >::type qpp::saveMATLAB (const Eigen::MatrixBase< Derived > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)

  Saves a complex Eigen dynamic matrix to a MATLAB .mat file,.
- template<typename Derived >
  std::enable_if< !std::is_same< typename Derived::Scalar, cplx >::value >::type qpp::saveMATLAB (const Eigen::MatrixBase< Derived > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)

  Saves a non-complex Eigen dynamic matrix to a MATLAB .mat file,.

8.19.1 Detailed Description

Input/output interfacing with MATLAB.

8.20 number_theory.h File Reference

Number theory functions.

Namespaces

- qpp

  Quantum++ main namespace.

Functions

- std::vector< int > qpp::x2contfrac (double x, idx N, idx cut=1e5)

  Simple continued fraction expansion.
- double qpp::contfrac2x (const std::vector< int > &cf, idx N=idx(-1))

  Real representation of a simple continued fraction.
- bigint qpp::gcd (bigint a, bigint b)

  Greatest common divisor of two integers.
- bigint qpp::gcd (const std::vector< bigint > &as)

  Greatest common divisor of a list of integers.
- bigint qpp::lcm (bigint a, bigint b)

  Least common multiple of two integers.
- bigint qpp::lcm (const std::vector< bigint > &as)

  Least common multiple of a list of integers.

- std::vector< idx > qpp::invperm (const std::vector< idx > &perm)

  Inverse permutation.
- std::vector< idx > qpp::compperm (const std::vector< idx > &perm, const std::vector< idx > &sigma)

  Compose permutations.
- std::vector< bigint > qpp::factors (bigint a)

  Prime factor decomposition.
- bigint qpp::modmul (bigint a, bigint b, bigint p)

  Modular multiplication without overflow.
- bigint qpp::modpow (bigint a, bigint n, bigint p)

  Fast integer power modulo p based on the SQUARE-AND-MULTIPLY algorithm.
- std::tuple< bigint, bigint, bigint > qpp::egcd (bigint a, bigint b)

  Extended greatest common divisor of two integers.
- bigint qpp::modinv (bigint a, bigint p)

  Modular inverse of a mod p.
- bool qpp::isprime (bigint p, idx k=80)

  Primality test based on the Miller-Rabin's algorithm.
- bigint qpp::randprime (bigint a, bigint b, idx N=1000)

  Generates a random big prime uniformly distributed in the interval [a, b].

8.20.1 Detailed Description

Number theory functions.

8.21 operations.h File Reference

Quantum operation functions.

Namespaces

- qpp

  Quantum++ main namespace.

Functions

- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > qpp::applyCTRL (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< Derived2 > &A, const std::vector< idx > &ctrl, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Applies the controlled-gate A to the part subsys of the multi-partite state vector or density matrix state.
- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > qpp::applyCTRL (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< Derived2 > &A, const std::vector< idx > &ctrl, const std::vector< idx > &subsys, idx d=2)

  Applies the controlled-gate A to the part subsys of the multi-partite state vector or density matrix state.
- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > qpp::apply (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< Derived2 > &A, const std::vector< idx > &subsys, const std::vector< idx > &dims)

Applies the gate A to the part subsys of the multi-partite state vector or density matrix state.

- template<typename Derived1 , typename Derived2 >
  dyn_mat< typename Derived1::Scalar > qpp::apply (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< Derived2 > &A, const std::vector< idx > &subsys, idx d=2)

  Applies the gate A to the part subsys of the multi-partite state vector or density matrix state.
- template<typename Derived >
  cmat qpp::apply (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks)

  Applies the channel specified by the set of Kraus operators Ks to the density matrix A.
- template<typename Derived >
  cmat qpp::apply (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks, const std↩ ::vector< idx > &subsys, const std::vector< idx > &dims)

  Applies the channel specified by the set of Kraus operators Ks to the part subsys of the multi-partite density matrix A.
- template<typename Derived >
  cmat qpp::apply (const Eigen::MatrixBase< Derived > &A, const std::vector< cmat > &Ks, const std↩ ::vector< idx > &subsys, idx d=2)

  Applies the channel specified by the set of Kraus operators Ks to the part subsys of the multi-partite density matrix A.
- cmat qpp::kraus2super (const std::vector< cmat > &Ks)

  Superoperator matrix.
- cmat qpp::kraus2choi (const std::vector< cmat > &Ks)

  Choi matrix.
- std::vector< cmat > qpp::choi2kraus (const cmat &A)

  Orthogonal Kraus operators from Choi matrix.
- cmat qpp::choi2super (const cmat &A)

  Converts Choi matrix to superoperator matrix.
- cmat qpp::super2choi (const cmat &A)

  Converts superoperator matrix to Choi matrix.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::ptrace1 (const Eigen::MatrixBase< Derived > &A, const std↩ ::vector< idx > &dims)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::ptrace1 (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::ptrace2 (const Eigen::MatrixBase< Derived > &A, const std↩ ::vector< idx > &dims)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::ptrace2 (const Eigen::MatrixBase< Derived > &A, idx d=2)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::ptrace (const Eigen::MatrixBase< Derived > &A, const std↩ ::vector< idx > &subsys, const std::vector< idx > &dims)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::ptrace (const Eigen::MatrixBase< Derived > &A, const std↩ ::vector< idx > &subsys, idx d=2)

  Partial trace.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::ptranspose (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsys, const std::vector< idx > &dims)

  Partial transpose.

- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::ptranspose (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &subsys, idx d=2)

  Partial transpose.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::syspermute (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &perm, const std::vector< idx > &dims)

  Subsystem permutation.
- template<typename Derived >
  dyn_mat< typename Derived::Scalar > qpp::syspermute (const Eigen::MatrixBase< Derived > &A, const std::vector< idx > &perm, idx d=2)

  Subsystem permutation.

8.21.1 Detailed Description

Quantum operation functions.

8.22 qpp.h File Reference

Quantum++ main header file, includes all other necessary headers.

```
#include <algorithm>
#include <cassert>
#include <chrono>
#include <cmath>
#include <complex>
#include <cstdlib>
#include <cstring>
#include <exception>
#include <fstream>
#include <functional>
#include <initializer_list>
#include <iomanip>
#include <iterator>
#include <limits>
#include <memory>
#include <numeric>
#include <ostream>
#include <random>
#include <sstream>
#include <stdexcept>
#include <string>
#include <tuple>
#include <type_traits>
#include <utility>
#include <vector>
#include <Eigen/Dense>
#include <Eigen/SVD>
#include "types.h"
#include "classes/exception.h"
#include "constants.h"
#include "traits.h"
#include "classes/idisplay.h"
```

```
#include "internal/util.h"
#include "internal/classes/iomanip.h"
#include "input_output.h"
#include "internal/classes/singleton.h"
#include "classes/init.h"
#include "functions.h"
```

Namespaces

- qpp

  Quantum++ main namespace.

Macros

- #define QPP_UNUSED_

8.22.1 Detailed Description

Quantum++ main header file, includes all other necessary headers.

8.22.2 Macro Definition Documentation

8.22.2.1 QPP_UNUSED_

```
#define QPP_UNUSED_
```

8.23 random.h File Reference

Randomness-related functions.

Namespaces

- qpp

  Quantum++ main namespace.

Functions

- double qpp::rand (double a, double b)

  Generates a random real number uniformly distributed in the interval [a, b)
- bigint qpp::rand (bigint a, bigint b)

  Generates a random big integer uniformly distributed in the interval [a, b].
- idx qpp::randidx (idx a=std::numeric_limits< idx >::min(), idx b=std::numeric_limits< idx >::max())

  Generates a random index (idx) uniformly distributed in the interval [a, b].
- template<typename Derived >
  Derived qpp::rand (idx rows, idx cols, double a=0, double b=1)

  Generates a random matrix with entries uniformly distributed in the interval [a, b)
- template<>
  dmat qpp::rand (idx rows, idx cols, double a, double b)

  Generates a random real matrix with entries uniformly distributed in the interval [a, b), specialization for double matrices (qpp::dmat)
- template<>
  cmat qpp::rand (idx rows, idx cols, double a, double b)

  Generates a random complex matrix with entries (both real and imaginary) uniformly distributed in the interval [a, b), specialization for complex matrices (qpp::cmat)
- template<typename Derived >
  Derived qpp::randn (idx rows, idx cols, double mean=0, double sigma=1)

  Generates a random matrix with entries normally distributed in N(mean, sigma)
- template<>
  dmat qpp::randn (idx rows, idx cols, double mean, double sigma)

  Generates a random real matrix with entries normally distributed in N(mean, sigma), specialization for double matrices (qpp::dmat)
- template<>
  cmat qpp::randn (idx rows, idx cols, double mean, double sigma)

  Generates a random complex matrix with entries (both real and imaginary) normally distributed in N(mean, sigma), specialization for complex matrices (qpp::cmat)
- double qpp::randn (double mean=0, double sigma=1)

  Generates a random real number (double) normally distributed in N(mean, sigma)
- cmat qpp::randU (idx D=2)

  Generates a random unitary matrix.
- cmat qpp::randV (idx Din, idx Dout)

  Generates a random isometry matrix.
- std::vector< cmat > qpp::randkraus (idx N, idx D=2)

  Generates a set of random Kraus operators.
- cmat qpp::randH (idx D=2)

  Generates a random Hermitian matrix.
- ket qpp::randket (idx D=2)

  Generates a random normalized ket (pure state vector)
- cmat qpp::randrho (idx D=2)

  Generates a random density matrix.
- std::vector< idx > qpp::randperm (idx N)

  Generates a random uniformly distributed permutation.
- std::vector< double > qpp::randprob (idx N)

  Generates a random probability vector uniformly distributed over the probability simplex.

8.23.1 Detailed Description

Randomness-related functions.

8.24 statistics.h File Reference

Statistics functions.

Namespaces

- qpp

  Quantum++ main namespace.

Functions

- std::vector< double > qpp::uniform (idx N)

  Uniform probability distribution vector.
- std::vector< double > qpp::marginalX (const dmat &probXY)

  Marginal distribution.
- std::vector< double > qpp::marginalY (const dmat &probXY)

  Marginal distribution.
- template<typename Container >
  double qpp::avg (const std::vector< double > &prob, const Container &X, typename std::enable_if< is_↩ iterable< Container >::value >::type ∗=nullptr)

  Average.
- template<typename Container >
  double qpp::cov (const dmat &probXY, const Container &X, const Container &Y, typename std::enable_if< is_iterable< Container >::value >::type ∗=nullptr)

  Covariance.
- template<typename Container >
  double qpp::var (const std::vector< double > &prob, const Container &X, typename std::enable_if< is_↩ iterable< Container >::value >::type ∗=nullptr)

  Variance.
- template<typename Container >
  double qpp::sigma (const std::vector< double > &prob, const Container &X, typename std::enable_if< is_↩ iterable< Container >::value >::type ∗=nullptr)

  Standard deviation.
- template<typename Container >
  double qpp::cor (const dmat &probXY, const Container &X, const Container &Y, typename std::enable_if< is_iterable< Container >::value >::type ∗=nullptr)

  Correlation.

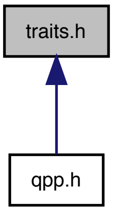
8.24.1 Detailed Description

Statistics functions.

8.25 traits.h File Reference

Type traits.

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



Classes

- struct qpp::make_void< Ts >

  Helper for qpp::to_void<> alias template.
- struct qpp::is_iterable< T, typename >

  Checks whether T is compatible with an STL-like iterable container.
- struct qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().end()), typename T::value_type > >

  Checks whether T is compatible with an STL-like iterable container, specialization for STL-like iterable containers.
- struct qpp::is_matrix_expression< Derived >

  Checks whether the type is an Eigen matrix expression.
- struct qpp::is_complex< T >

  Checks whether the type is a complex type.
- struct qpp::is_complex< std::complex< T > >

  Checks whether the type is a complex number type, specialization for complex types.

Namespaces

- qpp

  Quantum++ main namespace.

Typedefs

- template<typename... Ts>
  using qpp::to_void = typename make_void< Ts... >::type

  Alias template that implements the proposal for void_t.

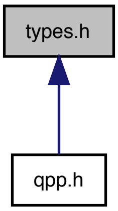
8.25.1 Detailed Description

Type traits.

8.26 types.h File Reference

Type aliases.

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



Namespaces

- qpp

  Quantum++ main namespace.

Typedefs

- using qpp::idx = std::size_t

  Non-negative integer index.
- using qpp::bigint = long long int

  Big integer.
- using qpp::cplx = std::complex< double >

  Complex number in double precision.
- using qpp::ket = Eigen::VectorXcd

  Complex (double precision) dynamic Eigen column vector.
- using qpp::bra = Eigen::RowVectorXcd

  Complex (double precision) dynamic Eigen row vector.
- using qpp::cmat = Eigen::MatrixXcd

  Complex (double precision) dynamic Eigen matrix.
- using qpp::dmat = Eigen::MatrixXd

  Real (double precision) dynamic Eigen matrix.
- template< typename Scalar >
  using qpp::dyn_mat = Eigen::Matrix< Scalar, Eigen::Dynamic, Eigen::Dynamic >

  Dynamic Eigen matrix over the field specified by Scalar.
- template< typename Scalar >
  using qpp::dyn_col_vect = Eigen::Matrix< Scalar, Eigen::Dynamic, 1 >

  Dynamic Eigen column vector over the field specified by Scalar.
- template< typename Scalar >
  using qpp::dyn_row_vect = Eigen::Matrix< Scalar, 1, Eigen::Dynamic >

  Dynamic Eigen row vector over the field specified by Scalar.

8.26.1 Detailed Description

Type aliases.

8.27 /Users/vlad/Dropbox/programming/cpp/qpp/README.md File Reference

Index

/Users/vlad/Dropbox/programming/cpp/qpp/README.md, 245
~Codes
  qpp::Codes, 94
~Gates
  qpp::Gates, 127
~IDisplay
  qpp::IDisplay, 136
~Init
  qpp::Init, 138
~RandomDevices
  qpp::RandomDevices, 189
~Singleton
  qpp::internal::Singleton, 192
~States
  qpp::States, 197
~Timer
  qpp::Timer, 208

A_
  qpp::internal::IOManipEigen, 141
all
  qpp::experimental::Dynamic_bitset, 112
any
  qpp::experimental::Dynamic_bitset, 112
apply
  qpp, 24–26
applyCTRL
  qpp, 26, 27
avg
  qpp, 28

b00
  qpp::States, 200
b01
  qpp::States, 200
b10
  qpp::States, 200
b11
  qpp::States, 200
bigint
  qpp, 22
bra
  qpp, 22

CNOTba
  qpp::Gates, 132
CNOT
  qpp::Gates, 132
  qpp::experimental::Bit_circuit, 90
  qpp::experimental::Bit_circuit::Gate_count, 123
CTRL
  qpp::Gates, 127
check_cvector
  qpp::internal, 84
check_dims
  qpp::internal, 84
check_dims_match_cvect
  qpp::internal, 84
check_dims_match_mat
  qpp::internal, 85
check_dims_match_rvect
  qpp::internal, 85
check_eq_dims
  qpp::internal, 85
check_matching_sizes
  qpp::internal, 85
check_nonzero_size
  qpp::internal, 85
check_perm
  qpp::internal, 85
check_qubit_cvector
  qpp::internal, 86
check_qubit_matrix
  qpp::internal, 86
check_qubit_rvector
  qpp::internal, 86
check_qubit_vector
  qpp::internal, 86
check_rvector
  qpp::internal, 86
check_square_mat
  qpp::internal, 86
check_subsys_match_dims
  qpp::internal, 87
check_vector
  qpp::internal, 87
choi2kraus
  qpp, 28
choi2super
  qpp, 29
chop
  qpp, 80
chop_
  qpp::internal::IOManipEigen, 141
classes/codes.h, 219
classes/exception.h, 219
classes/gates.h, 221
classes/idisplay.h, 222

classes/init.h, 222
classes/random_devices.h, 223
classes/states.h, 223
classes/timer.h, 224
cmat
    qpp, 22
Codes
    qpp::Codes, 94
codeword
    qpp::Codes, 94
compperm
    qpp, 29
concurrence
    qpp, 29
constants.h, 224
contfrac2x
    qpp, 30
cor
    qpp, 30
count
    qpp::experimental::Dynamic_bitset, 113
cov
    qpp, 31
cplx
    qpp, 22
CustomException
    qpp::exception::CustomException, 96
CZ
    qpp::Gates, 132

data
    qpp::experimental::Dynamic_bitset, 113
dirsum2
    qpp::internal, 87
disp
    qpp, 31–33
display
    qpp::IDisplay, 136
    qpp::Timer, 208
    qpp::internal::IOManipEigen, 140
    qpp::internal::IOManipPointer, 143
    qpp::internal::IOManipRange, 146
display_impl_
    qpp::internal::Display_Impl_, 109
dmat
    qpp, 22
dyn_col_vect
    qpp, 22
dyn_mat
    qpp, 23
dyn_row_vect
    qpp, 23
Dynamic_bitset
    qpp::experimental::Dynamic_bitset, 112

ee
    qpp, 80
egcd
    qpp, 34
end_
    qpp::Timer, 210
    qpp::internal::IOManipPointer, 143
    qpp::internal::IOManipRange, 147
entanglement
    qpp, 34, 35
entanglement.h, 225
entropies.h, 227
entropy
    qpp, 35
eps
    qpp, 80
Exception
    qpp::exception::Exception, 122
expandout
    qpp::Gates, 128, 129
experimental.h
    idx, 228
experimental/experimental.h, 228

FRED
    qpp::Gates, 132
    qpp::experimental::Bit_circuit, 91
    qpp::experimental::Bit_circuit::Gate_count, 123
factors
    qpp, 36
Fd
    qpp::Gates, 129
first_
    qpp::internal::IOManipRange, 147
flip
    qpp::experimental::Dynamic_bitset, 113
functions.h, 229

GHZ
    qpp::States, 200
gate_count
    qpp::experimental::Bit_circuit, 92
Gates
    qpp::Gates, 126
gcd
    qpp, 36, 37
gconcurrence
    qpp, 37
get
    qpp::experimental::Dynamic_bitset, 114
get_dim_subsys
    qpp::internal, 87
get_duration
    qpp::Timer, 209
get_instance
    qpp::internal::Singleton, 192
get_num_subsys
    qpp::internal, 87
get_prng
    qpp::RandomDevices, 189
get_thread_local_instance
    qpp::internal::Singleton, 193

H
    qpp::Gates, 132

IDisplay
    qpp::IDisplay, 135, 136
IOManipEigen
    qpp::internal::IOManipEigen, 140
IOManipPointer
    qpp::internal::IOManipPointer, 142, 143
IOManipRange
    qpp::internal::IOManipRange, 146
Id
    qpp::Gates, 130
Id2
    qpp::Gates, 132
idx
    experimental.h, 228
    qpp, 23
index_
    qpp::experimental::Dynamic_bitset, 114
infty
    qpp, 80
Init
    qpp::Init, 138
input_output.h, 229
instruments.h, 230
internal/classes/iomanip.h, 232
internal/classes/singleton.h, 233
internal/util.h, 233
internal::Singleton< const Codes >
    qpp::Codes, 95
internal::Singleton< const Gates >
    qpp::Gates, 131
internal::Singleton< const Init >
    qpp::Init, 139
internal::Singleton< const States >
    qpp::States, 200
internal::Singleton< RandomDevices >
    qpp::RandomDevices, 190
invperm
    qpp, 38
ip
    qpp, 38, 39
isprime
    qpp, 39

jn
    qpp::States, 197

ket
    qpp, 23
kraus2choi
    qpp, 39
kraus2super
    qpp, 40
kron2
    qpp::internal, 87

last_
    qpp::internal::IOManipRange, 147
lcm
    qpp, 40, 41
load
    qpp, 41
    qpp::RandomDevices, 189
loadMATLAB
    qpp, 42
lognegativity
    qpp, 43

MATLAB/matlab.h, 235
marginalX
    qpp, 44
marginalY
    qpp, 44
maxn
    qpp, 81
measure
    qpp, 45–49
measure_seq
    qpp, 50
mes
    qpp::States, 198
minus
    qpp::States, 198
modinv
    qpp, 51
modmul
    qpp, 52
modpow
    qpp, 52
multiidx2n
    qpp::internal, 88

n2multiidx
    qpp::internal, 88
N_
    qpp::experimental::Dynamic_bitset, 119
    qpp::internal::IOManipPointer, 144
NOT
    qpp::experimental::Bit_circuit, 91
    qpp::experimental::Bit_circuit::Gate_count, 123
negativity
    qpp, 53
none
    qpp::experimental::Dynamic_bitset, 114
number_theory.h, 236

offset_
    qpp::experimental::Dynamic_bitset, 114
omega
    qpp, 53
one
    qpp::States, 198
operations.h, 237
operator!=
    qpp::experimental::Dynamic_bitset, 115
operator<<

qpp::IDisplay, 137
qpp::experimental::Dynamic_bitset, 118
operator=
qpp::IDisplay, 136
qpp::Timer, 209
qpp::internal::IOManipPointer, 143
qpp::internal::IOManipRange, 146
qpp::internal::Singleton, 193
operator==
qpp::experimental::Dynamic_bitset, 115
operator"" _i
qpp, 55

p_
qpp::internal::IOManipPointer, 144
pGHZ
qpp::States, 201
pb00
qpp::States, 201
pb01
qpp::States, 201
pb10
qpp::States, 201
pb11
qpp::States, 201
pi
qpp, 81
plus
qpp::States, 199
prng_
qpp::RandomDevices, 190
ptrace
qpp, 55, 56
ptrace1
qpp, 56, 57
ptrace2
qpp, 57, 58
ptranspose
qpp, 58, 59
pW
qpp::States, 201
px0
qpp::States, 202
px1
qpp::States, 202
py0
qpp::States, 202
py1
qpp::States, 202
pz0
qpp::States, 202
pz1
qpp::States, 202

QPP_UNUSED_
qpp.h, 240
qmutualinfo
qpp, 59, 60
qpp, 13
apply, 24–26
applyCTRL, 26, 27
avg, 28
bigint, 22
bra, 22
choi2kraus, 28
choi2super, 29
chop, 80
cmat, 22
compperm, 29
concurrence, 29
contfrac2x, 30
cor, 30
cov, 31
cplx, 22
disp, 31–33
dmat, 22
dyn_col_vect, 22
dyn_mat, 23
dyn_row_vect, 23
ee, 80
egcd, 34
entanglement, 34, 35
entropy, 35
eps, 80
factors, 36
gcd, 36, 37
gconcurrence, 37
idx, 23
infty, 80
invperm, 38
ip, 38, 39
isprime, 39
ket, 23
kraus2choi, 39
kraus2super, 40
lcm, 40, 41
load, 41
loadMATLAB, 42
lognegativity, 43
marginalX, 44
marginalY, 44
maxn, 81
measure, 45–49
measure_seq, 50
modinv, 51
modmul, 52
modpow, 52
negativity, 53
omega, 53
operator"" _i, 55
pi, 81
ptrace, 55, 56
ptrace1, 56, 57
ptrace2, 57, 58
ptranspose, 58, 59
qmutualinfo, 59, 60
rand, 60–62

  randH, 63
  randidx, 63
  randket, 63
  randkraus, 64
  randn, 64–66
  randperm, 66
  randprime, 67
  randprob, 67
  randrho, 67
  randU, 68
  randV, 68
  renyi, 68, 69
  save, 69
  saveMATLAB, 70
  schmidtA, 71
  schmidtB, 72
  schmidtcoeffs, 73
  schmidtprobs, 74
  sigma, 75
  super2choi, 75
  syspermute, 77
  to_void, 24
  tsallis, 78
  uniform, 79
  var, 79
  x2contfrac, 79
qpp.h, 239
  QPP_UNUSED_, 240
qpp::Bit_circuit, 92
qpp::Codes, 92
  ∼Codes, 94
  Codes, 94
  codeword, 94
  internal::Singleton< const Codes >, 95
  Type, 93
qpp::Dynamic_bitset, 119
qpp::Gates, 124
  ∼Gates, 127
  CNOTba, 132
  CNOT, 132
  CTRL, 127
  CZ, 132
  expandout, 128, 129
  FRED, 132
  Fd, 129
  Gates, 126
  H, 132
  Id, 130
  Id2, 132
  internal::Singleton< const Gates >, 131
  Rn, 130
  S, 133
  SWAP, 133
  T, 133
  TOF, 133
  X, 133
  Xd, 131
  Y, 133
  Z, 134
  Zd, 131
qpp::IDisplay, 134
  ∼IDisplay, 136
  display, 136
  IDisplay, 135, 136
  operator<<, 137
  operator=, 136
qpp::Init, 137
  ∼Init, 138
  Init, 138
  internal::Singleton< const Init >, 139
qpp::RandomDevices, 187
  ∼RandomDevices, 189
  get_prng, 189
  internal::Singleton< RandomDevices >, 190
  load, 189
  prng_, 190
  RandomDevices, 189
  rd_, 190
  save, 190
qpp::States, 195
  ∼States, 197
  b00, 200
  b01, 200
  b10, 200
  b11, 200
  GHZ, 200
  internal::Singleton< const States >, 200
  jn, 197
  mes, 198
  minus, 198
  one, 198
  pGHZ, 201
  pb00, 201
  pb01, 201
  pb10, 201
  pb11, 201
  plus, 199
  pW, 201
  px0, 202
  px1, 202
  py0, 202
  py1, 202
  pz0, 202
  pz1, 202
  States, 197
  W, 203
  x0, 203
  x1, 203
  y0, 203
  y1, 203
  z0, 203
  z1, 204
  zero, 199
qpp::Timer
  ∼Timer, 208
  display, 208

  end_, 210
  get_duration, 209
  operator=, 209
  start_, 210
  tic, 209
  tics, 210
  Timer, 208
  toc, 210
qpp::Timer< T, CLOCK_T >, 206
qpp::exception, 81
qpp::exception::CustomException, 95
  CustomException, 96
  type_description, 97
  what_, 97
qpp::exception::DimsInvalid, 98
  type_description, 99
qpp::exception::DimsMismatchCvector, 99
  type_description, 101
qpp::exception::DimsMismatchMatrix, 101
  type_description, 102
qpp::exception::DimsMismatchRvector, 103
  type_description, 104
qpp::exception::DimsMismatchVector, 105
  type_description, 106
qpp::exception::DimsNotEqual, 107
  type_description, 108
qpp::exception::Exception, 120
  Exception, 122
  type_description, 122
  what, 122
  where_, 123
qpp::exception::MatrixMismatchSubsys, 154
  type_description, 155
qpp::exception::MatrixNotCvector, 155
  type_description, 157
qpp::exception::MatrixNotRvector, 157
  type_description, 158
qpp::exception::MatrixNotSquare, 159
  type_description, 160
qpp::exception::MatrixNotSquareNorCvector, 161
  type_description, 162
qpp::exception::MatrixNotSquareNorRvector, 163
  type_description, 164
qpp::exception::MatrixNotSquareNorVector, 165
  type_description, 166
qpp::exception::MatrixNotVector, 167
  type_description, 168
qpp::exception::NoCodeword, 169
  type_description, 170
qpp::exception::NotBipartite, 171
  type_description, 172
qpp::exception::NotQubitCvector, 172
  type_description, 174
qpp::exception::NotQubitMatrix, 174
  type_description, 175
qpp::exception::NotQubitRvector, 176
  type_description, 177
qpp::exception::NotQubitSubsys, 178
  type_description, 179
qpp::exception::NotQubitVector, 180
  type_description, 181
qpp::exception::OutOfRange, 182
  type_description, 183
qpp::exception::PermInvalid, 184
  type_description, 185
qpp::exception::PermMismatchDims, 185
  type_description, 187
qpp::exception::SizeMismatch, 193
  type_description, 194
qpp::exception::SubsysMismatchDims, 204
  type_description, 205
qpp::exception::TypeMismatch, 211
  type_description, 212
qpp::exception::UndefinedType, 213
  type_description, 214
qpp::exception::Unknown, 214
  type_description, 216
qpp::exception::ZeroSize, 216
  type_description, 217
qpp::experimental, 83
qpp::experimental::Bit_circuit, 89
  CNOT, 90
  FRED, 91
  gate_count, 92
  NOT, 91
  reset, 91
  SWAP, 91
  TOF, 91
  X, 91
qpp::experimental::Bit_circuit::Gate_count, 123
  CNOT, 123
  FRED, 123
  NOT, 123
  SWAP, 124
  TOF, 124
  X, 124
qpp::experimental::Dynamic_bitset, 110
  all, 112
  any, 112
  count, 113
  data, 113
  Dynamic_bitset, 112
  flip, 113
  get, 114
  index_, 114
  N_, 119
  none, 114
  offset_, 114
  operator!=, 115
  operator<<, 118
  operator==, 115
  rand, 115, 116
  reset, 116
  set, 117
  size, 117
  storage_size, 117

  - storage_size_, 119
  - storage_type, 112
  - to_string, 118
  - v_, 119
  - value_type, 112
- qpp::internal, 83
  - check_cvector, 84
  - check_dims, 84
  - check_dims_match_cvect, 84
  - check_dims_match_mat, 85
  - check_dims_match_rvect, 85
  - check_eq_dims, 85
  - check_matching_sizes, 85
  - check_nonzero_size, 85
  - check_perm, 85
  - check_qubit_cvector, 86
  - check_qubit_matrix, 86
  - check_qubit_rvector, 86
  - check_qubit_vector, 86
  - check_rvector, 86
  - check_square_mat, 86
  - check_subsys_match_dims, 87
  - check_vector, 87
  - dirsum2, 87
  - get_dim_subsys, 87
  - get_num_subsys, 87
  - kron2, 87
  - multiidx2n, 88
  - n2multiidx, 88
  - variadic_vector_emplace, 88
- qpp::internal::Display_Impl_, 109
  - display_impl_, 109
- qpp::internal::IOManipEigen, 139
  - A_, 141
  - chop_, 141
  - display, 140
  - IOManipEigen, 140
- qpp::internal::IOManipPointer
  - display, 143
  - end_, 143
  - IOManipPointer, 142, 143
  - N_, 144
  - operator=, 143
  - p_, 144
  - separator_, 144
  - start_, 144
- qpp::internal::IOManipPointer< PointerType >, 141
- qpp::internal::IOManipRange
  - display, 146
  - end_, 147
  - first_, 147
  - IOManipRange, 146
  - last_, 147
  - operator=, 146
  - separator_, 147
  - start_, 147
- qpp::internal::IOManipRange< InputIterator >, 145
- qpp::internal::Singleton
  - ∼Singleton, 192
  - get_instance, 192
  - get_thread_local_instance, 193
  - operator=, 193
  - Singleton, 192
- qpp::internal::Singleton< T >, 191
- qpp::is_complex< std::complex< T > >, 149
- qpp::is_complex< T >, 148
- qpp::is_iterable< T, to_void< decltype(std::declval< T >().begin()), decltype(std::declval< T >().↩ end()), typename T::value_type > >, 151
- qpp::is_iterable< T, typename >, 150
- qpp::is_matrix_expression< Derived >, 152
- qpp::make_void
  - type, 153
- qpp::make_void< Ts >, 153

- rand
  - qpp, 60–62
  - qpp::experimental::Dynamic_bitset, 115, 116
- randH
  - qpp, 63
- randidx
  - qpp, 63
- randket
  - qpp, 63
- randkraus
  - qpp, 64
- randn
  - qpp, 64–66
- random.h, 240
- RandomDevices
  - qpp::RandomDevices, 189
- randperm
  - qpp, 66
- randprime
  - qpp, 67
- randprob
  - qpp, 67
- randrho
  - qpp, 67
- randU
  - qpp, 68
- randV
  - qpp, 68
- rd_
  - qpp::RandomDevices, 190
- renyi
  - qpp, 68, 69
- reset
  - qpp::experimental::Bit_circuit, 91
  - qpp::experimental::Dynamic_bitset, 116
- Rn
  - qpp::Gates, 130

- S
  - qpp::Gates, 133
- SWAP
  - qpp::Gates, 133

  qpp::experimental::Bit_circuit, 91
  qpp::experimental::Bit_circuit::Gate_count, 124
save
  qpp, 69
  qpp::RandomDevices, 190
saveMATLAB
  qpp, 70
schmidtA
  qpp, 71
schmidtB
  qpp, 72
schmidtcoeffs
  qpp, 73
schmidtprobs
  qpp, 74
separator_
  qpp::internal::IOManipPointer, 144
  qpp::internal::IOManipRange, 147
set
  qpp::experimental::Dynamic_bitset, 117
sigma
  qpp, 75
Singleton
  qpp::internal::Singleton, 192
size
  qpp::experimental::Dynamic_bitset, 117
start_
  qpp::Timer, 210
  qpp::internal::IOManipPointer, 144
  qpp::internal::IOManipRange, 147
States
  qpp::States, 197
statistics.h, 242
storage_size
  qpp::experimental::Dynamic_bitset, 117
storage_size_
  qpp::experimental::Dynamic_bitset, 119
storage_type
  qpp::experimental::Dynamic_bitset, 112
super2choi
  qpp, 75
syspermute
  qpp, 77

T
  qpp::Gates, 133
TOF
  qpp::Gates, 133
  qpp::experimental::Bit_circuit, 91
  qpp::experimental::Bit_circuit::Gate_count, 124
tic
  qpp::Timer, 209
tics
  qpp::Timer, 210
Timer
  qpp::Timer, 208
to_string
  qpp::experimental::Dynamic_bitset, 118
to_void
  qpp, 24
toc
  qpp::Timer, 210
traits.h, 243
tsallis
  qpp, 78
Type
  qpp::Codes, 93
type
  qpp::make_void, 153
type_description
  qpp::exception::CustomException, 97
  qpp::exception::DimsInvalid, 99
  qpp::exception::DimsMismatchCvector, 101
  qpp::exception::DimsMismatchMatrix, 102
  qpp::exception::DimsMismatchRvector, 104
  qpp::exception::DimsMismatchVector, 106
  qpp::exception::DimsNotEqual, 108
  qpp::exception::Exception, 122
  qpp::exception::MatrixMismatchSubsys, 155
  qpp::exception::MatrixNotCvector, 157
  qpp::exception::MatrixNotRvector, 158
  qpp::exception::MatrixNotSquare, 160
  qpp::exception::MatrixNotSquareNorCvector, 162
  qpp::exception::MatrixNotSquareNorRvector, 164
  qpp::exception::MatrixNotSquareNorVector, 166
  qpp::exception::MatrixNotVector, 168
  qpp::exception::NoCodeword, 170
  qpp::exception::NotBipartite, 172
  qpp::exception::NotQubitCvector, 174
  qpp::exception::NotQubitMatrix, 175
  qpp::exception::NotQubitRvector, 177
  qpp::exception::NotQubitSubsys, 179
  qpp::exception::NotQubitVector, 181
  qpp::exception::OutOfRange, 183
  qpp::exception::PermInvalid, 185
  qpp::exception::PermMismatchDims, 187
  qpp::exception::SizeMismatch, 194
  qpp::exception::SubsysMismatchDims, 205
  qpp::exception::TypeMismatch, 212
  qpp::exception::UndefinedType, 214
  qpp::exception::Unknown, 216
  qpp::exception::ZeroSize, 217
types.h, 244

uniform
  qpp, 79

v_
  qpp::experimental::Dynamic_bitset, 119
value_type
  qpp::experimental::Dynamic_bitset, 112
var
  qpp, 79
variadic_vector_emplace
  qpp::internal, 88

W
  qpp::States, 203

what
    qpp::exception::Exception, 122
what_
    qpp::exception::CustomException, 97
where_
    qpp::exception::Exception, 123

X
    qpp::Gates, 133
    qpp::experimental::Bit_circuit, 91
    qpp::experimental::Bit_circuit::Gate_count, 124
x0
    qpp::States, 203
x1
    qpp::States, 203
x2contfrac
    qpp, 79
Xd
    qpp::Gates, 131

Y
    qpp::Gates, 133
y0
    qpp::States, 203
y1
    qpp::States, 203

Z
    qpp::Gates, 134
z0
    qpp::States, 203
z1
    qpp::States, 204
Zd
    qpp::Gates, 131
zero
    qpp::States, 199