

qpp  
0.1

Generated by Doxygen 1.8.7

Thu Oct 23 2014 22:16:17



# Contents

<b>1</b>	<b>quantum++ - A C++11 quantum computing library</b>	<b>1</b>
<b>2</b>	<b>Namespace Index</b>	<b>3</b>
2.1	Namespace List . . . . .	3
<b>3</b>	<b>Hierarchical Index</b>	<b>5</b>
3.1	Class Hierarchy . . . . .	5
<b>4</b>	<b>Class Index</b>	<b>7</b>
4.1	Class List . . . . .	7
<b>5</b>	<b>File Index</b>	<b>9</b>
5.1	File List . . . . .	9
<b>6</b>	<b>Namespace Documentation</b>	<b>11</b>
6.1	qpp Namespace Reference . . . . .	11
6.1.1	Typedef Documentation . . . . .	17
6.1.1.1	bra . . . . .	17
6.1.1.2	cmat . . . . .	17
6.1.1.3	cplx . . . . .	17
6.1.1.4	dmat . . . . .	17
6.1.1.5	DynMat . . . . .	17
6.1.1.6	ket . . . . .	17
6.1.2	Function Documentation . . . . .	17
6.1.2.1	absm . . . . .	17
6.1.2.2	adjoint . . . . .	18
6.1.2.3	anticomm . . . . .	18
6.1.2.4	channel . . . . .	19
6.1.2.5	channel . . . . .	20
6.1.2.6	choi . . . . .	21
6.1.2.7	choi2kraus . . . . .	22
6.1.2.8	comm . . . . .	23
6.1.2.9	compperm . . . . .	24

6.1.2.10	conjugate	25
6.1.2.11	cosm	25
6.1.2.12	cwise	26
6.1.2.13	det	26
6.1.2.14	disp	27
6.1.2.15	disp	27
6.1.2.16	disp	27
6.1.2.17	disp	27
6.1.2.18	displn	28
6.1.2.19	displn	28
6.1.2.20	displn	28
6.1.2.21	displn	29
6.1.2.22	entanglement	29
6.1.2.23	evals	29
6.1.2.24	evects	30
6.1.2.25	expandout	31
6.1.2.26	expm	31
6.1.2.27	funm	32
6.1.2.28	gconcurrence	33
6.1.2.29	grams	33
6.1.2.30	grams	33
6.1.2.31	grams	34
6.1.2.32	hevals	34
6.1.2.33	hevects	35
6.1.2.34	inverse	36
6.1.2.35	invperm	37
6.1.2.36	kron	37
6.1.2.37	kron	38
6.1.2.38	kron	38
6.1.2.39	kron	39
6.1.2.40	kronpow	39
6.1.2.41	load	40
6.1.2.42	loadMATLABmatrix	40
6.1.2.43	loadMATLABmatrix	40
6.1.2.44	loadMATLABmatrix	40
6.1.2.45	logdet	40
6.1.2.46	logm	41
6.1.2.47	mket	41
6.1.2.48	mket	42
6.1.2.49	mket	42

6.1.2.50	multiidx2n	43
6.1.2.51	n2multiidx	43
6.1.2.52	norm	44
6.1.2.53	omega	44
6.1.2.54	operator""_i	45
6.1.2.55	operator""_i	45
6.1.2.56	powm	45
6.1.2.57	prj	46
6.1.2.58	ptrace	46
6.1.2.59	ptrace1	47
6.1.2.60	ptrace2	48
6.1.2.61	ptranspose	49
6.1.2.62	qmutualinfo	51
6.1.2.63	rand	51
6.1.2.64	rand	51
6.1.2.65	rand	52
6.1.2.66	rand	52
6.1.2.67	randH	52
6.1.2.68	randint	52
6.1.2.69	randket	53
6.1.2.70	randkraus	53
6.1.2.71	randn	53
6.1.2.72	randn	53
6.1.2.73	randn	54
6.1.2.74	randn	54
6.1.2.75	randperm	54
6.1.2.76	randrho	55
6.1.2.77	randU	55
6.1.2.78	randV	55
6.1.2.79	renyi	55
6.1.2.80	renyi_inf	56
6.1.2.81	reshape	56
6.1.2.82	save	56
6.1.2.83	saveMATLABmatrix	56
6.1.2.84	saveMATLABmatrix	57
6.1.2.85	saveMATLABmatrix	57
6.1.2.86	schmidtcoeff	57
6.1.2.87	schmidtprob	58
6.1.2.88	schmidtU	58
6.1.2.89	schmidtV	59

6.1.2.90	shannon	59
6.1.2.91	sinm	59
6.1.2.92	spectralpowm	60
6.1.2.93	sqrtn	60
6.1.2.94	sum	61
6.1.2.95	super	61
6.1.2.96	syspermute	62
6.1.2.97	trace	63
6.1.2.98	transpose	64
6.1.2.99	tsallis	65
6.1.3	Variable Documentation	65
6.1.3.1	chop	65
6.1.3.2	ee	65
6.1.3.3	eps	65
6.1.3.4	gt	65
6.1.3.5	maxn	65
6.1.3.6	pi	65
6.1.3.7	rdevs	66
6.1.3.8	st	66
6.2	qpp::internal Namespace Reference	66
6.2.1	Detailed Description	66
6.2.2	Function Documentation	67
6.2.2.1	_check_col_vector	67
6.2.2.2	_check_dims	67
6.2.2.3	_check_dims_match_cvect	67
6.2.2.4	_check_dims_match_mat	67
6.2.2.5	_check_dims_match_rvect	67
6.2.2.6	_check_eq_dims	67
6.2.2.7	_check_nonzero_size	67
6.2.2.8	_check_perm	67
6.2.2.9	_check_row_vector	67
6.2.2.10	_check_square_mat	67
6.2.2.11	_check_subsys_match_dims	67
6.2.2.12	_check_vector	67
6.2.2.13	_kron2	67
6.2.2.14	_multiidx2n	67
6.2.2.15	_n2multiidx	67
6.2.2.16	variadic_vector_emplace	67
6.2.2.17	variadic_vector_emplace	68

<b>7</b>	<b>Class Documentation</b>	<b>69</b>
7.1	qpp::DiscreteDistribution Class Reference	69
7.1.1	Constructor & Destructor Documentation	69
7.1.1.1	DiscreteDistribution	69
7.1.1.2	DiscreteDistribution	69
7.1.1.3	DiscreteDistribution	69
7.1.2	Member Function Documentation	69
7.1.2.1	probabilities	69
7.1.2.2	sample	70
7.1.3	Member Data Documentation	70
7.1.3.1	_d	70
7.2	qpp::DiscreteDistributionAbsSquare Class Reference	70
7.2.1	Constructor & Destructor Documentation	71
7.2.1.1	DiscreteDistributionAbsSquare	71
7.2.1.2	DiscreteDistributionAbsSquare	71
7.2.1.3	DiscreteDistributionAbsSquare	71
7.2.1.4	DiscreteDistributionAbsSquare	71
7.2.2	Member Function Documentation	71
7.2.2.1	cplx2weights	71
7.2.2.2	probabilities	71
7.2.2.3	sample	71
7.2.3	Member Data Documentation	71
7.2.3.1	_d	71
7.3	qpp::Exception Class Reference	71
7.3.1	Member Enumeration Documentation	73
7.3.1.1	Type	73
7.3.2	Constructor & Destructor Documentation	74
7.3.2.1	Exception	74
7.3.2.2	Exception	74
7.3.3	Member Function Documentation	74
7.3.3.1	_construct_exception_msg	74
7.3.3.2	what	74
7.3.4	Member Data Documentation	74
7.3.4.1	_custom	74
7.3.4.2	_msg	74
7.3.4.3	_type	74
7.3.4.4	_where	74
7.4	qpp::Gates Class Reference	74
7.4.1	Constructor & Destructor Documentation	76
7.4.1.1	Gates	76

7.4.2	Member Function Documentation	76
7.4.2.1	apply	77
7.4.2.2	applyCTRL	77
7.4.2.3	CTRL	78
7.4.2.4	Fd	78
7.4.2.5	Id	78
7.4.2.6	Rn	78
7.4.2.7	Xd	79
7.4.2.8	Zd	79
7.4.3	Friends And Related Function Documentation	79
7.4.3.1	Singleton< const Gates >	79
7.4.4	Member Data Documentation	79
7.4.4.1	CNOTab	79
7.4.4.2	CNOTba	79
7.4.4.3	CZ	79
7.4.4.4	FRED	79
7.4.4.5	H	79
7.4.4.6	Id2	79
7.4.4.7	S	79
7.4.4.8	SWAP	79
7.4.4.9	T	79
7.4.4.10	TOF	80
7.4.4.11	X	80
7.4.4.12	Y	80
7.4.4.13	Z	80
7.5	qpp::NormalDistribution Class Reference	80
7.5.1	Constructor & Destructor Documentation	80
7.5.1.1	NormalDistribution	80
7.5.2	Member Function Documentation	80
7.5.2.1	sample	80
7.5.3	Member Data Documentation	80
7.5.3.1	_d	80
7.6	qpp::Qudit Class Reference	81
7.6.1	Constructor & Destructor Documentation	81
7.6.1.1	Qudit	81
7.6.2	Member Function Documentation	81
7.6.2.1	getD	81
7.6.2.2	getRho	81
7.6.2.3	measure	82
7.6.2.4	measure	82



7.6.3	Member Data Documentation . . . . .	82
7.6.3.1	_D . . . . .	82
7.6.3.2	_rho . . . . .	82
7.7	qpp::RandomDevices Class Reference . . . . .	83
7.7.1	Constructor & Destructor Documentation . . . . .	84
7.7.1.1	RandomDevices . . . . .	84
7.7.2	Friends And Related Function Documentation . . . . .	84
7.7.2.1	Singleton< RandomDevices > . . . . .	84
7.7.3	Member Data Documentation . . . . .	84
7.7.3.1	_rd . . . . .	84
7.7.3.2	_rng . . . . .	84
7.8	qpp::Singleton< T > Class Template Reference . . . . .	84
7.8.1	Constructor & Destructor Documentation . . . . .	85
7.8.1.1	Singleton . . . . .	85
7.8.1.2	~Singleton . . . . .	85
7.8.1.3	Singleton . . . . .	85
7.8.2	Member Function Documentation . . . . .	85
7.8.2.1	get_instance . . . . .	85
7.8.2.2	operator= . . . . .	85
7.9	qpp::States Class Reference . . . . .	85
7.9.1	Constructor & Destructor Documentation . . . . .	87
7.9.1.1	States . . . . .	87
7.9.2	Friends And Related Function Documentation . . . . .	87
7.9.2.1	Singleton< const States > . . . . .	87
7.9.3	Member Data Documentation . . . . .	87
7.9.3.1	b00 . . . . .	87
7.9.3.2	b01 . . . . .	87
7.9.3.3	b10 . . . . .	87
7.9.3.4	b11 . . . . .	87
7.9.3.5	GHZ . . . . .	87
7.9.3.6	pb00 . . . . .	87
7.9.3.7	pb01 . . . . .	87
7.9.3.8	pb10 . . . . .	87
7.9.3.9	pb11 . . . . .	87
7.9.3.10	pGHZ . . . . .	87
7.9.3.11	pW . . . . .	87
7.9.3.12	px0 . . . . .	87
7.9.3.13	px1 . . . . .	87
7.9.3.14	py0 . . . . .	87
7.9.3.15	py1 . . . . .	87

7.9.3.16	pz0 . . . . .	87
7.9.3.17	pz1 . . . . .	87
7.9.3.18	W . . . . .	87
7.9.3.19	x0 . . . . .	87
7.9.3.20	x1 . . . . .	87
7.9.3.21	y0 . . . . .	87
7.9.3.22	y1 . . . . .	87
7.9.3.23	z0 . . . . .	88
7.9.3.24	z1 . . . . .	88
7.10	qpp::Timer Class Reference . . . . .	88
7.10.1	Constructor & Destructor Documentation . . . . .	88
7.10.1.1	Timer . . . . .	88
7.10.2	Member Function Documentation . . . . .	88
7.10.2.1	seconds . . . . .	88
7.10.2.2	tic . . . . .	88
7.10.2.3	toc . . . . .	88
7.10.3	Friends And Related Function Documentation . . . . .	88
7.10.3.1	operator<< . . . . .	88
7.10.4	Member Data Documentation . . . . .	88
7.10.4.1	_end . . . . .	88
7.10.4.2	_start . . . . .	88
7.11	qpp::UniformIntDistribution Class Reference . . . . .	89
7.11.1	Constructor & Destructor Documentation . . . . .	89
7.11.1.1	UniformIntDistribution . . . . .	89
7.11.2	Member Function Documentation . . . . .	89
7.11.2.1	sample . . . . .	89
7.11.3	Member Data Documentation . . . . .	89
7.11.3.1	_d . . . . .	89
7.12	qpp::UniformRealDistribution Class Reference . . . . .	89
7.12.1	Constructor & Destructor Documentation . . . . .	90
7.12.1.1	UniformRealDistribution . . . . .	90
7.12.2	Member Function Documentation . . . . .	90
7.12.2.1	sample . . . . .	90
7.12.3	Member Data Documentation . . . . .	90
7.12.3.1	_d . . . . .	90
<b>8</b>	<b>File Documentation</b>	<b>91</b>
8.1	include/channels.h File Reference . . . . .	91
8.2	include/classes/exception.h File Reference . . . . .	92
8.3	include/classes/gates.h File Reference . . . . .	92

8.4	include/classes/qudit.h File Reference	93
8.5	include/classes/randevs.h File Reference	93
8.6	include/classes/singleton.h File Reference	94
8.6.1	Macro Definition Documentation	94
8.6.1.1	CLASS_CONST_SINGLETON	94
8.6.1.2	CLASS_SINGLETON	94
8.7	include/classes/stat.h File Reference	95
8.8	include/classes/states.h File Reference	95
8.9	include/classes/timer.h File Reference	96
8.10	include/constants.h File Reference	96
8.11	include/entanglement.h File Reference	97
8.12	include/entropies.h File Reference	98
8.13	include/functions.h File Reference	99
8.14	include/internal.h File Reference	102
8.15	include/io.h File Reference	103
8.16	include/matlab.h File Reference	104
8.17	include/qpp.h File Reference	105
8.18	include/random.h File Reference	106
8.19	include/types.h File Reference	107
	<b>Index</b>	<b>108</b>



## Chapter 1

# quantum++ - A C++11 quantum computing library

Version

0.1

Author

Vlad Gheorghiu

Date

24 October 2014



## Chapter 2

# Namespace Index

### 2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">qpp</a> . . . . .	<a href="#">11</a>
<a href="#">qpp::internal</a> . . . . .	<a href="#">66</a>





## Chapter 3

# Hierarchical Index

### 3.1 Class Hierarchy

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

qpp::DiscreteDistribution . . . . .	69
qpp::DiscreteDistributionAbsSquare . . . . .	70
exception	
qpp::Exception . . . . .	71
qpp::NormalDistribution . . . . .	80
qpp::Qudit . . . . .	81
qpp::Singleton< T > . . . . .	84
qpp::Gates . . . . .	74
qpp::RandomDevices . . . . .	83
qpp::Singleton< const Gates > . . . . .	84
qpp::Singleton< const States > . . . . .	84
qpp::States . . . . .	85
qpp::Singleton< RandomDevices > . . . . .	84
qpp::Timer . . . . .	88
qpp::UniformIntDistribution . . . . .	89
qpp::UniformRealDistribution . . . . .	89



## Chapter 4

# Class Index

### 4.1 Class List

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

<a href="#">qpp::DiscreteDistribution</a>	69
<a href="#">qpp::DiscreteDistributionAbsSquare</a>	70
<a href="#">qpp::Exception</a>	71
<a href="#">qpp::Gates</a>	74
<a href="#">qpp::NormalDistribution</a>	80
<a href="#">qpp::Qudit</a>	81
<a href="#">qpp::RandomDevices</a>	83
<a href="#">qpp::Singleton&lt; T &gt;</a>	84
<a href="#">qpp::States</a>	85
<a href="#">qpp::Timer</a>	88
<a href="#">qpp::UniformIntDistribution</a>	89
<a href="#">qpp::UniformRealDistribution</a>	89



## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

include/ <a href="#">channels.h</a>	91
include/ <a href="#">constants.h</a>	96
include/ <a href="#">entanglement.h</a>	97
include/ <a href="#">entropies.h</a>	98
include/ <a href="#">functions.h</a>	99
include/ <a href="#">internal.h</a>	102
include/ <a href="#">io.h</a>	103
include/ <a href="#">matlab.h</a>	104
include/ <a href="#">qpp.h</a>	105
include/ <a href="#">random.h</a>	106
include/ <a href="#">types.h</a>	107
include/classes/ <a href="#">exception.h</a>	92
include/classes/ <a href="#">gates.h</a>	92
include/classes/ <a href="#">qudit.h</a>	93
include/classes/ <a href="#">randevs.h</a>	93
include/classes/ <a href="#">singleton.h</a>	94
include/classes/ <a href="#">stat.h</a>	95
include/classes/ <a href="#">states.h</a>	95
include/classes/ <a href="#">timer.h</a>	96



## Chapter 6

# Namespace Documentation

### 6.1 qpp Namespace Reference

#### Namespaces

- [internal](#)

#### Classes

- class [DiscreteDistribution](#)
- class [DiscreteDistributionAbsSquare](#)
- class [Exception](#)
- class [Gates](#)
- class [NormalDistribution](#)
- class [Qudit](#)
- class [RandomDevices](#)
- class [Singleton](#)
- class [States](#)
- class [Timer](#)
- class [UniformIntDistribution](#)
- class [UniformRealDistribution](#)

#### Typedefs

- using [cplx](#) = std::complex< double >  
*Complex number in double precision.*
- using [cmat](#) = Eigen::MatrixXcd  
*Complex (double precision) dynamic Eigen matrix.*
- using [dmat](#) = Eigen::MatrixXd  
*Real (double precision) dynamic Eigen matrix.*
- using [ket](#) = Eigen::Matrix< [cplx](#), Eigen::Dynamic, 1 >  
*Complex (double precision) dynamic Eigen column matrix.*
- using [bra](#) = Eigen::Matrix< [cplx](#), 1, Eigen::Dynamic >  
*Complex (double precision) dynamic Eigen row matrix.*
- template<typename Scalar >  
using [DynMat](#) = Eigen::Matrix< Scalar, Eigen::Dynamic, Eigen::Dynamic >  
*Dynamic Eigen matrix over the field specified by Scalar.*

## Functions

- `cmat super` (const std::vector< `cmat` > &Ks)  
*Superoperator matrix representation.*
- `cmat choi` (const std::vector< `cmat` > &Ks)  
*Choi matrix representation.*
- `std::vector< cmat > choi2kraus` (const `cmat` &A)  
*Extracts orthogonal Kraus operators from Choi matrix.*
- `template<typename Derived >`  
`cmat channel` (const Eigen::MatrixBase< Derived > &rho, const std::vector< `cmat` > &Ks)  
*Applies the channel specified by the set of Kraus operators Ks to the density matrix rho.*
- `template<typename Derived >`  
`cmat channel` (const Eigen::MatrixBase< Derived > &rho, const std::vector< `cmat` > &Ks, const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims)  
*Applies the channel specified by the set of Kraus operators Ks to the part of the density matrix rho specified by subsys.*
- `constexpr std::complex< double > operator""_i` (unsigned long long int x)  
*User-defined literal for complex  $i = \sqrt{-1}$  (integer overload)*
- `constexpr std::complex< double > operator""_i` (long double x)  
*User-defined literal for complex  $i = \sqrt{-1}$  (real overload)*
- `std::complex< double > omega` (std::size\_t D)  
*D-th root of unity.*
- `template<typename Derived >`  
`cmat schmidtcoeff` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`cmat schmidtU` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`cmat schmidtV` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`cmat schmidtprob` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`double entanglement` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`double gconcurrence` (const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double shannon` (const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double renyi` (const double alpha, const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double renyi_inf` (const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double tsallis` (const double alpha, const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`double qmutualinfo` (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims)
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > transpose` (const Eigen::MatrixBase< Derived > &A)  
*Transpose.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > conjugate` (const Eigen::MatrixBase< Derived > &A)  
*Complex conjugate.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > adjoint` (const Eigen::MatrixBase< Derived > &A)  
*Adjoint.*



- template<typename Derived >  
**DynMat**< typename Derived::Scalar > **inverse** (const Eigen::MatrixBase< Derived > &A)  
*Inverse.*
- template<typename Derived >  
 Derived::Scalar **trace** (const Eigen::MatrixBase< Derived > &A)  
*Trace.*
- template<typename Derived >  
 Derived::Scalar **det** (const Eigen::MatrixBase< Derived > &A)  
*Determinant.*
- template<typename Derived >  
 Derived::Scalar **logdet** (const Eigen::MatrixBase< Derived > &A)  
*Logarithm of the determinant.*
- template<typename Derived >  
 Derived::Scalar **sum** (const Eigen::MatrixBase< Derived > &A)  
*Element-wise sum.*
- template<typename Derived >  
 double **norm** (const Eigen::MatrixBase< Derived > &A)  
*Trace norm.*
- template<typename Derived >  
**cmat evals** (const Eigen::MatrixBase< Derived > &A)  
*Eigenvalues.*
- template<typename Derived >  
**cmat evecs** (const Eigen::MatrixBase< Derived > &A)  
*Eigenvectors.*
- template<typename Derived >  
**dmat hevals** (const Eigen::MatrixBase< Derived > &A)  
*Hermitian eigenvalues.*
- template<typename Derived >  
**cmat hevecs** (const Eigen::MatrixBase< Derived > &A)  
*Hermitian eigenvectors.*
- template<typename Derived >  
**cmat funm** (const Eigen::MatrixBase< Derived > &A, **cplx**(\*f)(const **cplx** &))  
*Functional calculus  $f(A)$*
- template<typename Derived >  
**cmat sqrtm** (const Eigen::MatrixBase< Derived > &A)  
*Matrix square root.*
- template<typename Derived >  
**cmat absm** (const Eigen::MatrixBase< Derived > &A)  
*Matrix absolut value.*
- template<typename Derived >  
**cmat expm** (const Eigen::MatrixBase< Derived > &A)  
*Matrix exponential.*
- template<typename Derived >  
**cmat logm** (const Eigen::MatrixBase< Derived > &A)  
*Matrix logarithm.*
- template<typename Derived >  
**cmat sinm** (const Eigen::MatrixBase< Derived > &A)  
*Matrix sin.*
- template<typename Derived >  
**cmat cosm** (const Eigen::MatrixBase< Derived > &A)  
*Matrix cos.*
- template<typename Derived >  
**cmat spectralpowm** (const Eigen::MatrixBase< Derived > &A, const **cplx** z)

*Matrix power.*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > powm (const Eigen::MatrixBase< Derived > &A, std::size_t n)`

*Matrix power.*

- `template<typename OutputScalar , typename Derived >`  
`DynMat< OutputScalar > cwise (const Eigen::MatrixBase< Derived > &A, OutputScalar(*f)(const typename Derived::Scalar &))`

*Functor.*

- `template<typename T >`  
`DynMat< typename T::Scalar > kron (const T &head)`

*Kronecker product (variadic overload)*

- `template<typename T , typename... Args>`  
`DynMat< typename T::Scalar > kron (const T &head, const Args &...tail)`

*Kronecker product (variadic overload)*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > kron (const std::vector< Derived > &As)`

*Kronecker product (std::vector overload)*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > kron (const std::initializer_list< Derived > &As)`

*Kronecker product (std::initializer\_list overload)*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > kronpow (const Eigen::MatrixBase< Derived > &A, std::size_t n)`

*Kronecker power.*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > reshape (const Eigen::MatrixBase< Derived > &A, std::size_t rows, std::size_t cols)`

*Reshape.*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > syspermute (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &perm, const std::vector< std::size_t > &dims)`

*System permutation.*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > ptrace1 (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`

*Partial trace.*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > ptrace2 (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`

*Partial trace.*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > ptrace (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`

*Partial trace.*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > ptranspose (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`

*Partial transpose.*

- `template<typename Derived1 , typename Derived2 >`  
`DynMat< typename Derived1::Scalar > comm (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)`

*Commutator.*

- `template<typename Derived1 , typename Derived2 >`  
`DynMat< typename Derived1::Scalar > anticomm (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)`

- Anti-commutator.*
- template<typename Derived >  
[DynMat](#)< typename Derived::Scalar > [prj](#) (const Eigen::MatrixBase< Derived > &V)
- Projector.*
- template<typename Derived >  
[DynMat](#)< typename Derived::Scalar > [expandout](#) (const Eigen::MatrixBase< Derived > &A, std::size\_t pos, const std::vector< std::size\_t > &dims)
- Expand out.*
- template<typename Derived >  
[DynMat](#)< typename Derived::Scalar > [grams](#) (const std::vector< Derived > &Vs)
- Gram-Schmidt orthogonalization (std::vector overload)*
- template<typename Derived >  
[DynMat](#)< typename Derived::Scalar > [grams](#) (const std::initializer\_list< Derived > &Vs)
- Gram-Schmidt orthogonalization (std::initializer\_list overload)*
- template<typename Derived >  
[DynMat](#)< typename Derived::Scalar > [grams](#) (const Eigen::MatrixBase< Derived > &A)
- Gram-Schmidt orthogonalization (Eigen expression (matrix) overload)*
- std::vector< std::size\_t > [n2multiidx](#) (std::size\_t n, const std::vector< std::size\_t > &dims)
- Non-negative integer index to multi-index.*
- std::size\_t [multiidx2n](#) (const std::vector< std::size\_t > &midx, const std::vector< std::size\_t > &dims)
- Multi-index to non-negative integer index.*
- [ket mket](#) (const std::vector< std::size\_t > &mask)
- Multi-partite qubit ket.*
- [ket mket](#) (const std::vector< std::size\_t > &mask, const std::vector< std::size\_t > &dims)
- Multi-partite qudit ket (different dimensions overload)*
- [ket mket](#) (const std::vector< std::size\_t > &mask, std::size\_t d)
- Multi-partite qudit ket (same dimensions overload)*
- std::vector< std::size\_t > [invperm](#) (const std::vector< std::size\_t > &perm)
- Inverse permutation.*
- std::vector< std::size\_t > [compperm](#) (const std::vector< std::size\_t > &perm, const std::vector< std::size\_t > &sigma)
- Compose permutations.*
- template<typename T >  
void [disp](#) (const T &x, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)
- template<typename T >  
void [displn](#) (const T &x, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)
- template<typename T >  
void [disp](#) (const T \*x, const std::size\_t n, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)
- template<typename T >  
void [displn](#) (const T \*x, const std::size\_t n, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)
- template<typename Derived >  
void [disp](#) (const Eigen::MatrixBase< Derived > &A, double [chop=chop](#), std::ostream &os=std::cout)
- template<typename Derived >  
void [displn](#) (const Eigen::MatrixBase< Derived > &A, double [chop=chop](#), std::ostream &os=std::cout)
- void [disp](#) (const [cplx](#) c, double [chop=chop](#), std::ostream &os=std::cout)
- void [displn](#) (const [cplx](#) c, double [chop=chop](#), std::ostream &os=std::cout)
- template<typename Derived >  
void [save](#) (const Eigen::MatrixBase< Derived > &A, const std::string &fname)
- template<typename Derived >  
[DynMat](#)< typename Derived::Scalar > [load](#) (const std::string &fname)

- `template<typename Derived >`  
Derived `loadMATLABmatrix` (const std::string &mat\_file, const std::string &var\_name)
- `template<>`  
`dmat loadMATLABmatrix` (const std::string &mat\_file, const std::string &var\_name)
- `template<>`  
`cmat loadMATLABmatrix` (const std::string &mat\_file, const std::string &var\_name)
- `template<typename Derived >`  
void `saveMATLABmatrix` (const Eigen::MatrixBase< Derived > &A, const std::string &mat\_file, const std::string &var\_name, const std::string &mode)
- `template<>`  
void `saveMATLABmatrix` (const Eigen::MatrixBase< `dmat` > &A, const std::string &mat\_file, const std::string &var\_name, const std::string &mode)
- `template<>`  
void `saveMATLABmatrix` (const Eigen::MatrixBase< `cmat` > &A, const std::string &mat\_file, const std::string &var\_name, const std::string &mode)
- `template<typename Derived >`  
Derived `rand` (std::size\_t rows, std::size\_t cols, double a=0, double b=1)
- `template<>`  
`dmat rand` (std::size\_t rows, std::size\_t cols, double a, double b)
- `template<>`  
`cmat rand` (std::size\_t rows, std::size\_t cols, double a, double b)
- double `rand` (double a=0, double b=1)
- long long `randint` (long long a, long long b)
- `template<typename Derived >`  
Derived `randn` (std::size\_t rows, std::size\_t cols, double mean=0, double sigma=1)
- `template<>`  
`dmat randn` (std::size\_t rows, std::size\_t cols, double mean, double sigma)
- `template<>`  
`cmat randn` (std::size\_t rows, std::size\_t cols, double mean, double sigma)
- double `randn` (double mean=0, double sigma=1)
- `cmat randU` (std::size\_t D)
- `cmat randV` (std::size\_t Din, std::size\_t Dout)
- std::vector< `cmat` > `randkraus` (std::size\_t n, std::size\_t D)
- `cmat randH` (std::size\_t D)
- `ket randket` (std::size\_t D)
- `cmat randrho` (std::size\_t D)
- std::vector< std::size\_t > `randperm` (std::size\_t n)

## Variables

- constexpr double `chop` = 1e-10  
*Used in `qpp::disp()` and `qpp::displn()` for setting to zero numbers that have their absolute value smaller than `qpp::ct::chop`.*
- constexpr double `eps` = 1e-12  
*Used to decide whether a number or expression in double precision is zero or not.*
- constexpr std::size\_t `maxn` = 64  
*Maximum number of qubits.*
- constexpr double `pi` = 3.141592653589793238462643383279502884  
 $\pi$
- constexpr double `ee` = 2.718281828459045235360287471352662497  
*Base of natural logarithm,  $e$ .*
- `RandomDevices` & `rdevs` = `RandomDevices::get_instance()`  
*`qpp::RandomDevices` Singleton*
- const `Gates` & `gt` = `Gates::get_instance()`

- `qpp::Gates` *const Singleton*
- `const States & st = States::get_instance()`
- `qpp::States` *const Singleton*

## 6.1.1 Typedef Documentation

### 6.1.1.1 `using qpp::bra = typedef Eigen::Matrix<cplx, 1, Eigen::Dynamic>`

Complex (double precision) dynamic Eigen row matrix.

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

Complex (double precision) dynamic Eigen matrix.

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

Complex number in double precision.

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

Real (double precision) dynamic Eigen matrix.

### 6.1.1.5 `template<typename Scalar > using qpp::DynMat = typedef Eigen::Matrix<Scalar, Eigen::Dynamic, Eigen::Dynamic>`

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

Example:

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

### 6.1.1.6 `using qpp::ket = typedef Eigen::Matrix<cplx, Eigen::Dynamic, 1>`

Complex (double precision) dynamic Eigen column matrix.

## 6.1.2 Function Documentation

### 6.1.2.1 `template<typename Derived > cmat qpp::absm ( const Eigen::MatrixBase< Derived > & A )`

Matrix absolut value.

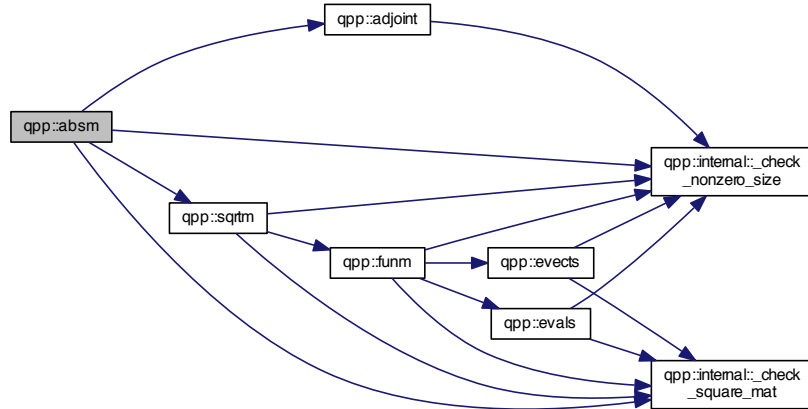
Parameters

<i>A</i>	Eigen expression
----------	------------------

## Returns

Matrix absolut value of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.2 `template<typename Derived > DynMat<typename Derived::Scalar> qpp::adjoint ( const Eigen::MatrixBase< Derived > & A )`

Adjoint.

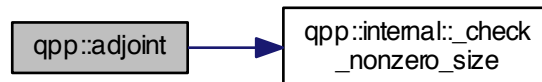
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Adjoint (Hermitian conjugate) of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.3 `template<typename Derived1 , typename Derived2 > DynMat<typename Derived1::Scalar> qpp::anticomm ( const Eigen::MatrixBase< Derived1 > & A, const Eigen::MatrixBase< Derived2 > & B )`

Anti-commutator.

Anti-commutator  $\{A, B\} = AB + BA$

Both  $A$  and  $B$  must be Eigen expressions over the same scalar field

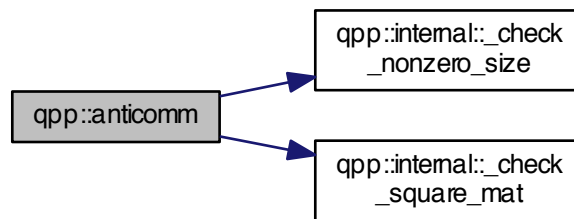
## Parameters

$A$	Eigen expression
$B$	Eigen expression

## Returns

Anti-commutator  $AB + BA$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.4 `template<typename Derived > cmat qpp::channel ( const Eigen::MatrixBase< Derived > & rho, const std::vector< cmat > & Ks )`

Applies the channel specified by the set of Kraus operators  $Ks$  to the density matrix  $\rho$ .

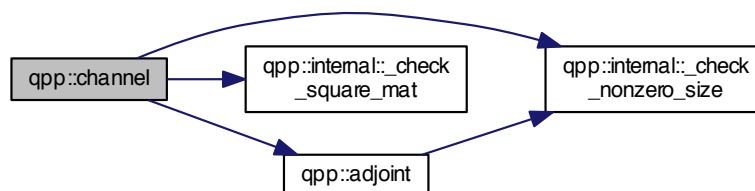
## Parameters

$\rho$	Eigen expression
$Ks$	<code>std::vector</code> of Eigen expressions representing the set of Kraus operators

## Returns

Output density matrix, as a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.5 `template<typename Derived > cmat qpp::channel ( const Eigen::MatrixBase< Derived > & rho, const std::vector< cmat > & Ks, const std::vector< std::size_t > & subsys, const std::vector< std::size_t > & dims )`

Applies the channel specified by the set of Kraus operators *Ks* to the part of the density matrix *rho* specified by *subsys*.



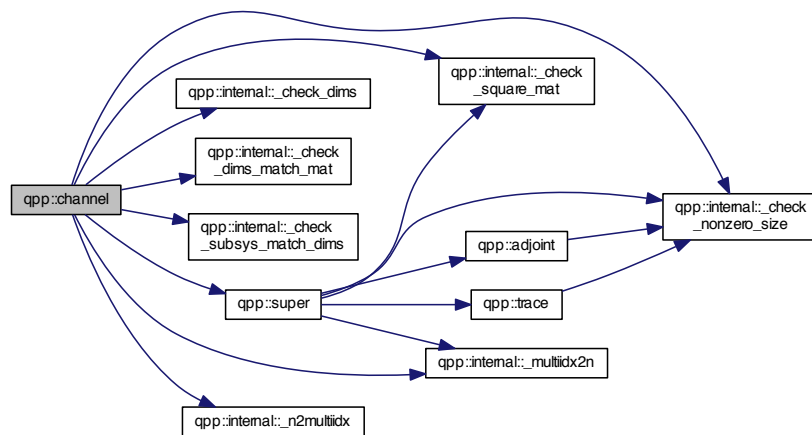
## Parameters

<i>rho</i>	Eigen expression
<i>Ks</i>	std::vector of Eigen expressions representing the set of Kraus operators
<i>subsys</i>	Subsystems' indexes
<i>dims</i>	Dimensions of the multi-partite system

## Returns

Output density matrix, as a dynamic matrix over the complex field

Here is the call graph for this function:



### 6.1.2.6 cmat qpp::choi ( const std::vector< cmat > & Ks )

Choi matrix representation.

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  $S_{ab,mn} = C_{ma,nb}$

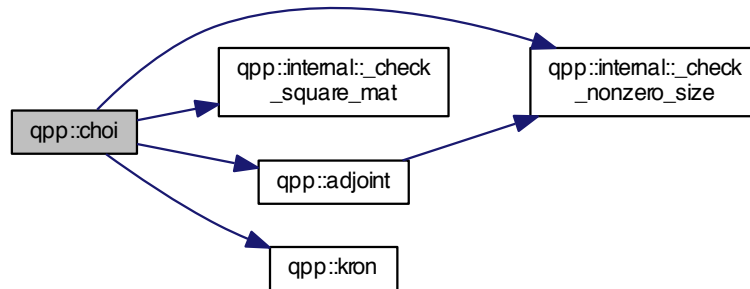
## Parameters

<i>Ks</i>	std::vector of Eigen expressions representing the set of Kraus operators
-----------	--

**Returns**

Choi matrix representation, as a dynamic matrix over the complex field

Here is the call graph for this function:



### 6.1.2.7 `std::vector<cmat> qpp::choi2kraus ( const cmat & A )`

Extracts orthogonal Kraus operators from Choi matrix.

Extracts a set of orthogonal (under Hilbert-Schmidt operator norm) Kraus operators from the Choi representation  $A$  of the channel

**Note**

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

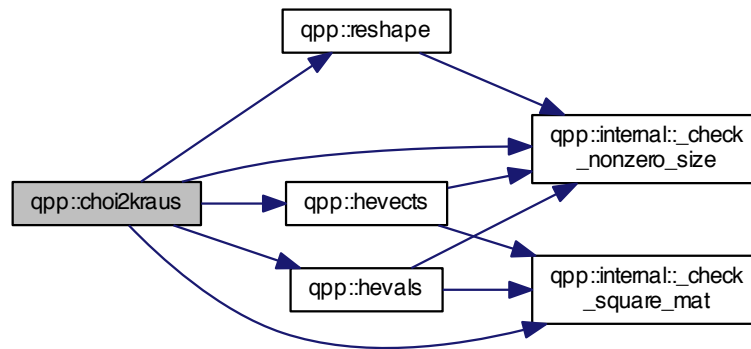
**Parameters**

$A$	Choi matrix
-----	-------------

## Returns

std::vector of dynamic matrices over the complex field representing the set of Kraus operators

Here is the call graph for this function:



**6.1.2.8** `template<typename Derived1 , typename Derived2 > DynMat<typename Derived1::Scalar> qpp::comm ( const Eigen::MatrixBase< Derived1 > & A, const Eigen::MatrixBase< Derived2 > & B )`

Commutator.

Commutator  $[A, B] = AB - BA$

Both  $A$  and  $B$  must be Eigen expressions over the same scalar field

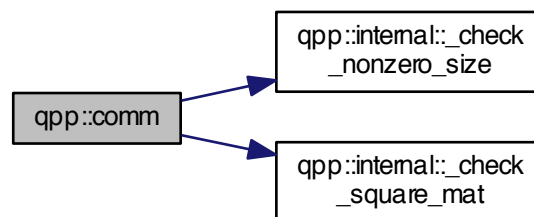
## Parameters

$A$	Eigen expression
$B$	Eigen expression

## Returns

Commutator  $AB - BA$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.9 `std::vector<std::size_t> qpp::compperm ( const std::vector< std::size_t > & perm, const std::vector< std::size_t > & sigma )`

Compose permutations.

## Parameters

<i>perm</i>	Permutation
<i>sigma</i>	Permutation

## Returns

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

Here is the call graph for this function:



6.1.2.10 `template<typename Derived> DynMat<typename Derived::Scalar> qpp::conjugate ( const Eigen::MatrixBase<Derived> & A )`

Complex conjugate.

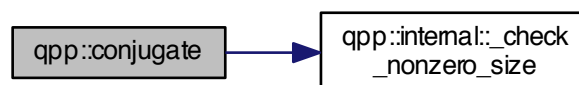
## Parameters

<i>A</i>	Eigen expression
----------	------------------

## Returns

Complex conjugate of *A*, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.11 `template<typename Derived> cmat qpp::cosm ( const Eigen::MatrixBase<Derived> & A )`

Matrix cos.

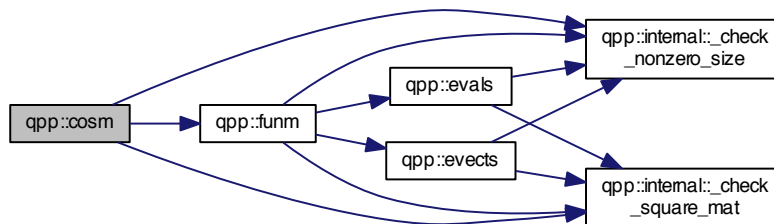
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix cosine of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



**6.1.2.12** `template<typename OutputScalar , typename Derived > DynMat<OutputScalar> qpp::cwise ( const Eigen::MatrixBase< Derived > & A, OutputScalar*)(const typename Derived::Scalar &) f )`

Functor.

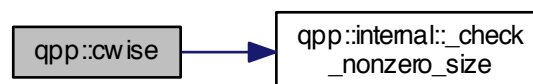
## Parameters

$A$	Eigen expression
$f$	Pointer-to-function from scalars of $A$ to <i>OutputScalar</i>

## Returns

Component-wise  $f(A)$ , as a dynamic matrix over the *OutputScalar* scalar field

Here is the call graph for this function:



**6.1.2.13** `template<typename Derived > Derived::Scalar qpp::det ( const Eigen::MatrixBase< Derived > & A )`

Determinant.

## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Determinant of  $A$ , as a dynamic matrix over the same scalar field  
 Returns  $\pm\infty$  when the determinant overflows/underflows

Here is the call graph for this function:



6.1.2.14 `template<typename T> void qpp::disp ( const T & x, const std::string & separator, const std::string & start = " [ ", const std::string & end = " ] ", std::ostream & os = std::cout )`

6.1.2.15 `template<typename T> void qpp::disp ( const T * x, const std::size_t n, const std::string & separator, const std::string & start = " [ ", const std::string & end = " ] ", std::ostream & os = std::cout )`

6.1.2.16 `template<typename Derived> void qpp::disp ( const Eigen::MatrixBase< Derived > & A, double chop = chop, std::ostream & os = std::cout )`

6.1.2.17 `void qpp::disp ( const cplx c, double chop = chop, std::ostream & os = std::cout )`

Here is the call graph for this function:



6.1.2.18 `template<typename T> void qpp::displn ( const T & x, const std::string & separator, const std::string & start = " [", const std::string & end = "]" , std::ostream & os = std::cout )`

Here is the call graph for this function:



6.1.2.19 `template<typename T> void qpp::displn ( const T * x, const std::size_t n, const std::string & separator, const std::string & start = " [", const std::string & end = "]" , std::ostream & os = std::cout )`

Here is the call graph for this function:



6.1.2.20 `template<typename Derived> void qpp::displn ( const Eigen::MatrixBase< Derived > & A, double chop = chop, std::ostream & os = std::cout )`

Here is the call graph for this function:





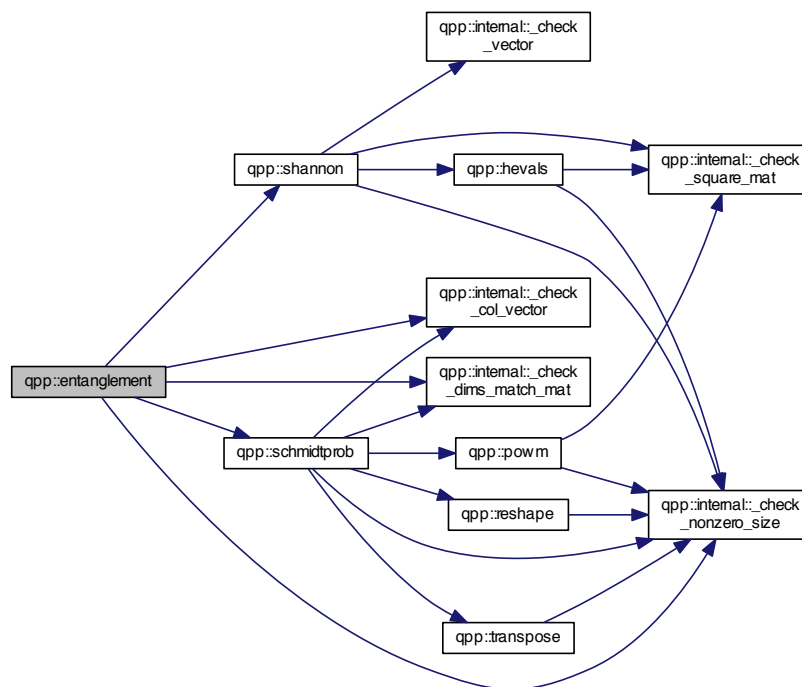
6.1.2.21 `void qpp::displn ( const cplx c, double chop = chop, std::ostream & os = std::cout )`

Here is the call graph for this function:



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

Here is the call graph for this function:



6.1.2.23 `template<typename Derived> cmat qpp::evals ( const Eigen::MatrixBase< Derived> & A )`

Eigenvalues.

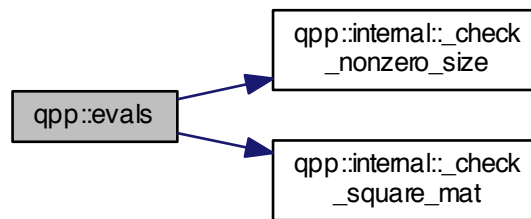
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Eigenvalues of  $A$ , as a diagonal dynamic matrix over the complex field, with eigenvalues on the diagonal

Here is the call graph for this function:



6.1.2.24 `template<typename Derived> cmat qpp::evecs ( const Eigen::MatrixBase< Derived> & A )`

Eigenvectors.

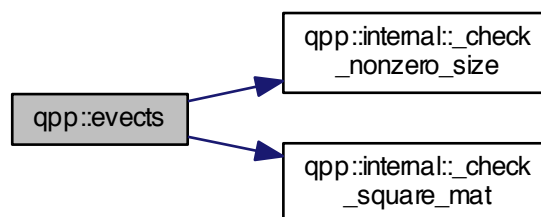
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Eigenvectors of  $A$ , as columns of a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.25 `template<typename Derived > DynMat<typename Derived::Scalar> qpp::expandout ( const Eigen::MatrixBase<Derived > & A, std::size_t pos, const std::vector< std::size_t > & dims )`

Expand out.

Expand out  $A$  as a matrix in a multi-partite system

Faster than using `qpp::kron(I, I, ..., I, A, I, ..., I)`

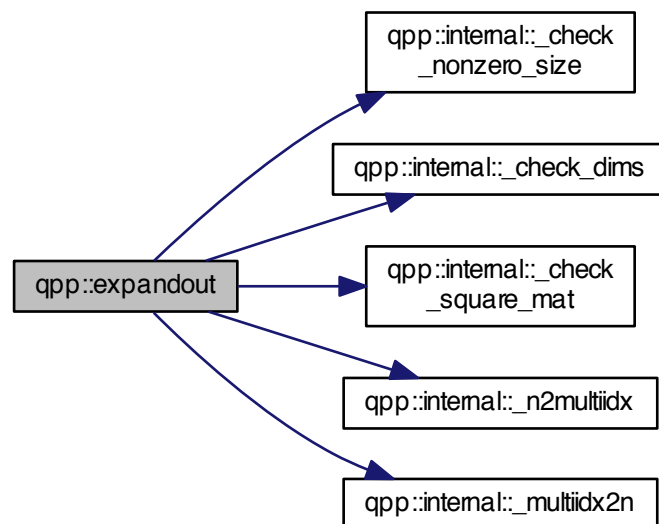
Parameters

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

Returns

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

Here is the call graph for this function:



6.1.2.26 `template<typename Derived > cmat qpp::expm ( const Eigen::MatrixBase< Derived > & A )`

Matrix exponential.

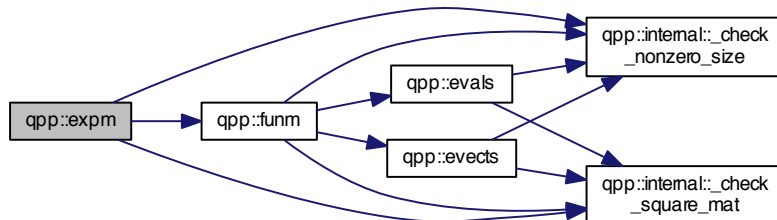
Parameters

$A$	Eigen expression
-----	------------------

**Returns**

Matrix exponential of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.27 `template<typename Derived> cmat qpp::funm ( const Eigen::MatrixBase< Derived> & A, cplx(*) (const cplx &) f )`

Functional calculus  $f(A)$

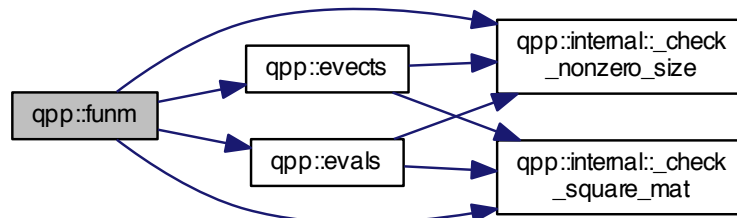
**Parameters**

$A$	Eigen expression
$f$	Pointer-to-function from complex to complex

**Returns**

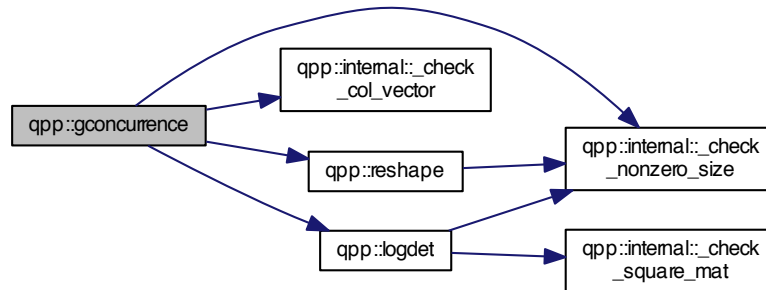
$f(A)$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



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

Here is the call graph for this function:



6.1.2.29 `template<typename Derived> DynMat<typename Derived::Scalar> qpp::grams ( const std::vector< Derived> & Vs )`

Gram-Schmidt orthogonalization (std::vector overload)

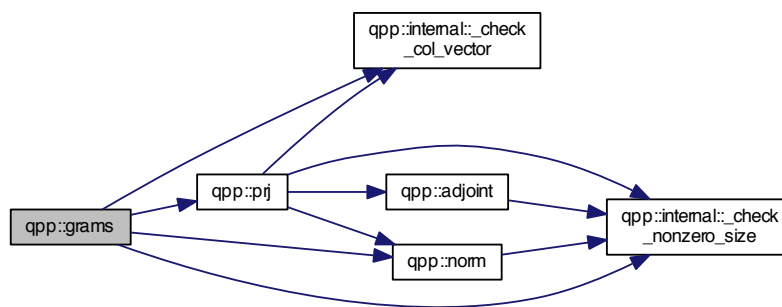
Parameters

<code>Vs</code>	std::vector of Eigen expressions as column vectors
-----------------	--

Returns

Gram-Schmidt vectors of `Vs` as columns of a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.30 `template<typename Derived> DynMat<typename Derived::Scalar> qpp::grams ( const std::initializer_list< Derived> & Vs )`

Gram-Schmidt orthogonalization (std::initializer\_list overload)

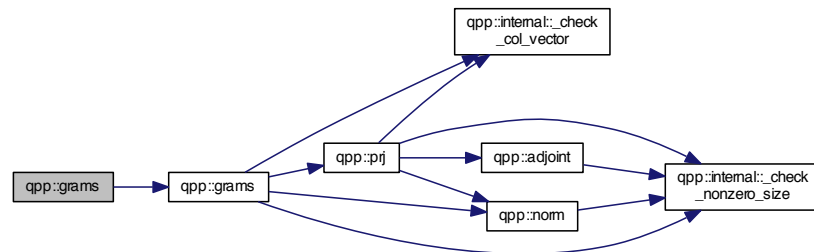
## Parameters

$V$ s	std::initializer_list of Eigen expressions as column vectors
-------	--

## Returns

Gram-Schmidt vectors of  $V$ s as columns of a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.31 `template<typename Derived> DynMat<typename Derived::Scalar> qpp::grams ( const Eigen::MatrixBase<Derived> & A )`

Gram-Schmidt orthogonalization (Eigen expression (matrix) overload)

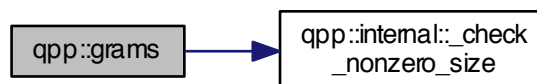
## Parameters

$A$	Eigen expression, the input vectors are the columns of $A$
-----	--

## Returns

Gram-Schmidt vectors of the columns of  $A$ , as columns of a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.32 `template<typename Derived> dmat qpp::hevals ( const Eigen::MatrixBase<Derived> & A )`

Hermitian eigenvalues.

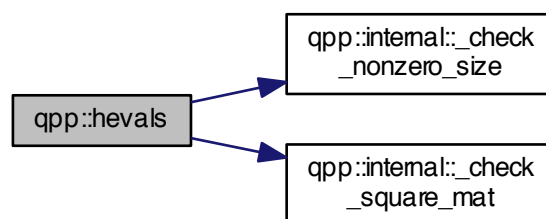
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Eigenvalues of Hermitian  $A$ , as a diagonal dynamic matrix over the real field, with eigenvalues on the diagonal

Here is the call graph for this function:



6.1.2.33 `template<typename Derived> cmat qpp::hevects ( const Eigen::MatrixBase< Derived > & A )`

Hermitian eigenvectors.

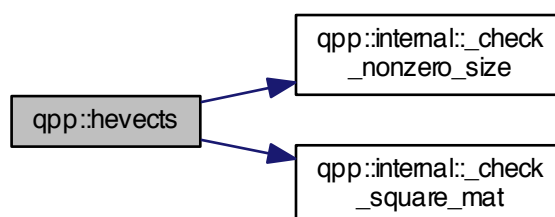
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Eigenvectors of Hermitian  $A$ , as columns of a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.34 `template<typename Derived > DynMat<typename Derived::Scalar> qpp::inverse ( const Eigen::MatrixBase<Derived > & A )`

Inverse.



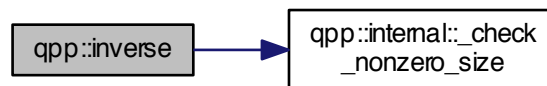
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Inverse of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.35 `std::vector<std::size_t> qpp::invperm ( const std::vector< std::size_t > & perm )`

Inverse permutation.

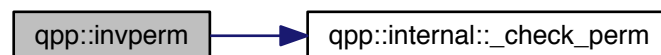
## Parameters

<i>perm</i>	Permutation
-------------	-------------

## Returns

Inverse of the permutation *perm*

Here is the call graph for this function:



6.1.2.36 `template<typename T> DynMat<typename T::Scalar> qpp::kron ( const T & head )`

Kronecker product (variadic overload)

Used to stop the recursion for the variadic template version of [qpp::kron\(\)](#)

## Parameters

<i>head</i>	Eigen expression
-------------	------------------

**Returns**

Its argument *head*

**6.1.2.37** `template<typename T, typename... Args> DynMat<typename T::Scalar> qpp::kron ( const T & head, const Args &... tail )`

Kronecker product (variadic overload)

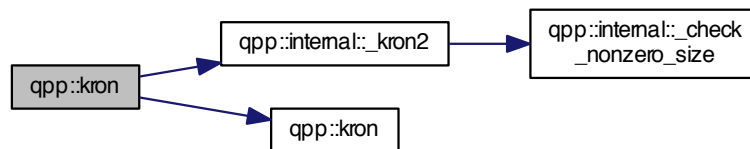
**Parameters**

<i>head</i>	Eigen expression
<i>tail</i>	Variadic Eigen expression (zero or more parameters)

**Returns**

Kronecker product of all input parameters, evaluated from left to right, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**6.1.2.38** `template<typename Derived> DynMat<typename Derived::Scalar> qpp::kron ( const std::vector< Derived > & As )`

Kronecker product (std::vector overload)

**Parameters**

<i>As</i>	std::vector of Eigen expressions
-----------	----------------------------------

**Returns**

Kronecker product of all elements in  $As$ , evaluated from left to right, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**6.1.2.39** `template<typename Derived> DynMat<typename Derived::Scalar> qpp::kron ( const std::initializer_list< Derived > & As )`

Kronecker product (std::initializer\_list overload)

**Parameters**

$As$	std::initializer_list of Eigen expressions, such as $\{A1, A2, \dots, Ak\}$
------	---

**Returns**

Kronecker product of all elements in  $As$ , evaluated from left to right, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**6.1.2.40** `template<typename Derived> DynMat<typename Derived::Scalar> qpp::kronpow ( const Eigen::MatrixBase< Derived > & A, std::size_t n )`

Kronecker power.

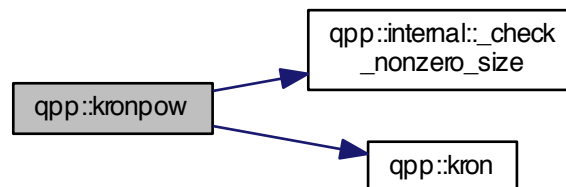
**Parameters**

$A$	Eigen expression
$n$	Non-negative integer

**Returns**

Kronecker product of  $A$  with itself  $n$  times  $A^{\otimes n}$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



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

6.1.2.42 `template<typename Derived> Derived qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

6.1.2.43 `template<> dmat qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

6.1.2.44 `template<> cmat qpp::loadMATLABmatrix ( const std::string & mat_file, const std::string & var_name )`

6.1.2.45 `template<typename Derived> Derived::Scalar qpp::logdet ( const Eigen::MatrixBase< Derived> & A )`

Logarithm of the determinant.

Especially useful when the determinant overflows/underflows

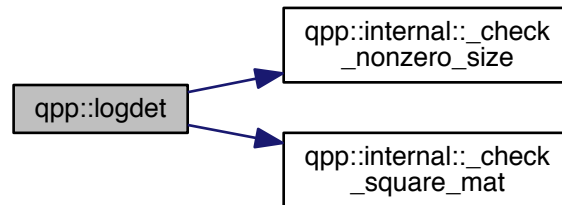
**Parameters**

$A$	Eigen expression
-----	------------------

## Returns

Logarithm of the determinant of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.46 `template<typename Derived> cmat qpp::logm ( const Eigen::MatrixBase< Derived> & A )`

Matrix logarithm.

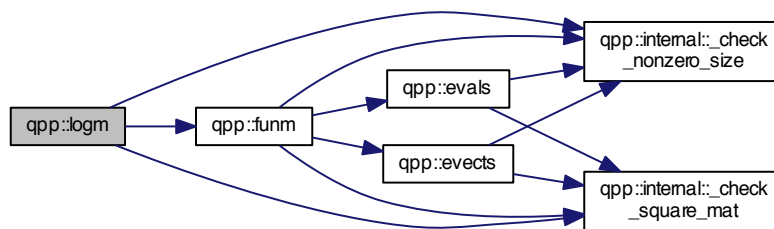
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix logarithm of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.47 `ket qpp::mket ( const std::vector< std::size_t> & mask )`

Multi-partite qubit ket.

Constructs the multi-partite qubit ket  $|\text{mask}\rangle$ , where  $\text{mask}$  is a `std::vector` of 0's and 1's

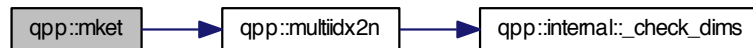
## Parameters

<i>mask</i>	std::vector of 0's and 1's
-------------	----------------------------

## Returns

Multi-partite qubit state vector, as a dynamic column vector over the complex field

Here is the call graph for this function:



#### 6.1.2.48 ket qpp::mket ( const std::vector< std::size\_t > & mask, const std::vector< std::size\_t > & dims )

Multi-partite qudit ket (different dimensions overload)

Constructs the multi-partite qudit ket  $|\text{mask}\rangle$ , where *mask* is a std::vector of non-negative integers. Each element in *mask* has to be smaller than the corresponding element in *dims*.

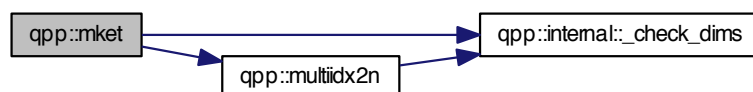
## Parameters

<i>mask</i>	std::vector of non-negative integers
<i>dims</i>	Dimensions of the multi-partite system

## Returns

Multi-partite qudit state vector, as a dynamic column vector over the complex field

Here is the call graph for this function:



#### 6.1.2.49 ket qpp::mket ( const std::vector< std::size\_t > & mask, std::size\_t d )

Multi-partite qudit ket (same dimensions overload)

Constructs the multi-partite qudit ket  $|\text{mask}\rangle$  in a multi-partite system, all subsystem having equal dimension *d*. *mask* is a std::vector of non-negative integers, and each element in *mask* has to be strictly smaller than *d*.

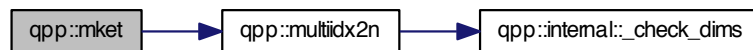
## Parameters

<i>mask</i>	std::vector of non-negative integers
<i>d</i>	Subsystems' dimension

## Returns

Multi-partite qudit state vector, as a dynamic column vector over the complex field

Here is the call graph for this function:



6.1.2.50 `std::size_t qpp::multiidx2n ( const std::vector< std::size_t > & midx, const std::vector< std::size_t > & dims )`

Multi-index to non-negative integer index.

Uses standard lexicographical order, i.e. 00...0, 00...1 etc.

## Parameters

<i>midx</i>	Multi-index
<i>dims</i>	Dimensions of the multi-partite system

## Returns

Non-negative integer index

Here is the call graph for this function:



6.1.2.51 `std::vector<std::size_t> qpp::n2multiidx ( std::size_t n, const std::vector< std::size_t > & dims )`

Non-negative integer index to multi-index.

Uses standard lexicographical order, i.e. 00...0, 00...1 etc.

## Parameters

<i>n</i>	Non-negative integer index
<i>dims</i>	Dimensions of the multi-partite system

## Returns

Multi-index of the same size as *dims*

Here is the call graph for this function:



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

Trace norm.

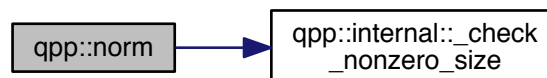
## Parameters

<i>A</i>	Eigen expression
----------	------------------

## Returns

Trace norm (Frobenius norm) of *A*, as a real number

Here is the call graph for this function:



#### 6.1.2.53 `std::complex<double> qpp::omega ( std::size_t D )`

D-th root of unity.

## Parameters



$D$	Non-negative integer
-----	----------------------

**Returns**

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

**6.1.2.54** `constexpr std::complex<double> qpp::operator""_i ( unsigned long long int x )`

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

Example:

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

**6.1.2.55** `constexpr std::complex<double> qpp::operator""_i ( long double x )`

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

Example:

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

**6.1.2.56** `template<typename Derived > DynMat<typename Derived::Scalar> qpp::powm ( const Eigen::MatrixBase<Derived > &A, std::size_t n )`

Matrix power.

Explicitly multiplies the matrix  $A$  with itself  $n$  times

By convention  $A^0 = I$

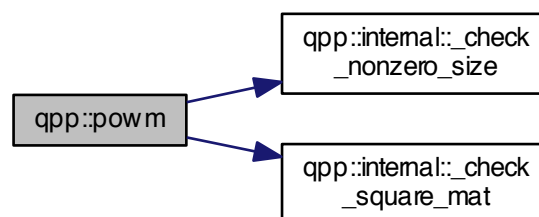
**Parameters**

$A$	Eigen expression
$n$	Non-negative integer

**Returns**

Matrix power  $A^n$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.57 `template<typename Derived> DynMat<typename Derived::Scalar> qpp::prj ( const Eigen::MatrixBase< Derived> & V )`

Projector.

Normalized projector onto state vector

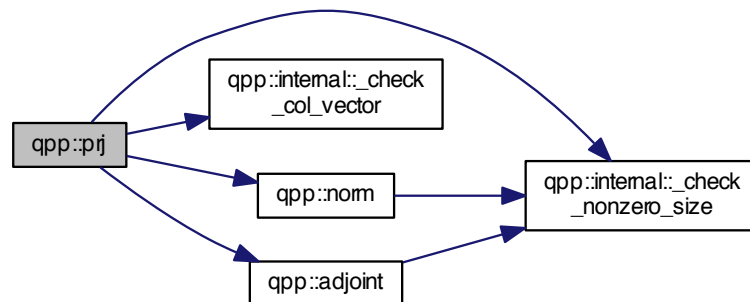
Parameters

<i>V</i>	Eigen expression
----------	------------------

Returns

Projector onto the state vector *V*, or the matrix *Zero* if *V* has norm zero (i.e. smaller than [qpp::eps](#)), as a dynamic matrix over the same scalar field

Here is the call graph for this function:



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

Partial trace.

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

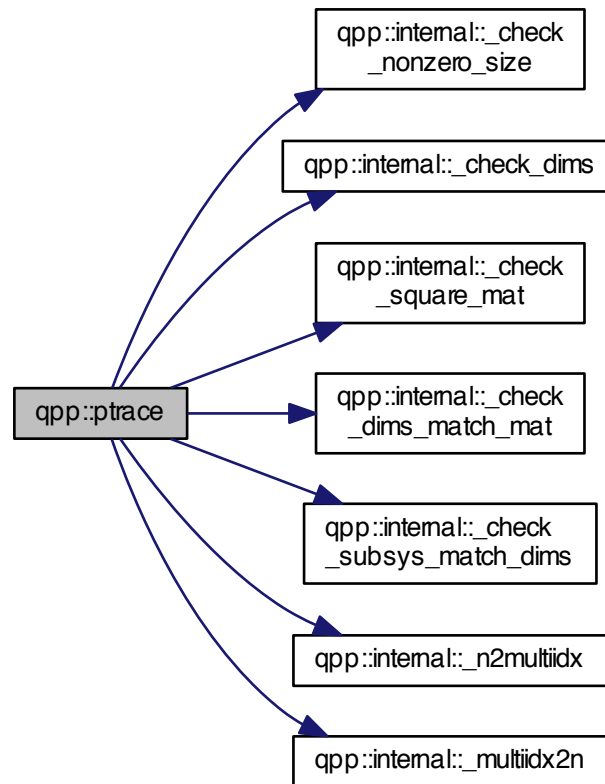
Parameters

<i>A</i>	Eigen expression
<i>subsys</i>	Subsystems' indexes
<i>dims</i>	Dimensions of the multi-partite system

## Returns

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

Here is the call graph for this function:



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

Partial trace.

Partial trace of density matrix over the first subsystem in a bi-partite system

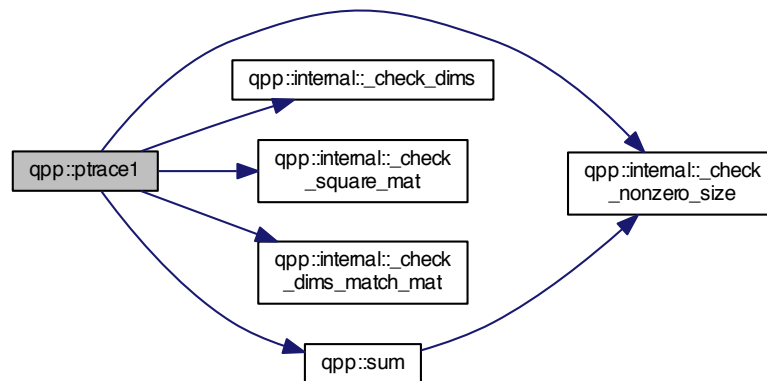
## Parameters

<i>A</i>	Eigen expression
<i>dims</i>	Dimensions of bi-partite system (must be a <code>std::vector</code> with 2 elements)

**Returns**

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

Here is the call graph for this function:



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

Partial trace.

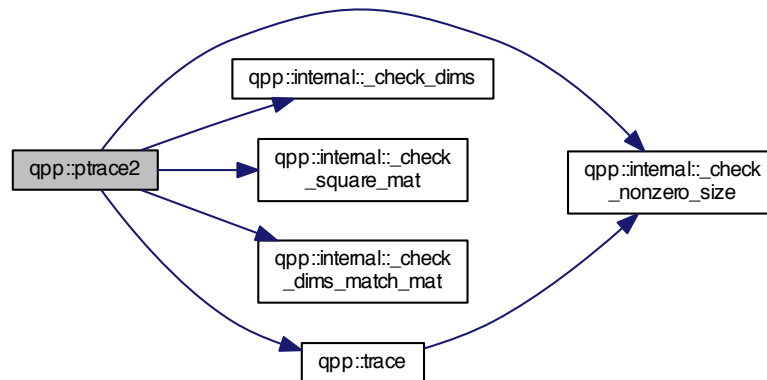
**Parameters**

$A$	Eigen expression
$dims$	Dimensions of bi-partite system (must be a <code>std::vector</code> with 2 elements)

**Returns**

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

Here is the call graph for this function:



**6.1.2.61** `template<typename Derived> DynMat<typename Derived::Scalar> qpp::ptranspose ( const Eigen::MatrixBase<Derived> & A, const std::vector< std::size_t > & subsys, const std::vector< std::size_t > & dims )`

Partial transpose.

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

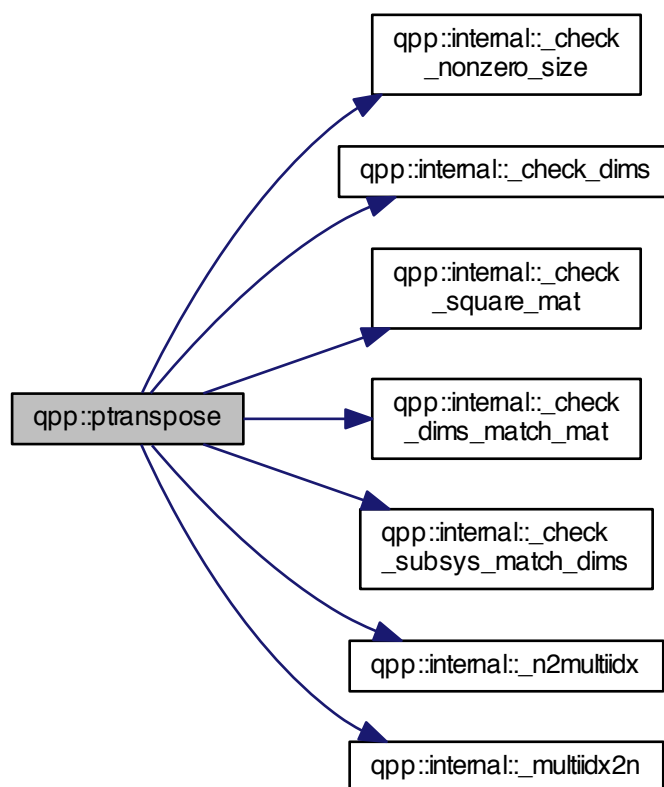
**Parameters**

<i>A</i>	Eigen expression
<i>subsys</i>	Subsystems' indexes
<i>dims</i>	Dimensions of the multi-partite system

**Returns**

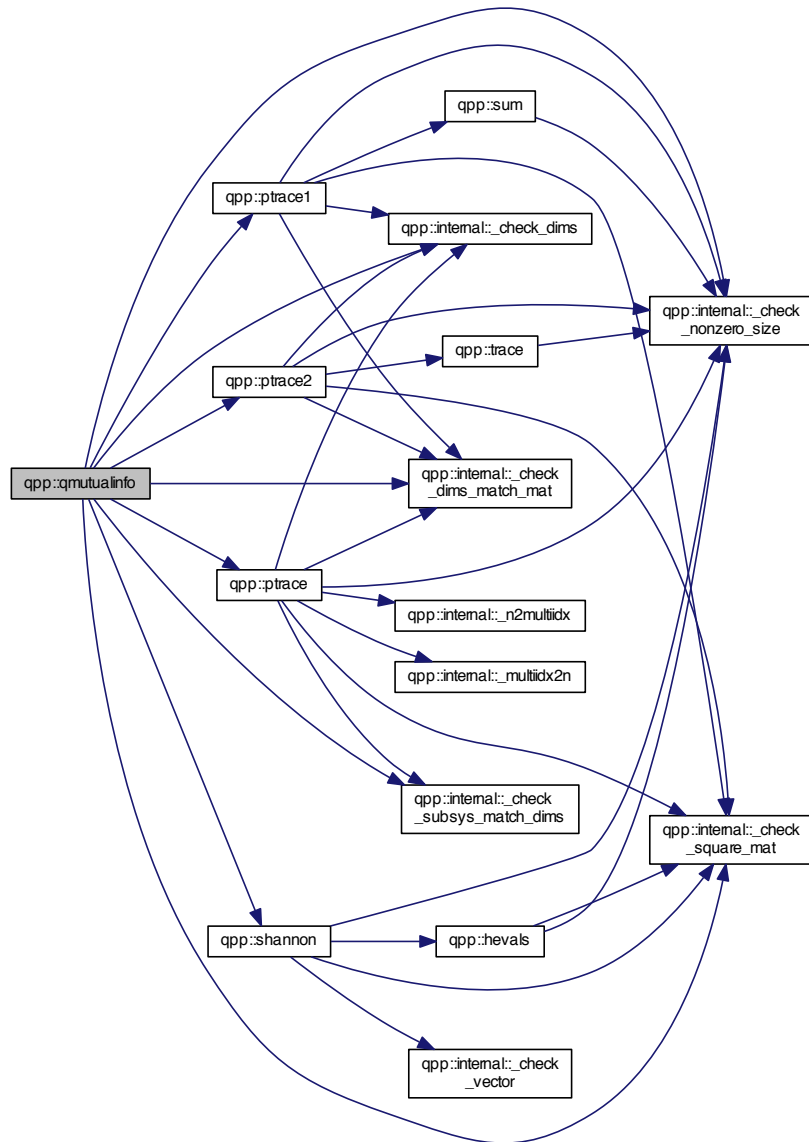
Partial transpose  $(\cdot)^{T_{subsys}}$  over the subsystems *subsys* in a multi-partite system, as a dynamic matrix over the same scalar field

Here is the call graph for this function:



6.1.2.62 `template<typename Derived> double qpp::qmutualinfo ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & subsys, const std::vector< std::size_t> & dims )`

Here is the call graph for this function:



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

6.1.2.64 `template<> dmat qpp::rand ( std::size_t rows, std::size_t cols, double a, double b )`

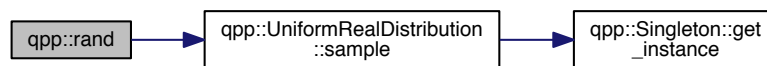
6.1.2.65 `template<> cmat qpp::rand ( std::size_t rows, std::size_t cols, double a, double b )`

Here is the call graph for this function:



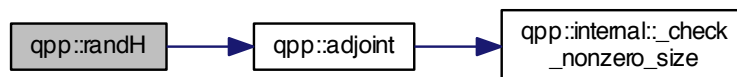
6.1.2.66 `double qpp::rand ( double a = 0, double b = 1 )`

Here is the call graph for this function:



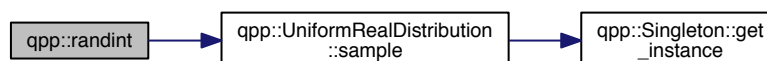
6.1.2.67 `cmat qpp::randH ( std::size_t D )`

Here is the call graph for this function:



6.1.2.68 `long long qpp::randint ( long long a, long long b )`

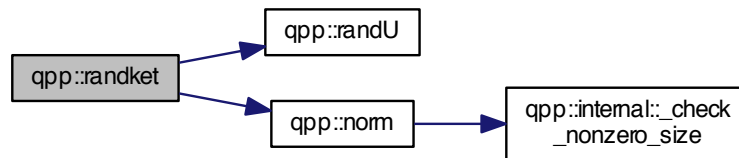
Here is the call graph for this function:



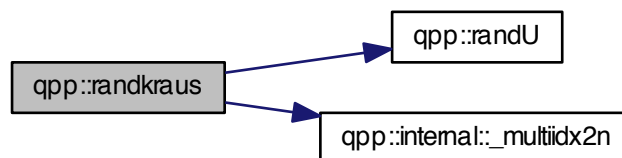


6.1.2.69 `ket qpp::randket ( std::size_t D )`

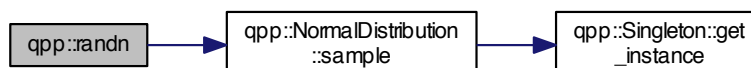
Here is the call graph for this function:

6.1.2.70 `std::vector<cmat> qpp::randkraus ( std::size_t n, std::size_t D )`

Here is the call graph for this function:

6.1.2.71 `template<typename Derived > Derived qpp::randn ( std::size_t rows, std::size_t cols, double mean = 0, double sigma = 1 )`6.1.2.72 `template<> dmat qpp::randn ( std::size_t rows, std::size_t cols, double mean, double sigma )`

Here is the call graph for this function:



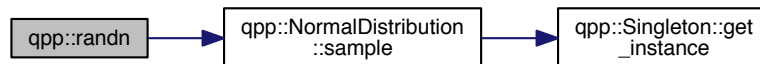
6.1.2.73 `template<> cmat qpp::randn ( std::size_t rows, std::size_t cols, double mean, double sigma )`

Here is the call graph for this function:



6.1.2.74 `double qpp::randn ( double mean = 0, double sigma = 1 )`

Here is the call graph for this function:



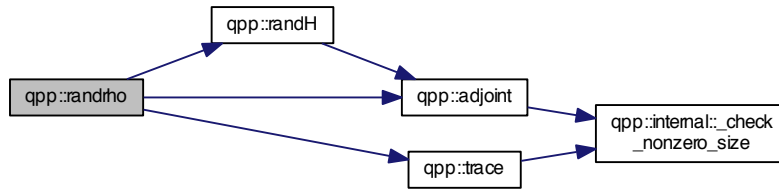
6.1.2.75 `std::vector<std::size_t> qpp::randperm ( std::size_t n )`

Here is the call graph for this function:



6.1.2.76 `cmat qpp::randrho ( std::size_t D )`

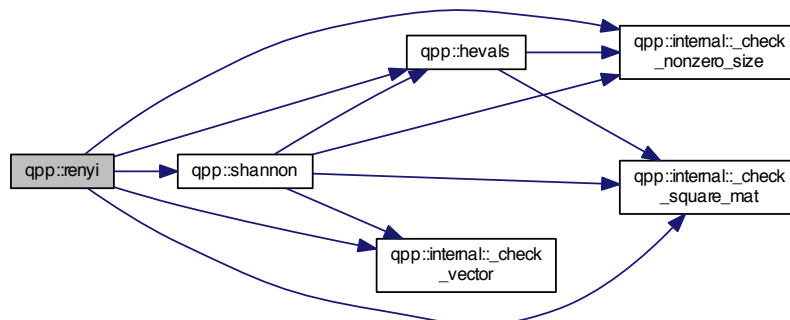
Here is the call graph for this function:

6.1.2.77 `cmat qpp::randU ( std::size_t D )`6.1.2.78 `cmat qpp::randV ( std::size_t Din, std::size_t Dout )`

Here is the call graph for this function:

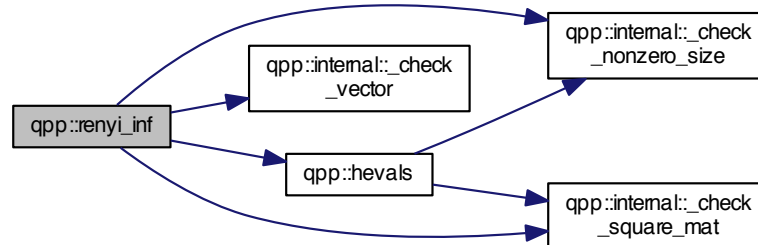
6.1.2.79 `template<typename Derived> double qpp::renyi ( const double alpha, const Eigen::MatrixBase< Derived > & A )`

Here is the call graph for this function:



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

Here is the call graph for this function:



6.1.2.81 `template<typename Derived> DynMat<typename Derived::Scalar> qpp::reshape ( const Eigen::MatrixBase< Derived> & A, std::size_t rows, std::size_t cols )`

Reshape.

Uses column-major order when reshaping (same as MATLAB)

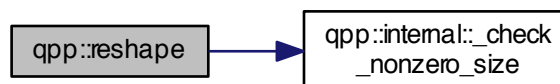
Parameters

<i>A</i>	Eigen expression
<i>rows</i>	Number of rows of the reshaped matrix
<i>cols</i>	Number of columns of the reshaped matrix

Returns

Reshaped matrix with *rows* rows and *cols* columns, as a dynamic matrix over the same scalar field

Here is the call graph for this function:

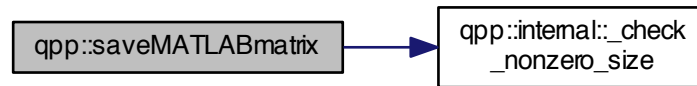


6.1.2.82 `template<typename Derived> void qpp::save ( const Eigen::MatrixBase< Derived> & A, const std::string & fname )`

6.1.2.83 `template<typename Derived> void qpp::saveMATLABmatrix ( const Eigen::MatrixBase< Derived> & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

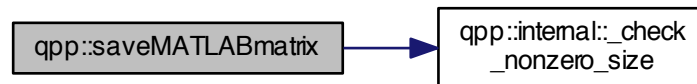
6.1.2.84 `template<> void qpp::saveMATLABmatrix ( const Eigen::MatrixBase< dmat > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



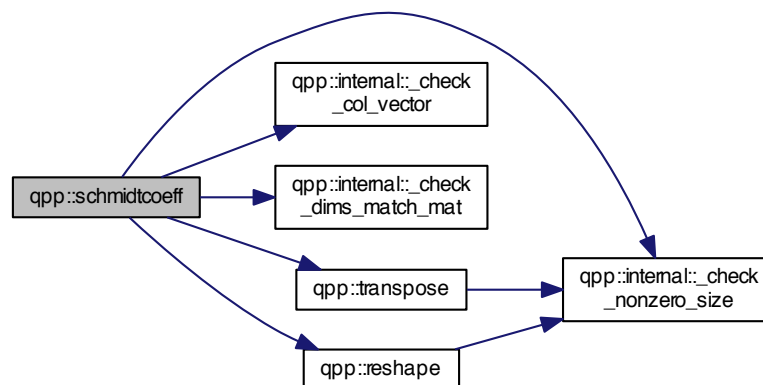
6.1.2.85 `template<> void qpp::saveMATLABmatrix ( const Eigen::MatrixBase< cmat > & A, const std::string & mat_file, const std::string & var_name, const std::string & mode )`

Here is the call graph for this function:



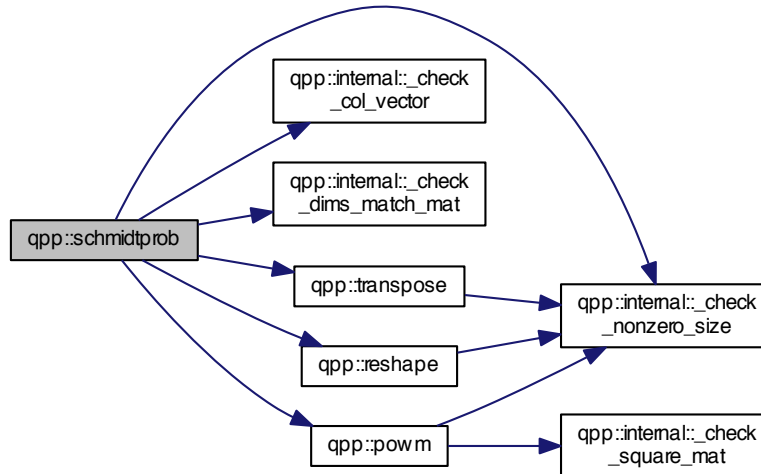
6.1.2.86 `template<typename Derived > cmat qpp::schmidtcoeff ( const Eigen::MatrixBase< Derived > & A, const std::vector< std::size_t > & dims )`

Here is the call graph for this function:



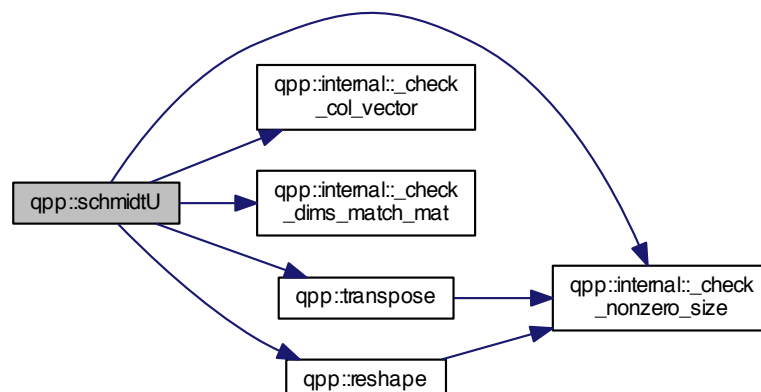
6.1.2.87 `template<typename Derived> cmat qpp::schmidtprob ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & dims )`

Here is the call graph for this function:



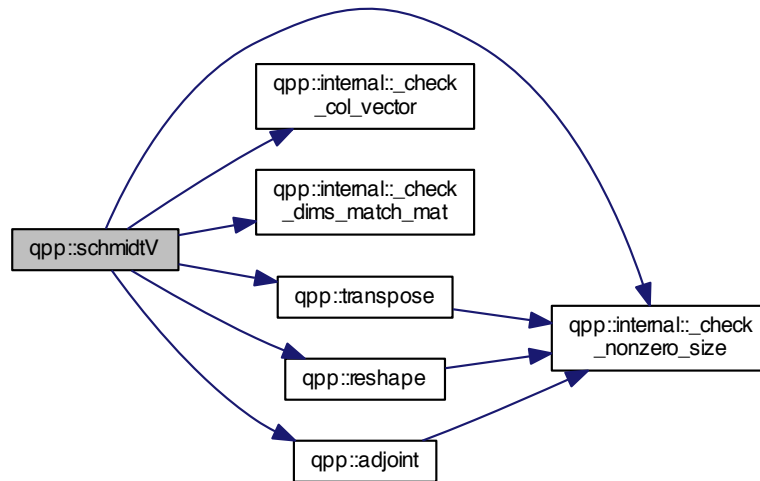
6.1.2.88 `template<typename Derived> cmat qpp::schmidtU ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & dims )`

Here is the call graph for this function:



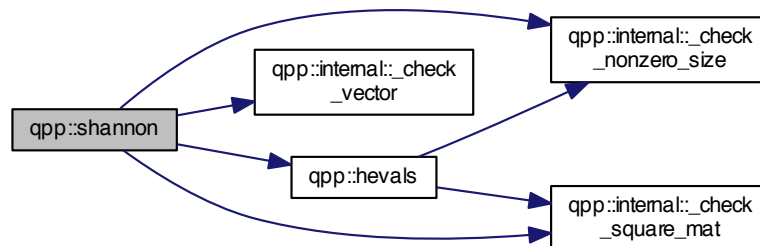
6.1.2.89 `template<typename Derived> cmat qpp::schmidtV ( const Eigen::MatrixBase< Derived> & A, const std::vector< std::size_t> & dims )`

Here is the call graph for this function:



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

Here is the call graph for this function:



6.1.2.91 `template<typename Derived> cmat qpp::sinm ( const Eigen::MatrixBase< Derived> & A )`

Matrix sin.

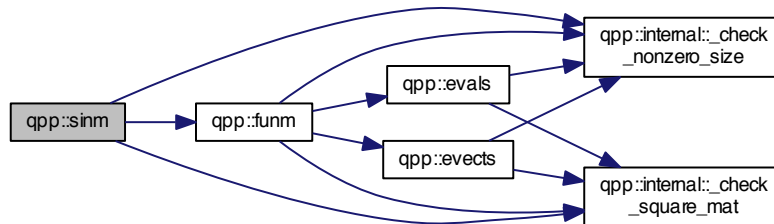
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix sine of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.92 `template<typename Derived> cmat qpp::spectralpowm ( const Eigen::MatrixBase< Derived> & A, const cplx z )`

Matrix power.

Uses the spectral decomposition of  $A$  to compute the matrix power

By convention  $A^0 = I$

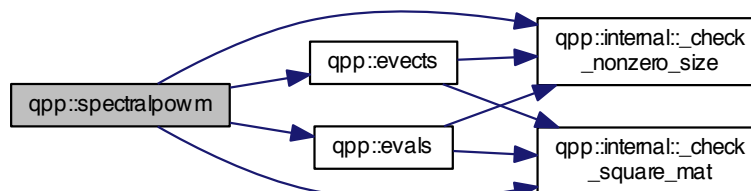
## Parameters

$A$	Eigen expression
$z$	Complex number

## Returns

Matrix power  $A^z$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



6.1.2.93 `template<typename Derived> cmat qpp::sqrtm ( const Eigen::MatrixBase< Derived> & A )`

Matrix square root.



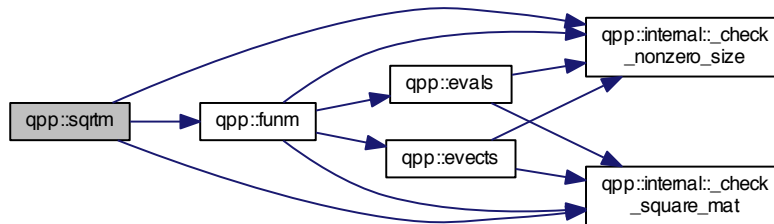
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Matrix square root of  $A$ , as a dynamic matrix over the complex field

Here is the call graph for this function:



**6.1.2.94** `template<typename Derived> Derived::Scalar qpp::sum ( const Eigen::MatrixBase< Derived> & A )`

Element-wise sum.

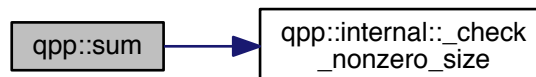
## Parameters

$A$	Eigen expression
-----	------------------

## Returns

Element-wise sum of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**6.1.2.95** `cmat qpp::super ( const std::vector< cmat> & Ks )`

Superoperator matrix representation.

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

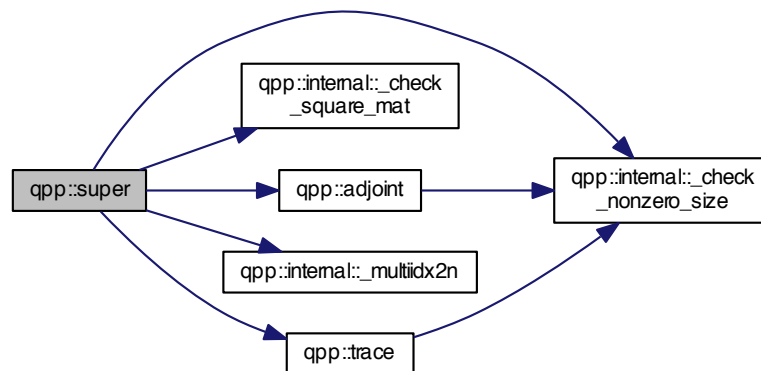
## Parameters

<i>Ks</i>	<code>std::vector</code> of Eigen expressions representing the set of Kraus operators
-----------	---

## Returns

Superoperator matrix representation, as a dynamic matrix over the complex field

Here is the call graph for this function:



**6.1.2.96** `template<typename Derived> DynMat<typename Derived::Scalar> qpp::syspermute ( const Eigen::MatrixBase<Derived> & A, const std::vector< std::size_t > & perm, const std::vector< std::size_t > & dims )`

System permutation.

Permutes the subsystems in a state vector or density matrix

The qubit *perm*[*i*] is permuted to the location *i*

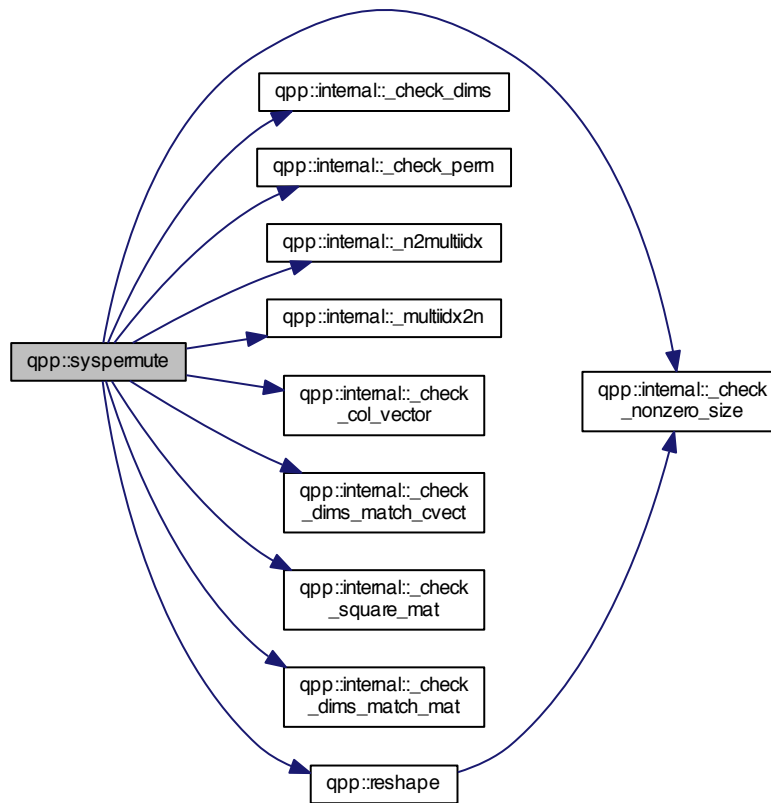
## Parameters

<i>A</i>	Eigen expression
<i>perm</i>	Permutation
<i>dims</i>	Subsystems' dimensions

## Returns

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

Here is the call graph for this function:



6.1.2.97 `template<typename Derived> Derived::Scalar qpp::trace ( const Eigen::MatrixBase< Derived> & A )`

Trace.

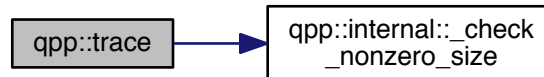
## Parameters

A	Eigen expression
---	------------------

**Returns**

Trace of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



**6.1.2.98** `template<typename Derived > DynMat<typename Derived::Scalar> qpp::transpose ( const Eigen::MatrixBase<Derived > & A )`

Transpose.

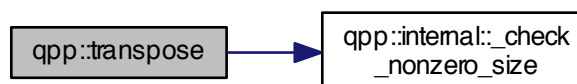
**Parameters**

$A$	Eigen expression
-----	------------------

**Returns**

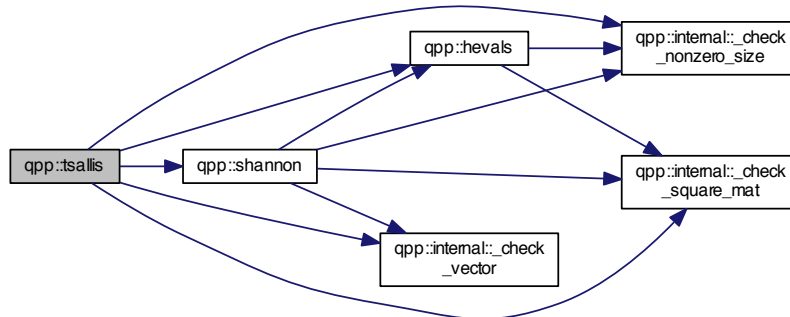
Transpose of  $A$ , as a dynamic matrix over the same scalar field

Here is the call graph for this function:



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

Here is the call graph for this function:



### 6.1.3 Variable Documentation

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

Used in [qpp::disp\(\)](#) and [qpp::displn\(\)](#) for setting to zero numbers that have their absolute value smaller than `qpp::ct::chop`.

6.1.3.2 `constexpr double qpp::ee = 2.718281828459045235360287471352662497`

Base of natural logarithm,  $e$ .

6.1.3.3 `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::ct::eps) // x is zero
```

6.1.3.4 `const Gates& qpp::gt = Gates::get_instance()`

[qpp::Gates](#) const [Singleton](#)

Initializes the gates, see the class [qpp::Gates](#)

6.1.3.5 `constexpr std::size_t qpp::maxn = 64`

Maximum number of qubits.

Used internally to statically allocate arrays (for speed reasons)

6.1.3.6 `constexpr double qpp::pi = 3.141592653589793238462643383279502884`

$\pi$

### 6.1.3.7 RandomDevices& qpp::rdevs = RandomDevices::get\_instance()

[qpp::RandomDevices](#) Singleton

Initializes the random devices, see the class [qpp::RandomDevices](#)

### 6.1.3.8 const States& qpp::st = States::get\_instance()

[qpp::States](#) const Singleton

Initializes the states, see the class [qpp::States](#)

## 6.2 qpp::internal Namespace Reference

### Functions

- void [\\_n2multiidx](#) (std::size\_t n, std::size\_t numdims, const std::size\_t \*dims, std::size\_t \*result)
- std::size\_t [\\_multiidx2n](#) (const std::size\_t \*midx, std::size\_t numdims, const std::size\_t \*dims)
- 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\\_row\\_vector](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [\\_check\\_col\\_vector](#) (const Eigen::MatrixBase< Derived > &A)
- template<typename T >  
bool [\\_check\\_nonzero\\_size](#) (const T &x)
- bool [\\_check\\_dims](#) (const std::vector< std::size\_t > &dims)
- template<typename Derived >  
bool [\\_check\\_dims\\_match\\_mat](#) (const std::vector< std::size\_t > &dims, const Eigen::MatrixBase< Derived > &A)
- template<typename Derived >  
bool [\\_check\\_dims\\_match\\_cvect](#) (const std::vector< std::size\_t > &dims, const Eigen::MatrixBase< Derived > &V)
- template<typename Derived >  
bool [\\_check\\_dims\\_match\\_rvect](#) (const std::vector< std::size\_t > &dims, const Eigen::MatrixBase< Derived > &V)
- bool [\\_check\\_eq\\_dims](#) (const std::vector< std::size\_t > &dims, std::size\_t dim)
- bool [\\_check\\_subsys\\_match\\_dims](#) (const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims)
- bool [\\_check\\_perm](#) (const std::vector< std::size\_t > &perm)
- template<typename Derived1 , typename Derived2 >  
[DynMat](#)< typename Derived1::Scalar > [\\_kron2](#) (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)

### 6.2.1 Detailed Description

Internal functions, do not modify or use directly

## 6.2.2 Function Documentation

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

6.2.2.2 `bool qpp::internal::_check_dims ( const std::vector< std::size_t > & dims )`

6.2.2.3 `template<typename Derived > bool qpp::internal::_check_dims_match_cvect ( const std::vector< std::size_t > & dims, const Eigen::MatrixBase< Derived > & V )`

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

6.2.2.5 `template<typename Derived > bool qpp::internal::_check_dims_match_rvect ( const std::vector< std::size_t > & dims, const Eigen::MatrixBase< Derived > & V )`

6.2.2.6 `bool qpp::internal::_check_eq_dims ( const std::vector< std::size_t > & dims, std::size_t dim )`

6.2.2.7 `template<typename T > bool qpp::internal::_check_nonzero_size ( const T & x )`

6.2.2.8 `bool qpp::internal::_check_perm ( const std::vector< std::size_t > & perm )`

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

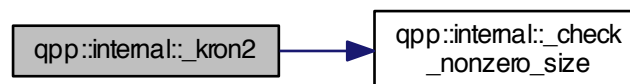
6.2.2.10 `template<typename Derived > bool qpp::internal::_check_square_mat ( const Eigen::MatrixBase< Derived > & A )`

6.2.2.11 `bool qpp::internal::_check_subsys_match_dims ( const std::vector< std::size_t > & subsys, const std::vector< std::size_t > & dims )`

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

6.2.2.13 `template<typename Derived1, typename Derived2 > DynMat<typename Derived1::Scalar> qpp::internal::_kron2 ( const Eigen::MatrixBase< Derived1 > & A, const Eigen::MatrixBase< Derived2 > & B )`

Here is the call graph for this function:



6.2.2.14 `std::size_t qpp::internal::_multiidx2n ( const std::size_t * midx, std::size_t numdims, const std::size_t * dims )`

6.2.2.15 `void qpp::internal::_n2multiidx ( std::size_t n, std::size_t numdims, const std::size_t * dims, std::size_t * result )`

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

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

Here is the call graph for this function:





# Chapter 7

## Class Documentation

### 7.1 qpp::DiscreteDistribution Class Reference

```
#include <stat.h>
```

#### Public Member Functions

- `template<typename InputIterator >`  
`DiscreteDistribution` (InputIterator first, InputIterator last)
- `DiscreteDistribution` (std::initializer\_list< double > weights)
- `DiscreteDistribution` (std::vector< double > weights)
- `std::size_t sample` ()
- `std::vector< double > probabilities` () const

#### Protected Attributes

- `std::discrete_distribution`  
< std::size\_t > `_d`

#### 7.1.1 Constructor & Destructor Documentation

7.1.1.1 `template<typename InputIterator > qpp::DiscreteDistribution::DiscreteDistribution ( InputIterator first, InputIterator last )` [inline]

7.1.1.2 `qpp::DiscreteDistribution::DiscreteDistribution ( std::initializer_list< double > weights )` [inline]

7.1.1.3 `qpp::DiscreteDistribution::DiscreteDistribution ( std::vector< double > weights )` [inline]

#### 7.1.2 Member Function Documentation

7.1.2.1 `std::vector<double> qpp::DiscreteDistribution::probabilities ( ) const` [inline]

### 7.1.2.2 `std::size_t qpp::DiscreteDistribution::sample ( )` `[inline]`

Here is the call graph for this function:



## 7.1.3 Member Data Documentation

### 7.1.3.1 `std::discrete_distribution<std::size_t> qpp::DiscreteDistribution::_d` `[protected]`

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

- [include/classes/stat.h](#)

## 7.2 `qpp::DiscreteDistributionAbsSquare` Class Reference

```
#include <stat.h>
```

### Public Member Functions

- `template<typename InputIterator >`  
[DiscreteDistributionAbsSquare](#) (InputIterator first, InputIterator last)
- [DiscreteDistributionAbsSquare](#) (std::initializer\_list< [cplx](#) > amplitudes)
- [DiscreteDistributionAbsSquare](#) (std::vector< [cplx](#) > amplitudes)
- `template<typename Derived >`  
[DiscreteDistributionAbsSquare](#) (const Eigen::MatrixBase< Derived > &V)
- `std::size_t sample ()`
- `std::vector< double > probabilities () const`

### Protected Member Functions

- `template<typename InputIterator >`  
`std::vector< double > cplx2weights (InputIterator first, InputIterator last) const`

### Protected Attributes

- `std::discrete_distribution`  
`< std::size_t > \_d`

### 7.2.1 Constructor & Destructor Documentation

7.2.1.1 `template<typename InputIterator > qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare ( InputIterator first, InputIterator last ) [inline]`

7.2.1.2 `qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare ( std::initializer_list< cplx > amplitudes ) [inline]`

7.2.1.3 `qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare ( std::vector< cplx > amplitudes ) [inline]`

7.2.1.4 `template<typename Derived > qpp::DiscreteDistributionAbsSquare::DiscreteDistributionAbsSquare ( const Eigen::MatrixBase< Derived > & V ) [inline]`

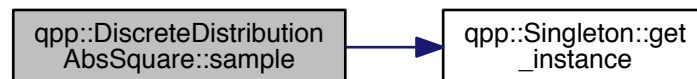
### 7.2.2 Member Function Documentation

7.2.2.1 `template<typename InputIterator > std::vector<double> qpp::DiscreteDistributionAbsSquare::cplx2weights ( InputIterator first, InputIterator last ) const [inline], [protected]`

7.2.2.2 `std::vector<double> qpp::DiscreteDistributionAbsSquare::probabilities ( ) const [inline]`

7.2.2.3 `std::size_t qpp::DiscreteDistributionAbsSquare::sample ( ) [inline]`

Here is the call graph for this function:



### 7.2.3 Member Data Documentation

7.2.3.1 `std::discrete_distribution<std::size_t> qpp::DiscreteDistributionAbsSquare::_d [protected]`

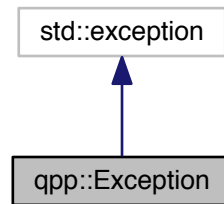
The documentation for this class was generated from the following file:

- [include/classes/stat.h](#)

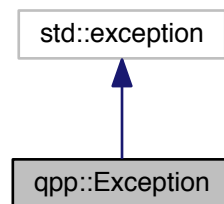
## 7.3 qpp::Exception Class Reference

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

Inheritance diagram for qpp::Exception:



Collaboration diagram for qpp::Exception:



## Public Types

- enum `Type` {  
`Type::UNKNOWN_EXCEPTION = 1`, `Type::ZERO_SIZE`, `Type::MATRIX_NOT_SQUARE`, `Type::MATRIX_NOT_CVECTOR`,  
`Type::MATRIX_NOT_RVECTOR`, `Type::MATRIX_NOT_VECTOR`, `Type::MATRIX_NOT_SQUARE_OR_CVECTOR`,  
`Type::MATRIX_NOT_SQUARE_OR_RVECTOR`,  
`Type::MATRIX_NOT_SQUARE_OR_VECTOR`, `Type::DIMS_INVALID`, `Type::DIMS_NOT_EQUAL`, `Type::DIMS_MISMATCH_MATRIX`,  
`Type::DIMS_MISMATCH_CVECTOR`, `Type::DIMS_MISMATCH_RVECTOR`, `Type::DIMS_MISMATCH_VECTOR`,  
`Type::SUBSYS_MISMATCH_DIMS`,  
`Type::PERM_INVALID`, `Type::NOT_QUBIT_GATE`, `Type::NOT_QUBIT_SUBSYS`, `Type::NOT_BIPARTITE`,  
`Type::OUT_OF_RANGE`, `Type::TYPE_MISMATCH`, `Type::UNDEFINED_TYPE`, `Type::CUSTOM_EXCEPTION` }

## Public Member Functions

- `Exception` (const std::string &where, const `Type` &type)
- `Exception` (const std::string &where, const std::string &custom)
- virtual const char \* `what` () const noexcept override

## Private Member Functions

- `std::string _construct_exception_msg ()`

## Private Attributes

- `std::string _where`
- `std::string _msg`
- `Type _type`
- `std::string _custom`

### 7.3.1 Member Enumeration Documentation

#### 7.3.1.1 enum `qpp::Exception::Type` [strong]

##### Enumerator

- UNKNOWN\_EXCEPTION** Unknown exception
- ZERO\_SIZE** Zero sized object, e.g. empty `Eigen::Matrix` or `std::vector` with no elements
- MATRIX\_NOT\_SQUARE** `Eigen::Matrix` is not square
- MATRIX\_NOT\_CVECTOR** `Eigen::Matrix` is not a column vector
- MATRIX\_NOT\_RVECTOR** `Eigen::Matrix` is not a row vector
- MATRIX\_NOT\_VECTOR** `Eigen::Matrix` is not a row/column vector
- MATRIX\_NOT\_SQUARE\_OR\_CVECTOR** `Eigen::Matrix` is not square nor a column vector
- MATRIX\_NOT\_SQUARE\_OR\_RVECTOR** `Eigen::Matrix` is not square nor a row vector
- MATRIX\_NOT\_SQUARE\_OR\_VECTOR** `Eigen::Matrix` is not square nor a row/column vector
- DIMS\_INVALID** `std::vector<std::size_t>` representing the dimensions has zero size or contains zeros
- DIMS\_NOT\_EQUAL** `std::vector<std::size_t>` representing the dimensions contains non-equal elements
- DIMS\_MISMATCH\_MATRIX** Product of the dimensions' `std::vector<std::size_t>` is not equal to the number of rows of `Eigen::Matrix` (assumed to be square)
- DIMS\_MISMATCH\_CVECTOR** Product of the dimensions' `std::vector<std::size_t>` is not equal to the number of cols of `Eigen::Matrix` (assumed to be a column vector)
- DIMS\_MISMATCH\_RVECTOR** Product of the dimensions' `std::vector<std::size_t>` is not equal to the number of cols of `Eigen::Matrix` (assumed to be a row vector)
- DIMS\_MISMATCH\_VECTOR** Product of the dimensions' `std::vector<std::size_t>` is not equal to the number of cols of `Eigen::Matrix` (assumed to be a row/column vector)
- SUBSYS\_MISMATCH\_DIMS** `std::vector<std::size_t>` representing the subsystems' labels has duplicates, or has entries that are larger than the size of the `std::vector<std::size_t>` representing the dimensions
- PERM\_INVALID** Invalid `std::vector<std::size_t>` permutation
- NOT\_QUBIT\_GATE** `Eigen::Matrix` is not 2 x 2
- NOT\_QUBIT\_SUBSYS** Subsystems are not 2-dimensional
- NOT\_BIPARTITE** `std::vector<std::size_t>` representing the dimensions has size different from 2
- OUT\_OF\_RANGE** Parameter out of range
- TYPE\_MISMATCH** Types do not match (i.e. `Matrix<double>` vs `Matrix<cplx>`)
- UNDEFINED\_TYPE** Templated function not defined for this type
- CUSTOM\_EXCEPTION** Custom exception, user must provide a custom message

### 7.3.2 Constructor & Destructor Documentation

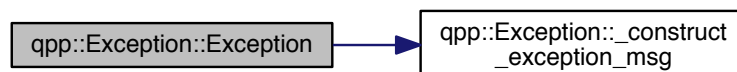
7.3.2.1 `qpp::Exception::Exception ( const std::string & where, const Type & type )` `[inline]`

Here is the call graph for this function:



7.3.2.2 `qpp::Exception::Exception ( const std::string & where, const std::string & custom )` `[inline]`

Here is the call graph for this function:



### 7.3.3 Member Function Documentation

7.3.3.1 `std::string qpp::Exception::_construct_exception_msg ( )` `[inline]`, `[private]`

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

### 7.3.4 Member Data Documentation

7.3.4.1 `std::string qpp::Exception::_custom` `[private]`

7.3.4.2 `std::string qpp::Exception::_msg` `[private]`

7.3.4.3 `Type qpp::Exception::_type` `[private]`

7.3.4.4 `std::string qpp::Exception::_where` `[private]`

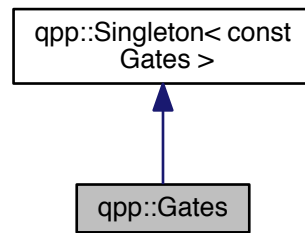
The documentation for this class was generated from the following file:

- [include/classes/exception.h](#)

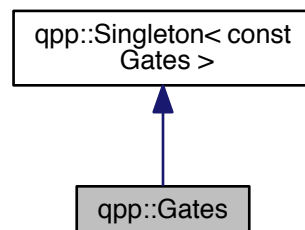
## 7.4 qpp::Gates Class Reference

```
#include <gates.h>
```

Inheritance diagram for qpp::Gates:



Collaboration diagram for qpp::Gates:



## Public Member Functions

- [cmat Rn](#) (double theta, std::vector< double > n) const
- [cmat Zd](#) (std::size\_t D) const
- [cmat Fd](#) (std::size\_t D) const
- [cmat Xd](#) (std::size\_t D) const
- template<typename Derived = Eigen::MatrixXcd>  
Derived [ld](#) (std::size\_t D) const
- template<typename Derived1 , typename Derived2 >  
[DynMat](#)< typename Derived1::Scalar > [applyCTRL](#) (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< Derived2 > &A, const std::vector< std::size\_t > &ctrl, const std::vector< std::size\_t > &subsys, std::size\_t n, std::size\_t d=2) const
- template<typename Derived1 , typename Derived2 >  
[DynMat](#)< typename Derived1::Scalar > [apply](#) (const Eigen::MatrixBase< Derived1 > &state, const Eigen::MatrixBase< Derived2 > &A, const std::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims) const
- template<typename Derived >  
[DynMat](#)< typename Derived::Scalar > [CTRL](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &ctrl, const std::vector< std::size\_t > &subsys, std::size\_t n, std::size\_t d=2) const

## Public Attributes

- [cmat Id2](#) { cmat::Identity(2, 2) }
- [cmat H](#) { cmat::Zero(2, 2) }
- [cmat X](#) { cmat::Zero(2, 2) }
- [cmat Y](#) { cmat::Zero(2, 2) }
- [cmat Z](#) { cmat::Zero(2, 2) }
- [cmat S](#) { cmat::Zero(2, 2) }
- [cmat T](#) { cmat::Zero(2, 2) }
- [cmat CNOTab](#) { cmat::Identity(4, 4) }
- [cmat CZ](#) { cmat::Identity(4, 4) }
- [cmat CNOTba](#) { cmat::Zero(4, 4) }
- [cmat SWAP](#) { cmat::Identity(4, 4) }
- [cmat TOF](#) { cmat::Identity(8, 8) }
- [cmat FRED](#) { cmat::Identity(8, 8) }

## Private Member Functions

- [Gates](#) ()

## Friends

- class [Singleton](#)< const [Gates](#) >

## Additional Inherited Members

### 7.4.1 Constructor & Destructor Documentation

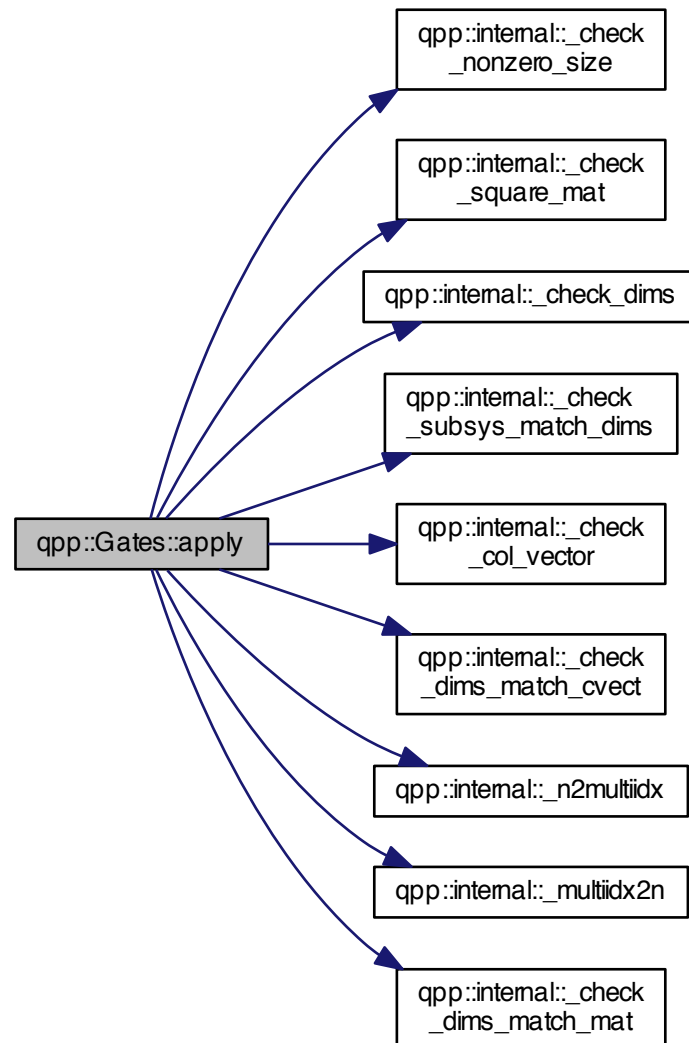
7.4.1.1 `qpp::Gates::Gates ( )` `[inline]`, `[private]`

### 7.4.2 Member Function Documentation



7.4.2.1 `template<typename Derived1 , typename Derived2 > DynMat<typename Derived1::Scalar> qpp::Gates::apply  
( const Eigen::MatrixBase< Derived1 > & state, const Eigen::MatrixBase< Derived2 > & A, const std::vector<  
std::size_t > & subsys, const std::vector< std::size_t > & dims ) const [inline]`

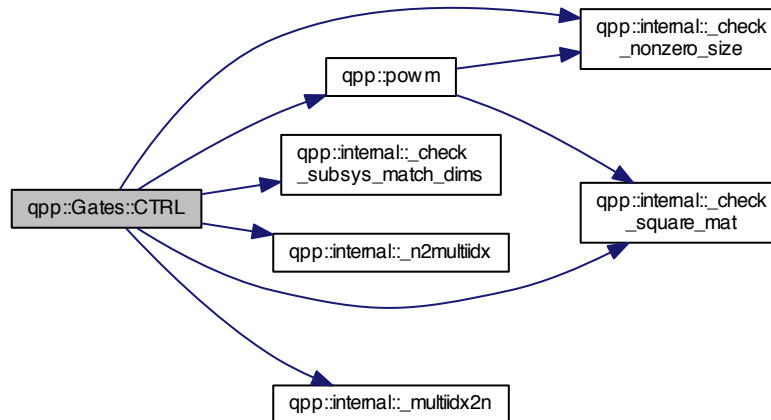
Here is the call graph for this function:



7.4.2.2 `template<typename Derived1 , typename Derived2 > DynMat<typename Derived1::Scalar> qpp::Gates::applyCTRL  
( const Eigen::MatrixBase< Derived1 > & state, const Eigen::MatrixBase< Derived2 > & A, const std::vector<  
std::size_t > & ctrl, const std::vector< std::size_t > & subsys, std::size_t n, std::size_t d=2 ) const [inline]`

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

Here is the call graph for this function:



7.4.2.4 `cmat qpp::Gates::Fd ( std::size_t D ) const [inline]`

Here is the call graph for this function:



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

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

#### 7.4.2.7 `cmat qpp::Gates::Xd ( std::size_t D ) const [inline]`

Here is the call graph for this function:



#### 7.4.2.8 `cmat qpp::Gates::Zd ( std::size_t D ) const [inline]`

Here is the call graph for this function:



### 7.4.3 Friends And Related Function Documentation

#### 7.4.3.1 `friend class Singleton< const Gates > [friend]`

### 7.4.4 Member Data Documentation

#### 7.4.4.1 `cmat qpp::Gates::CNOTab { cmat::Identity(4, 4) }`

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

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

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

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

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

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

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

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

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

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

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

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

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

- [include/classes/gates.h](#)

## 7.5 qpp::NormalDistribution Class Reference

```
#include <stat.h>
```

### Public Member Functions

- [NormalDistribution](#) (double mean=0, double sigma=1)
- double [sample](#) ()

### Protected Attributes

- `std::normal_distribution _d`

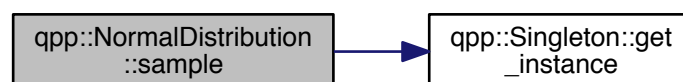
#### 7.5.1 Constructor & Destructor Documentation

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

#### 7.5.2 Member Function Documentation

7.5.2.1 `double qpp::NormalDistribution::sample ( )` `[inline]`

Here is the call graph for this function:



#### 7.5.3 Member Data Documentation

7.5.3.1 `std::normal_distribution qpp::NormalDistribution::_d` `[protected]`

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

- [include/classes/stat.h](#)

## 7.6 qpp::Qudit Class Reference

```
#include <qudit.h>
```

### Public Member Functions

- [Qudit](#) (const [cmat](#) &rho=[States::get\\_instance\(\)](#).pz0)
- [std::size\\_t measure](#) (const [cmat](#) &U, bool destructive=false)
- [std::size\\_t measure](#) (bool destructive=false)
- [cmat getRho](#) () const
- [std::size\\_t getD](#) () const

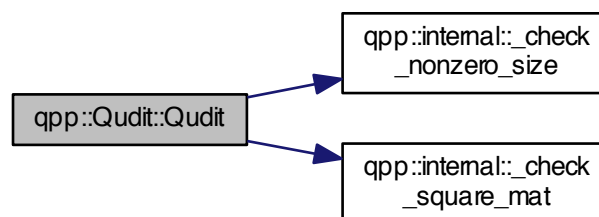
### Private Attributes

- [cmat \\_rho](#)
- [std::size\\_t \\_D](#)

### 7.6.1 Constructor & Destructor Documentation

7.6.1.1 `qpp::Qudit::Qudit ( const cmat & rho = States::get\_instance\(\) .pz0 ) [inline]`

Here is the call graph for this function:



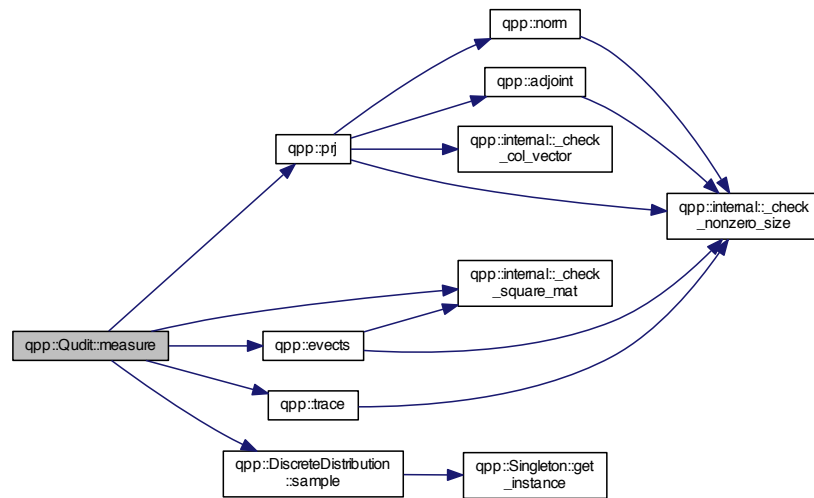
### 7.6.2 Member Function Documentation

7.6.2.1 `std::size_t qpp::Qudit::getD ( ) const [inline]`

7.6.2.2 `cmat qpp::Qudit::getRho ( ) const [inline]`

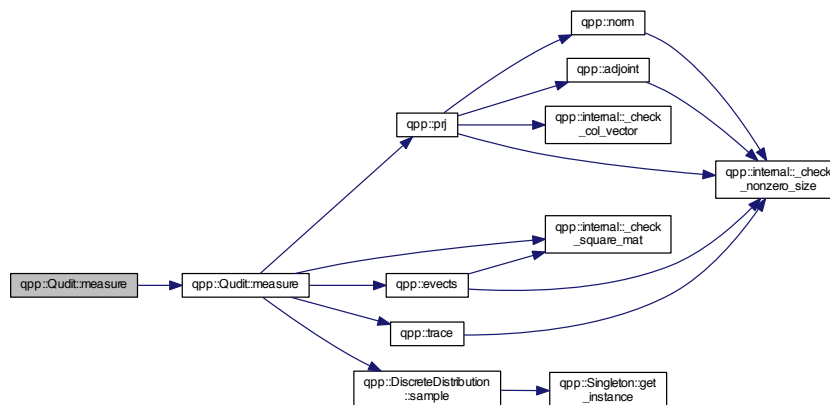
### 7.6.2.3 `std::size_t qpp::Qudit::measure ( const cmat & U, bool destructive = false ) [inline]`

Here is the call graph for this function:



### 7.6.2.4 `std::size_t qpp::Qudit::measure ( bool destructive = false ) [inline]`

Here is the call graph for this function:



## 7.6.3 Member Data Documentation

### 7.6.3.1 `std::size_t qpp::Qudit::_D [private]`

### 7.6.3.2 `cmat qpp::Qudit::_rho [private]`

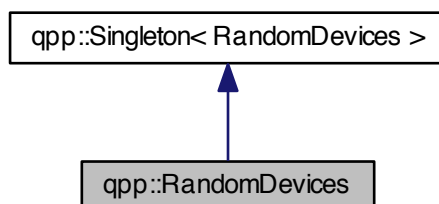
The documentation for this class was generated from the following file:

- [include/classes/qudit.h](#)

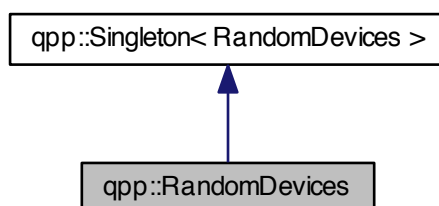
## 7.7 qpp::RandomDevices Class Reference

```
#include <randevs.h>
```

Inheritance diagram for qpp::RandomDevices:



Collaboration diagram for qpp::RandomDevices:



### Public Attributes

- [std::mt19937 \\_rng](#)

### Private Member Functions

- [RandomDevices \(\)](#)

### Private Attributes

- [std::random\\_device \\_rd](#)

### Friends

- class [Singleton< RandomDevices >](#)

## Additional Inherited Members

### 7.7.1 Constructor & Destructor Documentation

7.7.1.1 `qpp::RandomDevices::RandomDevices ( )` `[inline]`, `[private]`

### 7.7.2 Friends And Related Function Documentation

7.7.2.1 `friend class Singleton< RandomDevices >` `[friend]`

### 7.7.3 Member Data Documentation

7.7.3.1 `std::random_device qpp::RandomDevices::_rd` `[private]`

7.7.3.2 `std::mt19937 qpp::RandomDevices::_rng`

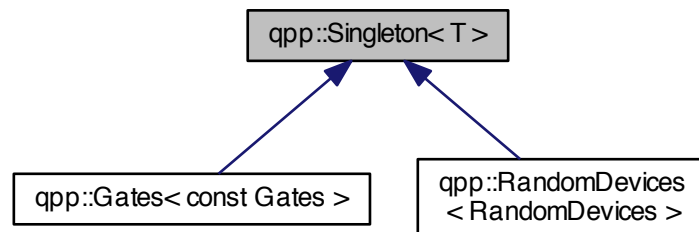
The documentation for this class was generated from the following file:

- `include/classes/randevs.h`

## 7.8 `qpp::Singleton< T >` Class Template Reference

```
#include <singleton.h>
```

Inheritance diagram for `qpp::Singleton< T >`:



## Static Public Member Functions

- static `T & get_instance ()`

## Protected Member Functions

- `Singleton ()`=default
- virtual `~Singleton ()`
- `Singleton (const Singleton &)=delete`
- `Singleton & operator= (const Singleton &)=delete`



### 7.8.1 Constructor & Destructor Documentation

7.8.1.1 `template<typename T> qpp::Singleton< T >::Singleton ( )` `[protected]`, `[default]`

7.8.1.2 `template<typename T> virtual qpp::Singleton< T >::~~Singleton ( )` `[inline]`, `[protected]`, `[virtual]`

7.8.1.3 `template<typename T> qpp::Singleton< T >::Singleton ( const Singleton< T > & )` `[protected]`, `[delete]`

### 7.8.2 Member Function Documentation

7.8.2.1 `template<typename T> static T& qpp::Singleton< T >::get_instance ( )` `[inline]`, `[static]`

7.8.2.2 `template<typename T> Singleton& qpp::Singleton< T >::operator= ( const Singleton< T > & )` `[protected]`, `[delete]`

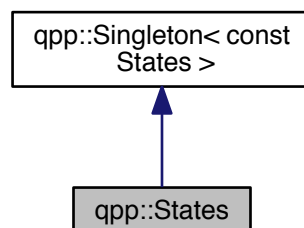
The documentation for this class was generated from the following file:

- `include/classes/singleton.h`

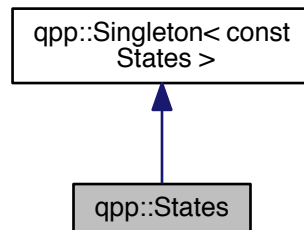
## 7.9 qpp::States Class Reference

```
#include <states.h>
```

Inheritance diagram for qpp::States:



Collaboration diagram for qpp::States:



### Public Attributes

- [ket x0](#) { ket::Zero(2) }
- [ket x1](#) { ket::Zero(2) }
- [ket y0](#) { ket::Zero(2) }
- [ket y1](#) { ket::Zero(2) }
- [ket z0](#) { ket::Zero(2) }
- [ket z1](#) { ket::Zero(2) }
- [cmat px0](#) { cmat::Zero(2, 2) }
- [cmat px1](#) { cmat::Zero(2, 2) }
- [cmat py0](#) { cmat::Zero(2, 2) }
- [cmat py1](#) { cmat::Zero(2, 2) }
- [cmat pz0](#) { cmat::Zero(2, 2) }
- [cmat pz1](#) { cmat::Zero(2, 2) }
- [ket b00](#) { ket::Zero(4) }
- [ket b01](#) { ket::Zero(4) }
- [ket b10](#) { ket::Zero(4) }
- [ket b11](#) { ket::Zero(4) }
- [cmat pb00](#) { cmat::Zero(4, 4) }
- [cmat pb01](#) { cmat::Zero(4, 4) }
- [cmat pb10](#) { cmat::Zero(4, 4) }
- [cmat pb11](#) { cmat::Zero(4, 4) }
- [ket GHZ](#) { ket::Zero(8) }
- [ket W](#) { ket::Zero(8) }
- [cmat pGHZ](#) { cmat::Zero(8, 8) }
- [cmat pW](#) { cmat::Zero(8, 8) }

### Private Member Functions

- [States](#) ()

### Friends

- class [Singleton< const States >](#)

## Additional Inherited Members

### 7.9.1 Constructor & Destructor Documentation

7.9.1.1 `qpp::States::States ( )` `[inline]`, `[private]`

### 7.9.2 Friends And Related Function Documentation

7.9.2.1 `friend class Singleton< const States >` `[friend]`

### 7.9.3 Member Data Documentation

7.9.3.1 `ket qpp::States::b00 { ket::Zero(4) }`

7.9.3.2 `ket qpp::States::b01 { ket::Zero(4) }`

7.9.3.3 `ket qpp::States::b10 { ket::Zero(4) }`

7.9.3.4 `ket qpp::States::b11 { ket::Zero(4) }`

7.9.3.5 `ket qpp::States::GHZ { ket::Zero(8) }`

7.9.3.6 `cmat qpp::States::pb00 { cmat::Zero(4, 4) }`

7.9.3.7 `cmat qpp::States::pb01 { cmat::Zero(4, 4) }`

7.9.3.8 `cmat qpp::States::pb10 { cmat::Zero(4, 4) }`

7.9.3.9 `cmat qpp::States::pb11 { cmat::Zero(4, 4) }`

7.9.3.10 `cmat qpp::States::pGHZ { cmat::Zero(8, 8) }`

7.9.3.11 `cmat qpp::States::pW { cmat::Zero(8, 8) }`

7.9.3.12 `cmat qpp::States::px0 { cmat::Zero(2, 2) }`

7.9.3.13 `cmat qpp::States::px1 { cmat::Zero(2, 2) }`

7.9.3.14 `cmat qpp::States::py0 { cmat::Zero(2, 2) }`

7.9.3.15 `cmat qpp::States::py1 { cmat::Zero(2, 2) }`

7.9.3.16 `cmat qpp::States::pz0 { cmat::Zero(2, 2) }`

7.9.3.17 `cmat qpp::States::pz1 { cmat::Zero(2, 2) }`

7.9.3.18 `ket qpp::States::W { ket::Zero(8) }`

7.9.3.19 `ket qpp::States::x0 { ket::Zero(2) }`

7.9.3.20 `ket qpp::States::x1 { ket::Zero(2) }`

7.9.3.21 `ket qpp::States::y0 { ket::Zero(2) }`

7.9.3.22 `ket qpp::States::y1 { ket::Zero(2) }`

7.9.3.23 `ket qpp::States::z0 { ket::Zero(2) }`

7.9.3.24 `ket qpp::States::z1 { ket::Zero(2) }`

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

- `include/classes/states.h`

## 7.10 qpp::Timer Class Reference

```
#include <timer.h>
```

### Public Member Functions

- [Timer](#) ()
- void [tic](#) ()
- void [toc](#) ()
- double [seconds](#) () const

### Protected Attributes

- `std::chrono::steady_clock::time_point` [\\_start](#)
- `std::chrono::steady_clock::time_point` [\\_end](#)

### Friends

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

### 7.10.1 Constructor & Destructor Documentation

7.10.1.1 `qpp::Timer::Timer ( )` [`inline`]

### 7.10.2 Member Function Documentation

7.10.2.1 `double qpp::Timer::seconds ( )` const [`inline`]

7.10.2.2 `void qpp::Timer::tic ( )` [`inline`]

7.10.2.3 `void qpp::Timer::toc ( )` [`inline`]

### 7.10.3 Friends And Related Function Documentation

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

### 7.10.4 Member Data Documentation

7.10.4.1 `std::chrono::steady_clock::time_point` `qpp::Timer::_end` [`protected`]

7.10.4.2 `std::chrono::steady_clock::time_point` `qpp::Timer::_start` [`protected`]

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

- `include/classes/timer.h`

## 7.11 qpp::UniformIntDistribution Class Reference

```
#include <stat.h>
```

### Public Member Functions

- [UniformIntDistribution](#) (int a=0, int b=1)
- int [sample](#) ()

### Protected Attributes

- std::uniform\_int\_distribution [\\_d](#)

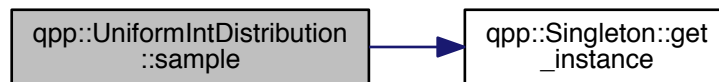
#### 7.11.1 Constructor & Destructor Documentation

7.11.1.1 `qpp::UniformIntDistribution::UniformIntDistribution ( int a = 0, int b = 1 )` `[inline]`

#### 7.11.2 Member Function Documentation

7.11.2.1 `int qpp::UniformIntDistribution::sample ( )` `[inline]`

Here is the call graph for this function:



#### 7.11.3 Member Data Documentation

7.11.3.1 `std::uniform_int_distribution qpp::UniformIntDistribution::_d` `[protected]`

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

- include/classes/[stat.h](#)

## 7.12 qpp::UniformRealDistribution Class Reference

```
#include <stat.h>
```

### Public Member Functions

- [UniformRealDistribution](#) (double a=0, double b=1)
- double [sample](#) ()

## Protected Attributes

- `std::uniform_real_distribution _d`

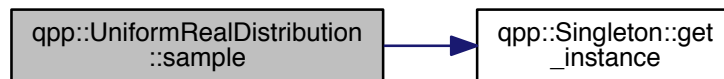
## 7.12.1 Constructor & Destructor Documentation

7.12.1.1 `qpp::UniformRealDistribution::UniformRealDistribution ( double a = 0, double b = 1 )` `[inline]`

## 7.12.2 Member Function Documentation

7.12.2.1 `double qpp::UniformRealDistribution::sample ( )` `[inline]`

Here is the call graph for this function:



## 7.12.3 Member Data Documentation

7.12.3.1 `std::uniform_real_distribution qpp::UniformRealDistribution::_d` `[protected]`

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

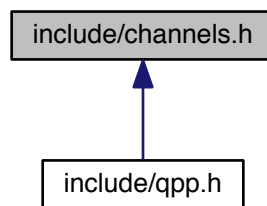
- `include/classes/stat.h`

## Chapter 8

# File Documentation

### 8.1 include/channels.h File Reference

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



#### Namespaces

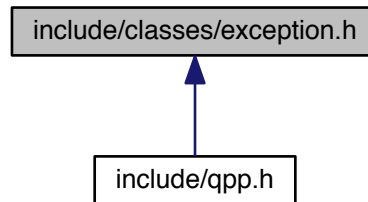
- [qpp](#)

#### Functions

- `cmat qpp::super (const std::vector< cmat > &Ks)`  
*Superoperator matrix representation.*
- `cmat qpp::choi (const std::vector< cmat > &Ks)`  
*Choi matrix representation.*
- `std::vector< cmat > qpp::choi2kraus (const cmat &A)`  
*Extracts orthogonal Kraus operators from Choi matrix.*
- `template<typename Derived >`  
`cmat qpp::channel (const Eigen::MatrixBase< Derived > &rho, const std::vector< cmat > &Ks)`  
*Applies the channel specified by the set of Kraus operators Ks to the density matrix rho.*
- `template<typename Derived >`  
`cmat qpp::channel (const Eigen::MatrixBase< Derived > &rho, const std::vector< cmat > &Ks, const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`  
*Applies the channel specified by the set of Kraus operators Ks to the part of the density matrix rho specified by subsys.*

## 8.2 include/classes/exception.h File Reference

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



### Classes

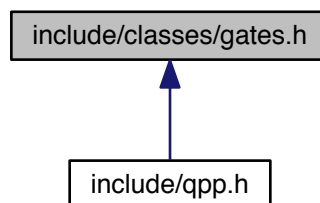
- class [qpp::Exception](#)

### Namespaces

- [qpp](#)

## 8.3 include/classes/gates.h File Reference

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



### Classes

- class [qpp::Gates](#)

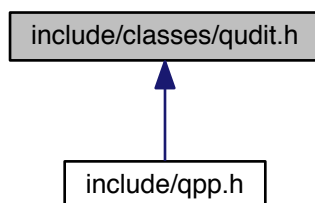
### Namespaces

- [qpp](#)



## 8.4 include/classes/qudit.h File Reference

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



### Classes

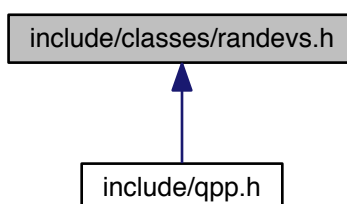
- class [qpp::Qudit](#)

### Namespaces

- [qpp](#)

## 8.5 include/classes/randevs.h File Reference

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



### Classes

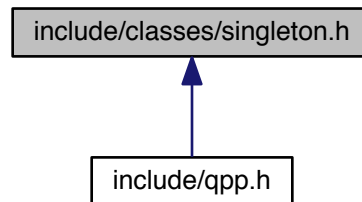
- class [qpp::RandomDevices](#)

### Namespaces

- [qpp](#)

## 8.6 include/classes/singleton.h File Reference

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



### Classes

- class [qpp::Singleton< T >](#)

### Namespaces

- [qpp](#)

### Macros

- [#define CLASS\\_SINGLETON\(Foo\)](#)
- [#define CLASS\\_CONST\\_SINGLETON\(Foo\)](#)

#### 8.6.1 Macro Definition Documentation

##### 8.6.1.1 [#define CLASS\\_CONST\\_SINGLETON\( Foo \)](#)

###### Value:

```

class Foo: public Singleton<const Foo>\
{
    friend class Singleton<const Foo>;
}
  
```

##### 8.6.1.2 [#define CLASS\\_SINGLETON\( Foo \)](#)

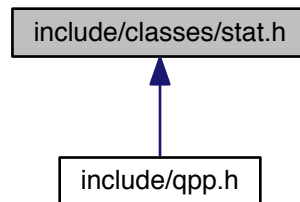
###### Value:

```

class Foo: public Singleton<Foo>\
{
    friend class Singleton<Foo>;
}
  
```

## 8.7 include/classes/stat.h File Reference

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



### Classes

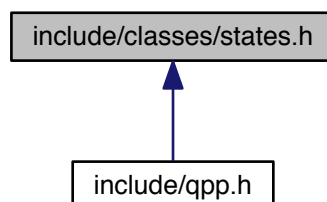
- class [qpp::NormalDistribution](#)
- class [qpp::UniformRealDistribution](#)
- class [qpp::UniformIntDistribution](#)
- class [qpp::DiscreteDistribution](#)
- class [qpp::DiscreteDistributionAbsSquare](#)

### Namespaces

- [qpp](#)

## 8.8 include/classes/states.h File Reference

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



### Classes

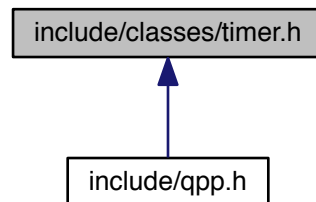
- class [qpp::States](#)

## Namespaces

- [qpp](#)

## 8.9 include/classes/timer.h File Reference

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



## Classes

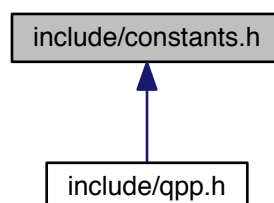
- class [qpp::Timer](#)

## Namespaces

- [qpp](#)

## 8.10 include/constants.h File Reference

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



## Namespaces

- [qpp](#)

## Functions

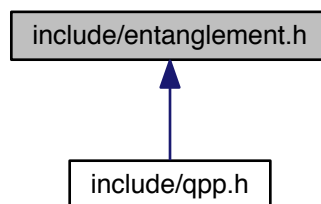
- constexpr std::complex< double > [qpp::operator""\\_i](#) (unsigned long long int x)  
*User-defined literal for complex  $i = \sqrt{-1}$  (integer overload)*
- constexpr std::complex< double > [qpp::operator""\\_i](#) (long double x)  
*User-defined literal for complex  $i = \sqrt{-1}$  (real overload)*
- std::complex< double > [qpp::omega](#) (std::size\_t D)  
*D-th root of unity.*

## Variables

- constexpr double [qpp::chop](#) = 1e-10  
*Used in [qpp::disp\(\)](#) and [qpp::displn\(\)](#) for setting to zero numbers that have their absolute value smaller than [qpp::ct->::chop](#).*
- constexpr double [qpp::eps](#) = 1e-12  
*Used to decide whether a number or expression in double precision is zero or not.*
- constexpr std::size\_t [qpp::maxn](#) = 64  
*Maximum number of qubits.*
- constexpr double [qpp::pi](#) = 3.141592653589793238462643383279502884  
 $\pi$
- constexpr double [qpp::ee](#) = 2.718281828459045235360287471352662497  
*Base of natural logarithm,  $e$ .*

## 8.11 include/entanglement.h File Reference

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



## Namespaces

- [qpp](#)

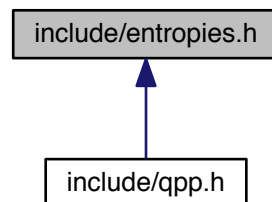
## Functions

- template<typename Derived >  
cmat [qpp::schmidtcoeff](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)
- template<typename Derived >  
cmat [qpp::schmidtU](#) (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size\_t > &dims)

- `template<typename Derived >`  
`cmat qpp::schmidtV (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`
- `template<typename Derived >`  
`cmat qpp::schmidtprob (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`
- `template<typename Derived >`  
`double qpp::entanglement (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`
- `template<typename Derived >`  
`double qpp::gconcurrence (const Eigen::MatrixBase< Derived > &A)`

## 8.12 include/entropies.h File Reference

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



## Namespaces

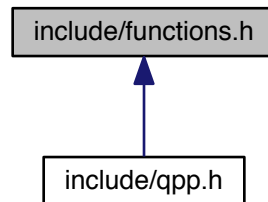
- [qpp](#)

## Functions

- `template<typename Derived >`  
`double qpp::shannon (const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`double qpp::renyi (const double alpha, const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`double qpp::renyi\_inf (const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`double qpp::tsallis (const double alpha, const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`double qpp::qmutualinfo (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`

## 8.13 include/functions.h File Reference

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



### Namespaces

- [qpp](#)

### Functions

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::transpose (const Eigen::MatrixBase< Derived > &A)`  
*Transpose.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::conjugate (const Eigen::MatrixBase< Derived > &A)`  
*Complex conjugate.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::adjoint (const Eigen::MatrixBase< Derived > &A)`  
*Adjoint.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::inverse (const Eigen::MatrixBase< Derived > &A)`  
*Inverse.*
- `template<typename Derived >`  
`Derived::Scalar qpp::trace (const Eigen::MatrixBase< Derived > &A)`  
*Trace.*
- `template<typename Derived >`  
`Derived::Scalar qpp::det (const Eigen::MatrixBase< Derived > &A)`  
*Determinant.*
- `template<typename Derived >`  
`Derived::Scalar qpp::logdet (const Eigen::MatrixBase< Derived > &A)`  
*Logarithm of the determinant.*
- `template<typename Derived >`  
`Derived::Scalar qpp::sum (const Eigen::MatrixBase< Derived > &A)`  
*Element-wise sum.*
- `template<typename Derived >`  
`double qpp::norm (const Eigen::MatrixBase< Derived > &A)`  
*Trace norm.*
- `template<typename Derived >`  
`cmat qpp::evals (const Eigen::MatrixBase< Derived > &A)`

*Eigenvalues.*

- `template<typename Derived >`  
`cmat qpp::evects (const Eigen::MatrixBase< Derived > &A)`

*Eigenvectors.*

- `template<typename Derived >`  
`dmat qpp::hevals (const Eigen::MatrixBase< Derived > &A)`

*Hermitian eigenvalues.*

- `template<typename Derived >`  
`cmat qpp::hevects (const Eigen::MatrixBase< Derived > &A)`

*Hermitian eigenvectors.*

- `template<typename Derived >`  
`cmat qpp::funm (const Eigen::MatrixBase< Derived > &A, cplx(*f)(const cplx &))`

*Functional calculus  $f(A)$* 

- `template<typename Derived >`  
`cmat qpp::sqrtm (const Eigen::MatrixBase< Derived > &A)`

*Matrix square root.*

- `template<typename Derived >`  
`cmat qpp::absm (const Eigen::MatrixBase< Derived > &A)`

*Matrix absolut value.*

- `template<typename Derived >`  
`cmat qpp::expm (const Eigen::MatrixBase< Derived > &A)`

*Matrix exponential.*

- `template<typename Derived >`  
`cmat qpp::logm (const Eigen::MatrixBase< Derived > &A)`

*Matrix logarithm.*

- `template<typename Derived >`  
`cmat qpp::sinm (const Eigen::MatrixBase< Derived > &A)`

*Matrix sin.*

- `template<typename Derived >`  
`cmat qpp::cosm (const Eigen::MatrixBase< Derived > &A)`

*Matrix cos.*

- `template<typename Derived >`  
`cmat qpp::spectralpowm (const Eigen::MatrixBase< Derived > &A, const cplx z)`

*Matrix power.*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::powm (const Eigen::MatrixBase< Derived > &A, std::size_t n)`

*Matrix power.*

- `template<typename OutputScalar , typename Derived >`  
`DynMat< OutputScalar > qpp::cwise (const Eigen::MatrixBase< Derived > &A, OutputScalar(*f)(const type-name Derived::Scalar &))`

*Functor.*

- `template<typename T >`  
`DynMat< typename T::Scalar > qpp::kron (const T &head)`

*Kronecker product (variadic overload)*

- `template<typename T , typename... Args>`  
`DynMat< typename T::Scalar > qpp::kron (const T &head, const Args &...tail)`

*Kronecker product (variadic overload)*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::kron (const std::vector< Derived > &As)`

*Kronecker product (std::vector overload)*

- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::kron (const std::initializer_list< Derived > &As)`

*Kronecker product (std::initializer\_list overload)*

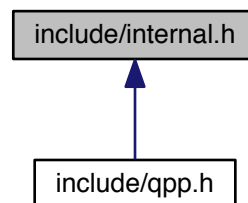


- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::kronpow (const Eigen::MatrixBase< Derived > &A, std::size_t n)`  
*Kronecker power.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::reshape (const Eigen::MatrixBase< Derived > &A, std::size_t rows, std::size_t cols)`  
*Reshape.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::syspermute (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &perm, const std::vector< std::size_t > &dims)`  
*System permutation.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::ptrace1 (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`  
*Partial trace.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::ptrace2 (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &dims)`  
*Partial trace.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::ptrace (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`  
*Partial trace.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::ptranspose (const Eigen::MatrixBase< Derived > &A, const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`  
*Partial transpose.*
- `template<typename Derived1, typename Derived2 >`  
`DynMat< typename Derived1::Scalar > qpp::comm (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)`  
*Commutator.*
- `template<typename Derived1, typename Derived2 >`  
`DynMat< typename Derived1::Scalar > qpp::anticomm (const Eigen::MatrixBase< Derived1 > &A, const Eigen::MatrixBase< Derived2 > &B)`  
*Anti-commutator.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::prj (const Eigen::MatrixBase< Derived > &V)`  
*Projector.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::expandout (const Eigen::MatrixBase< Derived > &A, std::size_t pos, const std::vector< std::size_t > &dims)`  
*Expand out.*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::grams (const std::vector< Derived > &Vs)`  
*Gram-Schmidt orthogonalization (std::vector overload)*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::grams (const std::initializer_list< Derived > &Vs)`  
*Gram-Schmidt orthogonalization (std::initializer\_list overload)*
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::grams (const Eigen::MatrixBase< Derived > &A)`  
*Gram-Schmidt orthogonalization (Eigen expression (matrix) overload)*
- `std::vector< std::size_t > qpp::n2multiidx (std::size_t n, const std::vector< std::size_t > &dims)`  
*Non-negative integer index to multi-index.*

- `std::size_t qpp::multiidx2n` (const std::vector< std::size\_t > &midx, const std::vector< std::size\_t > &dims)  
*Multi-index to non-negative integer index.*
- `ket qpp::mket` (const std::vector< std::size\_t > &mask)  
*Multi-partite qubit ket.*
- `ket qpp::mket` (const std::vector< std::size\_t > &mask, const std::vector< std::size\_t > &dims)  
*Multi-partite qudit ket (different dimensions overload)*
- `ket qpp::mket` (const std::vector< std::size\_t > &mask, std::size\_t d)  
*Multi-partite qudit ket (same dimensions overload)*
- `std::vector< std::size_t > qpp::invperm` (const std::vector< std::size\_t > &perm)  
*Inverse permutation.*
- `std::vector< std::size_t > qpp::compperm` (const std::vector< std::size\_t > &perm, const std::vector< std::size\_t > &sigma)  
*Compose permutations.*

## 8.14 include/internal.h File Reference

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



## Namespaces

- `qpp::internal`
- `qpp`

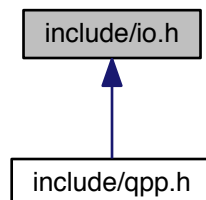
## Functions

- `void qpp::internal::_n2multiidx` (std::size\_t n, std::size\_t numdims, const std::size\_t \*dims, std::size\_t \*result)
- `std::size_t qpp::internal::_multiidx2n` (const std::size\_t \*midx, std::size\_t numdims, const std::size\_t \*dims)
- `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_row_vector` (const Eigen::MatrixBase< Derived > &A)
- `template<typename Derived >`  
`bool qpp::internal::_check_col_vector` (const Eigen::MatrixBase< Derived > &A)
- `template<typename T >`  
`bool qpp::internal::_check_nonzero_size` (const T &x)
- `bool qpp::internal::_check_dims` (const std::vector< std::size\_t > &dims)

- `template<typename Derived >`  
`bool qpp::internal::\_check\_dims\_match\_mat (const std::vector< std::size_t > &dims, const Eigen::MatrixBase< Derived > &A)`
- `template<typename Derived >`  
`bool qpp::internal::\_check\_dims\_match\_cvect (const std::vector< std::size_t > &dims, const Eigen::MatrixBase< Derived > &V)`
- `template<typename Derived >`  
`bool qpp::internal::\_check\_dims\_match\_rvect (const std::vector< std::size_t > &dims, const Eigen::MatrixBase< Derived > &V)`
- `bool qpp::internal::\_check\_eq\_dims (const std::vector< std::size_t > &dims, std::size_t dim)`
- `bool qpp::internal::\_check\_subsys\_match\_dims (const std::vector< std::size_t > &subsys, const std::vector< std::size_t > &dims)`
- `bool qpp::internal::\_check\_perm (const std::vector< std::size_t > &perm)`
- `template<typename Derived1 , typename Derived2 >`  
`DynMat< typename Derived1::Scalar > qpp::internal::\_kron2 (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)`

## 8.15 include/io.h File Reference

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



### Namespaces

- [qpp](#)

### Functions

- `template<typename T >`  
`void qpp::disp (const T &x, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename T >`  
`void qpp::displin (const T &x, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename T >`  
`void qpp::disp (const T *x, const std::size_t n, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`

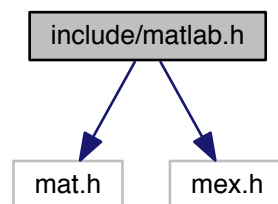
- `template<typename T >`  
`void qpp::displn (const T *x, const std::size_t n, const std::string &separator, const std::string &start="[" , const std::string &end="]", std::ostream &os=std::cout)`
- `template<typename Derived >`  
`void qpp::disp (const Eigen::MatrixBase< Derived > &A, double chop=chop, std::ostream &os=std::cout)`
- `template<typename Derived >`  
`void qpp::displn (const Eigen::MatrixBase< Derived > &A, double chop=chop, std::ostream &os=std::cout)`
- `void qpp::disp (const cplx c, double chop=chop, std::ostream &os=std::cout)`
- `void qpp::displn (const cplx c, double chop=chop, std::ostream &os=std::cout)`
- `template<typename Derived >`  
`void qpp::save (const Eigen::MatrixBase< Derived > &A, const std::string &fname)`
- `template<typename Derived >`  
`DynMat< typename Derived::Scalar > qpp::load (const std::string &fname)`

## 8.16 include/matlab.h File Reference

```
#include "mat.h"
```

```
#include "mex.h"
```

Include dependency graph for matlab.h:



## Namespaces

- [qpp](#)

## Functions

- `template<typename Derived >`  
`Derived qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<>`  
`dmat qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<>`  
`cmat qpp::loadMATLABmatrix (const std::string &mat_file, const std::string &var_name)`
- `template<typename Derived >`  
`void qpp::saveMATLABmatrix (const Eigen::MatrixBase< Derived > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)`
- `template<>`  
`void qpp::saveMATLABmatrix (const Eigen::MatrixBase< dmat > &A, const std::string &mat_file, const std::string &var_name, const std::string &mode)`

- `template<>`  
void [qpp::saveMATLABmatrix](#) (const Eigen::MatrixBase< cmat > &A, const std::string &mat\_file, const std::string &var\_name, const std::string &mode)

## 8.17 include/qpp.h File Reference

```
#include <algorithm>
#include <chrono>
#include <cmath>
#include <complex>
#include <cstdlib>
#include <cstring>
#include <exception>
#include <fstream>
#include <functional>
#include <iomanip>
#include <iostream>
#include <iterator>
#include <numeric>
#include <ostream>
#include <random>
#include <stdexcept>
#include <string>
#include <type_traits>
#include <utility>
#include <vector>
#include <Eigen/Dense>
#include <Eigen/SVD>
#include "constants.h"
#include "types.h"
#include "classes/exception.h"
#include "classes/singleton.h"
#include "classes/states.h"
#include "classes/randevs.h"
#include "internal.h"
#include "functions.h"
#include "classes/gates.h"
#include "classes/stat.h"
#include "entropies.h"
#include "entanglement.h"
#include "channels.h"
#include "io.h"
#include "random.h"
#include "classes/qudit.h"
#include "classes/timer.h"
```

Include dependency graph for qpp.h:



### Namespaces

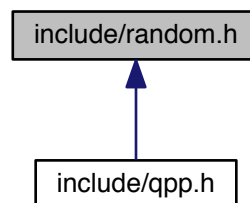
- [qpp](#)

## Variables

- RandomDevices & `qpp::rdevs` = RandomDevices::get\_instance()  
*qpp::RandomDevices Singleton*
- const Gates & `qpp::gt` = Gates::get\_instance()  
*qpp::Gates const Singleton*
- const States & `qpp::st` = States::get\_instance()  
*qpp::States const Singleton*

## 8.18 include/random.h File Reference

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



## Namespaces

- `qpp`

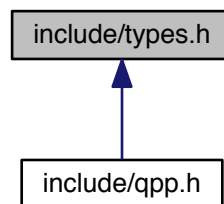
## Functions

- template<typename Derived >  
Derived `qpp::rand` (std::size\_t rows, std::size\_t cols, double a=0, double b=1)
- template<>  
dmat `qpp::rand` (std::size\_t rows, std::size\_t cols, double a, double b)
- template<>  
cmat `qpp::rand` (std::size\_t rows, std::size\_t cols, double a, double b)
- double `qpp::rand` (double a=0, double b=1)
- long long `qpp::randint` (long long a, long long b)
- template<typename Derived >  
Derived `qpp::randn` (std::size\_t rows, std::size\_t cols, double mean=0, double sigma=1)
- template<>  
dmat `qpp::randn` (std::size\_t rows, std::size\_t cols, double mean, double sigma)
- template<>  
cmat `qpp::randn` (std::size\_t rows, std::size\_t cols, double mean, double sigma)
- double `qpp::randn` (double mean=0, double sigma=1)
- cmatrix `qpp::randU` (std::size\_t D)
- cmatrix `qpp::randV` (std::size\_t Din, std::size\_t Dout)
- std::vector< cmatrix > `qpp::randkraus` (std::size\_t n, std::size\_t D)
- cmatrix `qpp::randH` (std::size\_t D)

- ket [qpp::randket](#) (std::size\_t D)
- cmat [qpp::randrho](#) (std::size\_t D)
- std::vector< std::size\_t > [qpp::randperm](#) (std::size\_t n)

## 8.19 include/types.h File Reference

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



### Namespaces

- [qpp](#)

### Typedefs

- using [qpp::cplx](#) = std::complex< double >  
*Complex number in double precision.*
- using [qpp::cmat](#) = Eigen::MatrixXcd  
*Complex (double precision) dynamic Eigen matrix.*
- using [qpp::dmat](#) = Eigen::MatrixXd  
*Real (double precision) dynamic Eigen matrix.*
- using [qpp::ket](#) = Eigen::Matrix< cplx, Eigen::Dynamic, 1 >  
*Complex (double precision) dynamic Eigen column matrix.*
- using [qpp::bra](#) = Eigen::Matrix< cplx, 1, Eigen::Dynamic >  
*Complex (double precision) dynamic Eigen row matrix.*
- template<typename Scalar >  
using [qpp::DynMat](#) = Eigen::Matrix< Scalar, Eigen::Dynamic, Eigen::Dynamic >  
*Dynamic Eigen matrix over the field specified by Scalar.*

# Index

absm  
    qpp, 17  
adjoint  
    qpp, 18  
anticomm  
    qpp, 18  
  
bra  
    qpp, 17  
  
CUSTOM\_EXCEPTION  
    qpp::Exception, 73  
channel  
    qpp, 19  
choi  
    qpp, 21  
choi2kraus  
    qpp, 22  
chop  
    qpp, 65  
cmat  
    qpp, 17  
comm  
    qpp, 23  
compperm  
    qpp, 23  
conjugate  
    qpp, 25  
cosm  
    qpp, 25  
cplx  
    qpp, 17  
cwise  
    qpp, 26  
  
DIMS\_INVALID  
    qpp::Exception, 73  
DIMS\_MISMATCH\_CVECTOR  
    qpp::Exception, 73  
DIMS\_MISMATCH\_MATRIX  
    qpp::Exception, 73  
DIMS\_MISMATCH\_RVECTOR  
    qpp::Exception, 73  
DIMS\_MISMATCH\_VECTOR  
    qpp::Exception, 73  
DIMS\_NOT\_EQUAL  
    qpp::Exception, 73  
det  
    qpp, 26  
disp  
    qpp, 27  
displn  
    qpp, 27, 28  
dmat  
    qpp, 17  
  
ee  
    qpp, 65  
entanglement  
    qpp, 29  
eps  
    qpp, 65  
evals  
    qpp, 29  
evects  
    qpp, 30  
expandout  
    qpp, 30  
expm  
    qpp, 31  
  
funm  
    qpp, 32  
  
gconcurrency  
    qpp, 32  
grams  
    qpp, 33, 34  
gt  
    qpp, 65  
  
hevals  
    qpp, 34  
hevects  
    qpp, 35  
  
inverse  
    qpp, 35  
invperm  
    qpp, 37  
  
ket  
    qpp, 17  
kron  
    qpp, 37–39  
kronpow  
    qpp, 39  
  
load  
    qpp, 40  
logdet



- qpp, [40](#)
- logm
  - qpp, [41](#)
- MATRIX\_NOT\_CVECTOR
  - qpp::Exception, [73](#)
- MATRIX\_NOT\_RVECTOR
  - qpp::Exception, [73](#)
- MATRIX\_NOT\_SQUARE
  - qpp::Exception, [73](#)
- MATRIX\_NOT\_SQUARE\_OR\_CVECTOR
  - qpp::Exception, [73](#)
- MATRIX\_NOT\_SQUARE\_OR\_RVECTOR
  - qpp::Exception, [73](#)
- MATRIX\_NOT\_SQUARE\_OR\_VECTOR
  - qpp::Exception, [73](#)
- MATRIX\_NOT\_VECTOR
  - qpp::Exception, [73](#)
- maxn
  - qpp, [65](#)
- mket
  - qpp, [41](#), [42](#)
- multiidx2n
  - qpp, [43](#)
- n2multiidx
  - qpp, [43](#)
- NOT\_BIPARTITE
  - qpp::Exception, [73](#)
- NOT\_QUBIT\_GATE
  - qpp::Exception, [73](#)
- NOT\_QUBIT\_SUBSYS
  - qpp::Exception, [73](#)
- norm
  - qpp, [44](#)
- OUT\_OF\_RANGE
  - qpp::Exception, [73](#)
- omega
  - qpp, [44](#)
- PERM\_INVALID
  - qpp::Exception, [73](#)
- pi
  - qpp, [65](#)
- powm
  - qpp, [45](#)
- prj
  - qpp, [45](#)
- ptrace
  - qpp, [46](#)
- ptrace1
  - qpp, [47](#)
- ptrace2
  - qpp, [48](#)
- ptranspose
  - qpp, [49](#)
- qmutualinfo
  - qpp, [50](#)
- qpp, [11](#)
  - absm, [17](#)
  - adjoint, [18](#)
  - anticomm, [18](#)
  - bra, [17](#)
  - channel, [19](#)
  - choi, [21](#)
  - choi2kraus, [22](#)
  - chop, [65](#)
  - cmat, [17](#)
  - comm, [23](#)
  - compperm, [23](#)
  - conjugate, [25](#)
  - cosm, [25](#)
  - cplx, [17](#)
  - cwise, [26](#)
  - det, [26](#)
  - disp, [27](#)
  - displn, [27](#), [28](#)
  - dmat, [17](#)
  - ee, [65](#)
  - entanglement, [29](#)
  - eps, [65](#)
  - evals, [29](#)
  - evects, [30](#)
  - expandout, [30](#)
  - expm, [31](#)
  - funm, [32](#)
  - gconcurrence, [32](#)
  - grams, [33](#), [34](#)
  - gt, [65](#)
  - hevals, [34](#)
  - hevects, [35](#)
  - inverse, [35](#)
  - invperm, [37](#)
  - ket, [17](#)
  - kron, [37–39](#)
  - kronpow, [39](#)
  - load, [40](#)
  - logdet, [40](#)
  - logm, [41](#)
  - maxn, [65](#)
  - mket, [41](#), [42](#)
  - multiidx2n, [43](#)
  - n2multiidx, [43](#)
  - norm, [44](#)
  - omega, [44](#)
  - pi, [65](#)
  - powm, [45](#)
  - prj, [45](#)
  - ptrace, [46](#)
  - ptrace1, [47](#)
  - ptrace2, [48](#)
  - ptranspose, [49](#)
  - qmutualinfo, [50](#)
  - rand, [51](#), [52](#)
  - randint, [52](#)

- randket, 53
- randkraus, 53
- randn, 53, 54
- randperm, 54
- randrho, 54
- rdevs, 65
- renyi, 55
- reshape, 56
- save, 56
- schmidtcoeff, 57
- schmidtprob, 58
- shannon, 59
- sinm, 59
- spectralpowm, 60
- sqrtn, 60
- st, 66
- sum, 61
- super, 61
- syspermute, 62
- trace, 63
- transpose, 64
- tsallis, 64
- qpp::Exception
  - CUSTOM\_EXCEPTION, 73
  - DIMS\_INVALID, 73
  - DIMS\_MISMATCH\_CVECTOR, 73
  - DIMS\_MISMATCH\_MATRIX, 73
  - DIMS\_MISMATCH\_RVECTOR, 73
  - DIMS\_MISMATCH\_VECTOR, 73
  - DIMS\_NOT\_EQUAL, 73
  - MATRIX\_NOT\_CVECTOR, 73
  - MATRIX\_NOT\_RVECTOR, 73
  - MATRIX\_NOT\_SQUARE, 73
  - MATRIX\_NOT\_SQUARE\_OR\_CVECTOR, 73
  - MATRIX\_NOT\_SQUARE\_OR\_RVECTOR, 73
  - MATRIX\_NOT\_SQUARE\_OR\_VECTOR, 73
  - MATRIX\_NOT\_VECTOR, 73
  - NOT\_BIPARTITE, 73
  - NOT\_QUBIT\_GATE, 73
  - NOT\_QUBIT\_SUBSYS, 73
  - OUT\_OF\_RANGE, 73
  - PERM\_INVALID, 73
  - SUBSYS\_MISMATCH\_DIMS, 73
  - TYPE\_MISMATCH, 73
  - UNDEFINED\_TYPE, 73
  - UNKNOWN\_EXCEPTION, 73
  - ZERO\_SIZE, 73
- rand
  - qpp, 51, 52
- randint
  - qpp, 52
- randket
  - qpp, 53
- randkraus
  - qpp, 53
- randn
  - qpp, 53, 54
- randperm
  - qpp, 54
- randrho
  - qpp, 54
- rdevs
  - qpp, 65
- renyi
  - qpp, 55
- reshape
  - qpp, 56
- SUBSYS\_MISMATCH\_DIMS
  - qpp::Exception, 73
- save
  - qpp, 56
- schmidtcoeff
  - qpp, 57
- schmidtprob
  - qpp, 58
- shannon
  - qpp, 59
- sinm
  - qpp, 59
- spectralpowm
  - qpp, 60
- sqrtn
  - qpp, 60
- st
  - qpp, 66
- sum
  - qpp, 61
- super
  - qpp, 61
- syspermute
  - qpp, 62
- TYPE\_MISMATCH
  - qpp::Exception, 73
- trace
  - qpp, 63
- transpose
  - qpp, 64
- tsallis
  - qpp, 64
- UNDEFINED\_TYPE
  - qpp::Exception, 73
- UNKNOWN\_EXCEPTION
  - qpp::Exception, 73
- ZERO\_SIZE
  - qpp::Exception, 73