quantum++ 0.1

Generated by Doxygen 1.8.7

Thu Oct 23 2014 22:47:43

## **Contents**

1	qua	ntum++	- A C++1	1 quantum computing library	1
2	Nan	nespace			5
	2.1	Names	space List		5
3	Hier	archica	Index		7
	3.1	Class I	Hierarchy		7
4	Clas	ss Index			9
	4.1	Class I	_ist		9
5	File	Index			11
	5.1	File Lis	st		11
6	Nam	nespace	Docume	ntation	13
	6.1	qpp Na	amespace	Reference	13
		6.1.1	Typedef	Documentation	19
			6.1.1.1	bra	19
			6.1.1.2	cmat	19
			6.1.1.3	cplx	19
			6.1.1.4	dmat	19
			6.1.1.5	DynMat	19
			6.1.1.6	ket	19
		6.1.2	Function	Documentation	19
			6.1.2.1	absm	19
			6.1.2.2	adjoint	20
			6.1.2.3	anticomm	20
			6.1.2.4	channel	21
			6.1.2.5	channel	22
			6.1.2.6	choi	23
			6.1.2.7	choi2kraus	24
			6.1.2.8	comm	25
			6.1.2.9	compperm	26

iv CONTENTS

6.1.2.10	conjugate	27
6.1.2.11	cosm	27
6.1.2.12	cwise	28
6.1.2.13	det	28
6.1.2.14	disp	29
6.1.2.15	disp	29
6.1.2.16	disp	29
6.1.2.17	disp	29
6.1.2.18	displn	30
6.1.2.19	displn	30
6.1.2.20	displn	30
6.1.2.21	displn	31
6.1.2.22	entanglement	31
6.1.2.23	evals	31
6.1.2.24	evects	32
6.1.2.25	expandout	33
6.1.2.26	expm	33
6.1.2.27	funm	34
6.1.2.28	gconcurrence	35
6.1.2.29	grams	35
6.1.2.30	grams	35
6.1.2.31	grams	36
6.1.2.32	hevals	36
6.1.2.33	hevects	37
6.1.2.34	inverse	38
6.1.2.35	invperm	39
6.1.2.36	kron	39
6.1.2.37	kron	40
6.1.2.38	kron	40
6.1.2.39	kron	41
6.1.2.40	kronpow	41
6.1.2.41	load	42
6.1.2.42	loadMATLABmatrix	42
6.1.2.43	loadMATLABmatrix	42
6.1.2.44	loadMATLABmatrix	42
6.1.2.45	logdet	42
6.1.2.46	logm	43
6.1.2.47	mket	43
6.1.2.48	mket	44
6.1.2.49	mket	44

CONTENTS

6.1.2.50	multiidx2n	45
6.1.2.51	n2multiidx	45
6.1.2.52	norm	46
6.1.2.53	omega	46
6.1.2.54	operator""""_i	47
6.1.2.55	operator""""_i	47
6.1.2.56	powm	47
6.1.2.57	prj	48
6.1.2.58	ptrace	48
6.1.2.59	ptrace1	49
6.1.2.60	ptrace2	50
6.1.2.61	ptranspose	51
6.1.2.62	qmutualinfo	53
6.1.2.63	rand	53
6.1.2.64	rand	53
6.1.2.65	rand	54
6.1.2.66	rand	54
6.1.2.67	randH	54
6.1.2.68	randint	54
6.1.2.69	randket	55
6.1.2.70	randkraus	55
6.1.2.71	randn	55
6.1.2.72	randn	55
6.1.2.73	randn	56
6.1.2.74	randn	56
6.1.2.75	randperm	56
6.1.2.76	randrho	57
6.1.2.77	randU	57
6.1.2.78	randV	57
6.1.2.79	renyi	57
6.1.2.80	renyi_inf	58
6.1.2.81	reshape	58
6.1.2.82	save	58
6.1.2.83	saveMATLABmatrix	58
6.1.2.84	saveMATLABmatrix	59
6.1.2.85	saveMATLABmatrix	59
6.1.2.86	schmidtcoeff	59
6.1.2.87	schmidtprob	60
6.1.2.88	schmidtU	60
6.1.2.89	schmidtV	61

vi CONTENTS

		6.1.2.90	shannon	61
		6.1.2.91	sinm	61
		6.1.2.92	spectralpowm	62
		6.1.2.93	sqrtm	62
		6.1.2.94	sum	63
		6.1.2.95	super	63
		6.1.2.96	syspermute	64
		6.1.2.97	trace	65
		6.1.2.98	transpose	66
		6.1.2.99	tsallis	67
	6.1.3	Variable I	Documentation	67
		6.1.3.1	chop	67
		6.1.3.2	ee	67
		6.1.3.3	eps	67
		6.1.3.4	gt	67
		6.1.3.5	maxn	67
		6.1.3.6	pi	67
		6.1.3.7	rdevs	68
		6.1.3.8	st	68
6.2	qpp::in	ternal Nan	nespace Reference	68
	6.2.1	Detailed	Description	68
	6.2.2	Function	Documentation	69
		6.2.2.1	_check_col_vector	69
		6.2.2.2	_check_dims	69
		6.2.2.3	_check_dims_match_cvect	69
		6.2.2.4	_check_dims_match_mat	69
		6.2.2.5	_check_dims_match_rvect	69
		6.2.2.6	_check_eq_dims	69
		6.2.2.7	_check_nonzero_size	69
		6.2.2.8	_check_perm	69
		6.2.2.9	_check_row_vector	69
		6.2.2.10	_check_square_mat	69
		6.2.2.11	_check_subsys_match_dims	69
		6.2.2.12	_check_vector	69
		6.2.2.13	_kron2	69
		6.2.2.14	_multiidx2n	69
		6.2.2.15	_n2multiidx	69
		6.2.2.16	variadic_vector_emplace	69
		6.2.2.17	variadic_vector_emplace	70

CONTENTS vii

7	Clas	s Docu	nentation 7	1
	7.1	qpp::D	screteDistribution Class Reference	'1
		7.1.1	Constructor & Destructor Documentation	'1
			7.1.1.1 Discrete Distribution	1
			7.1.1.2 DiscreteDistribution	1
			7.1.1.3 DiscreteDistribution	1
		7.1.2	Member Function Documentation	'1
			7.1.2.1 probabilities	1
			7.1.2.2 sample	2
		7.1.3	Member Data Documentation	2
			7.1.3.1 _d	2
	7.2	qpp::D	screteDistributionAbsSquare Class Reference	2
		7.2.1	Constructor & Destructor Documentation	'3
			7.2.1.1 DiscreteDistributionAbsSquare	'3
			7.2.1.2 DiscreteDistributionAbsSquare	'3
			7.2.1.3 DiscreteDistributionAbsSquare	'3
			7.2.1.4 DiscreteDistributionAbsSquare	'3
		7.2.2	Member Function Documentation	'3
			7.2.2.1 cplx2weights	'3
			7.2.2.2 probabilities	'3
			7.2.2.3 sample	'3
		7.2.3	Member Data Documentation	'3
			7.2.3.1 _d	'3
	7.3	qpp::E	cception Class Reference	'3
		7.3.1	Member Enumeration Documentation	'5
			7.3.1.1 Type	'5
		7.3.2	Constructor & Destructor Documentation	6'
			7.3.2.1 Exception	6
			7.3.2.2 Exception	6
		7.3.3	Member Function Documentation	6
			7.3.3.1 _construct_exception_msg	6
			7.3.3.2 what	6'
		7.3.4	Member Data Documentation	6
			7.3.4.1 _custom	6'
			7.3.4.2 _msg	6'
			7.3.4.3 _type	6
			7.3.4.4 _where	6'
	7.4	qpp::G	ates Class Reference	6
		7.4.1	Constructor & Destructor Documentation	'8
			7.4.1.1 Gates	8'

viii CONTENTS

	7.4.2	Member Function Documentation
		7.4.2.1 apply
		7.4.2.2 applyCTRL
		7.4.2.3 CTRL
		7.4.2.4 Fd
		7.4.2.5 ld
		7.4.2.6 Rn
		7.4.2.7 Xd
		7.4.2.8 Zd
	7.4.3	Friends And Related Function Documentation
		7.4.3.1 Singleton < const Gates >
	7.4.4	Member Data Documentation
		7.4.4.1 CNOTab
		7.4.4.2 CNOTba
		7.4.4.3 CZ
		7.4.4.4 FRED
		7.4.4.5 H
		7.4.4.6 ld2
		7.4.4.7 S
		7.4.4.8 SWAP
		7.4.4.9 T
		7.4.4.10 TOF
		7.4.4.11 X
		7.4.4.12 Y
		7.4.4.13 Z
7.5	qpp::N	rmalDistribution Class Reference
	7.5.1	Constructor & Destructor Documentation
		7.5.1.1 NormalDistribution
	7.5.2	Member Function Documentation
		7.5.2.1 sample
	7.5.3	Member Data Documentation
		7.5.3.1 _d
7.6	qpp::Q	dit Class Reference
	7.6.1	Constructor & Destructor Documentation
		7.6.1.1 Qudit
	7.6.2	Member Function Documentation
		7.6.2.1 getD
		7.6.2.2 getRho
		7.6.2.3 measure
		7.6.2.4 measure

CONTENTS

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

CONTENTS

		7.9.3.16 pz0
		7.9.3.17 pz1
		7.9.3.18 W
		7.9.3.19 x0
		7.9.3.20 x1
		7.9.3.21 y0
		7.9.3.22 y1
		7.9.3.23 z0
		7.9.3.24 z1
7.10	qpp::Ti	mer Class Reference
	7.10.1	Constructor & Destructor Documentation
		7.10.1.1 Timer
	7.10.2	Member Function Documentation
		7.10.2.1 seconds
		7.10.2.2 tic
		7.10.2.3 toc
	7.10.3	Friends And Related Function Documentation
		7.10.3.1 operator<< 90
	7.10.4	Member Data Documentation
		7.10.4.1 _end
		7.10.4.2 _start
7.11	qpp::Ur	niformIntDistribution Class Reference
	7.11.1	Constructor & Destructor Documentation
		7.11.1.1 UniformIntDistribution
	7.11.2	Member Function Documentation
		7.11.2.1 sample
	7.11.3	Member Data Documentation
		7.11.3.1 _d
7.12	qpp::Ur	niformRealDistribution Class Reference
	7.12.1	Constructor & Destructor Documentation
		7.12.1.1 UniformRealDistribution
	7.12.2	Member Function Documentation
		7.12.2.1 sample
	7.12.3	Member Data Documentation
		7.12.3.1 _d
File	Docume	entation 93
8.1	include	/channels.h File Reference
8.2	include	/classes/exception.h File Reference
8.3	include	/classes/gates.h File Reference

8

CONTENTS xi

8	3.4	include	e/classes/c	judit.h File	Referen	ce .			 	 	 	 	 	 	 95
8	3.5	include	e/classes/r	andevs.h F	ile Refe	rence			 	 	 	 	 	 	 95
8	3.6	include	e/classes/s	ingleton.h	File Ref	erence	е		 	 	 	 	 	 	 96
		8.6.1	Macro D	efinition Do	cument	ation			 	 	 	 	 	 	 96
			8.6.1.1	CLASS_	CONST_	SING	SLETO	N .	 	 	 	 	 	 	 96
			8.6.1.2	CLASS_	SINGLE	TON			 	 	 	 	 	 	 96
8	3.7	include	e/classes/s	tat.h File F	Referenc	е			 	 	 	 	 	 	 97
8	8.8	include	e/classes/s	tates.h File	e Refere	nce .			 	 	 	 	 	 	 97
8	3.9	include	e/classes/ti	imer.h File	Referen	ce .			 	 	 	 	 	 	 98
8	3.10	include	e/constants	s.h File Re	ference				 	 	 	 	 	 	 98
8	3.11	include	e/entanglei	ment.h File	Referer	nce .			 	 	 	 	 	 	 99
8	3.12	include	e/entropies	h File Ref	erence				 	 	 	 	 	 	 100
8	3.13	include	e/functions	.h File Ref	erence				 	 	 	 	 	 	 101
8	3.14	include	e/internal.h	File Refer	ence .				 	 	 	 	 	 	 104
8	3.15	include	e/io.h File F	Reference					 	 	 	 	 	 	 105
8	3.16	include	e/matlab.h	File Refere	ence				 	 	 	 	 	 	 106
8	3.17	include	e/qpp.h File	e Referenc	e				 	 	 	 	 	 	 107
8	3.18	include	e/random.h	r File Refer	rence .				 	 	 	 	 	 	 108
8	3.19	include	e/types.h F	ile Referer	nce				 	 	 	 	 		 109
Inde	X														110

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

Version

0.1

**Author** 

Vlad Gheorghiu

Date

24 October 2014

#### A simple example:

```
#include "qpp.h"
//#include "matlab.h" // support for MATLAB
using namespace std;
using namespace qpp;
cplx pow3(const cplx& z) // a test function
    return std::pow(z, 3);
int main()
    cout << "Starting qpp..." << endl;</pre>
     // output format
     //cout << std::scientific;</pre>
    cout << std::fixed; // use fixed format for nice formatting</pre>
    cout << std::setprecision(4); // only for fixed or scientific modes</pre>
     // TESTING
     // testing channel and Gates::apply
     cout << endl << "Testing channel(...) and Gates::apply(...)." << endl;</pre>
     cmat rho = randrho(16);
cmat K = kron(gt.Id2, gt.X, gt.Y, gt.Z);
    vector<std::size_t> p = randperm(4); // permutation
cout << "Permutation: ";</pre>
     displn(p, ", ");
    vector<std::size_t> invp = invperm(p); // inverse permutation
cout << "Inverse permutation: ";
displn(invp, ", ");</pre>
     cmat r1 = channel(rho, { K }, p, { 2, 2, 2, 2 });
     cmat r2 = syspermute(channel(syspermute(rho, p, { 2, 2, 2 }), { K },
      { 0,
    1, 2, 3 }, { 2, 2, 2, 2 }), invp, { 2, 2, 2, 2 }); cout << norm(r1 - r2) << endl << endl;
     r1 = gt.apply(rho, K, p, { 2, 2, 2, 2 });
     gt.apply(syspermute(rho, p, { 2, 2, 2, 2 }), K, { 0, 1, 2, 3 }, { 2, 2, 2, 2 }), invp, { 2, 2, 2, 2 });
cout << norm(r1 - r2) << endl << endl;
```

```
displn(channel(prj(mket( { 0, 1 })), { gt.CNOTab }, { 1, 0 }, { 2, 2 }));
displn(gt.apply(mket( { 0, 0 }), gt.CNOTab, { 0, 1 }, { 2, 2 }));
// quantum teleportation
cout << endl << "Qudit teleportation." << endl;</pre>
ket psi = randket(2); // a random state;
cout << "|psi><psi|:" << endl;</pre>
ket psiout = telecircuit * psiin; // output state before measurement
// measure Alice's qubits, measurement results are 1 0
psiout = kron(prj(st.z1), prj(st.z0), gt.Id2) * psiout;
// apply correction
psiout = expandout(powm(gt.Z, 1) * powm(gt.X, 0), { 2 }, { 2, 2, 2 })
           psiout;
// not necessary to normalize, prj() takes care of it below
cmat rhoout = ptrace(prj(psiout), { 0, 1 }, { 2, 2, 2 });
cout << endl << "Teleported state:" << endl;</pre>
displn(rhoout);
cout << "Difference in norm: " << norm(prj(psi) - rhoout) << endl;</pre>
// qudit measurements
cout << endl << "Qudit measurements." << endl;</pre>
cout << "Initially in state |0><0|." << endl;</pre>
ket zd0(3);
zd0 << 1, 0, 0;
Qudit q(prj(zd0));
cout << "Measuring Z operator non-destructively. Results:" << endl;</pre>
cout << q.measure() << endl;</pre>
cout << q.measure() << endl;</pre>
cout << q.measure() << endl;
cout << q.measureing X operator non-destructively. Results:" << endl;
cout << q.measure(gt.Xd(3)) << endl;
cout << q.measure(gt.Xd(3)) << endl;</pre>
cout << q.measure(gt.Xd(3)) << endl;</pre>
// von Neumann projective measurement
cout << "Measuring X operator destructively (collapse). Results:" << endl;
cout << q.measure(gt.Xd(3), true) << endl;
cout << q.measure(gt.Xd(3)) << endl;</pre>
cout << q.measure(gt.Xd(3)) << endl;
cout << g.measure(gt.Xd(3)) << endl;
cout << "Finally measuring Z operator destructively. Results:" << endl;</pre>
cout << q.measure(true) << endl;</pre>
cout << q.measure() << endl;</pre>
cout << q.measure() << endl;
cout << "Final state of qudit:" << endl;</pre>
displn(q.getRho());
// Bell state generator
cout << endl << "Bell state generator: " << endl;</pre>
cmat circuit;
circuit = gt.CTRL(gt.X, { 0 }, { 1 }, 2) * expandout(gt.
    H, 0, { 2, 2 });
cmat input = kron(st.z0, st.z0);
cmat output = circuit * input;
cout << "Circuit matrix representation: " << endl;</pre>
displn(circuit);
cout << endl << "Output (|Bell_0> state) of the circuit on |00>: " << endl;</pre>
displn(output);
// 3-qubit repetion code
cout << endl << "3-qubit repetition code: " << endl;</pre>
cmat rep;
rep = gt.CTRL(gt.X, { 0 }, { 2 }, 3) * gt.CTRL(gt.X, { 0 }, { 1 }, 3);
input = kron(st.z1, st.z0, st.z0);
output = rep * input;
cout << "Circuit acting on |000> produces |111>. Check: " << endl;
displn(output);
// functor test
cout << endl << "Functor z^3 acting on:" << endl;</pre>
cmat a(2, 2);
a << 1, 2, 3, 4;</pre>
displn(a);
cout << "Result (with lambda):" << endl;</pre>
// functor z^3 componentwise, specify OutputScalar and Derived for lambdas  \frac{\text{displn}(\text{cwise} < \text{cplx, cmat} > (a, [](\text{const cplx\& z}) -> \text{cplx} } ) + cplx 
    return z*z*z;}));
cout << "Result (with proper function):" << endl;</pre>
// automatic type deduction for proper functions
displn(cwise(a, &pow3));
// Gram-Schmidt
cout << endl << "Gram-Schmidt on matrix:" << endl;</pre>
```

```
cmat A(3, 3);
A << 1, 1, 0, 0, 2, 0, 0, 0;
displn(A);
cmat Ags = grams(A);
cout << endl << "Result:" << endl:
displn(Aqs):
cout << endl << "Projector is:" << endl;
displn(Ags * adjoint(Ags));
// spectral decomposition test
cout << endl << "Spectral decomposition tests." << endl;</pre>
std::size_t D = 4;
cmat rH = randH(D);
dmat evalsH = hevals(rH);
cmat evectsH = hevects(rH);
cmat spec = cmat::Zero(D, D);
for (std::size_t i = 0; i < D; i++)
    spec += evalsH(i) * prj((cmat) evectsH.col(i));
cout << "Original matrix: " << endl;</pre>
displn(rH);
cout << endl << "Reconstructed from spectral decomposition: " << endl;</pre>
displn(spec);
cout << "Difference in norm: " << norm(spec - rH) << endl;</pre>
// channel tests
cout << endl << "Channel tests." << endl;</pre>
std::size\_t nk = 10, d = 2; // nk Kraus on d-dimensional system
std::vector<cmat> Ks = randkraus(nk, d);
cmat rho_in = randrho(d); // input state
cmat rho_out = channel(rho_in, Ks); // output state
cout << "Computing its Choi matrix..." << endl;</pre>
cmat choim = choi(Ks);
cout << "Choi matrix:" << endl;</pre>
displn(choim);
cout << endl << "The eigenvalues of the Choi matrix are: " << endl;</pre>
displn(transpose(hevals(choim)));
cout << endl << "Their sum is: " << sum(hevals(choim)) << endl;</pre>
std::vector<cmat> Kperps = choi2kraus(choim);
cout << endl << "The Kraus rank of the channel is: " << Kperps.size()</pre>
         << endl;
cmat rho_out1 = channel(rho_in, Kperps);
cout << end1 << "Difference in norm on output states: "</pre>
         << norm(rho_out1 - rho_out) << endl;
cout << endl << "Superoperator matrix:" << endl;</pre>
cmat smat = super(Ks);
displn(smat);
cout << endl << "The eigenvalues of the superoperator matrix are: " << endl;</pre>
cmat evalsupop = evals(smat);
displn(transpose(evalsupop));
cout << endl << "Their absolute values are: " << endl;
for (std::size_t i = 0; i < (std::size_t) evalsupop.size(); i++)
    cout << std::abs((cplx) evalsupop(i)) << " ";</pre>
cout << endl << endl << "Diference in norm for superoperator action: ";</pre>
cmat rho_out2 = transpose(
         (cmat) reshape(smat * reshape(transpose(rho_in), d * d, 1), d, d));
cout << norm(rho_out - rho_out2) << endl;</pre>
// statistics tests
cout << endl << "Statistics tests." << endl;</pre>
std::vector<cplx> ampl = { 1. + 1_i, 1. - 1_i };
cmat va(1, 4);
va << 0.1, 1, 1. + 1_i, 1. + 2_i;
DiscreteDistributionAbsSquare dc(va);
cout << "The probabilities are: ";
displn(dc.probabilities(), ", ", "{", "}");</pre>
// // TIMING tests
cout << endl << "Timing tests..." << endl;
std::size_t n = 12; // number of qubits
std::size_t N = std::pow(2, n);
vector<std::size_t> dims(n, 2); // local dimensions
cout << "n = " << n << " qubits, matrix size " << N << " x " << N << "."
         << endl;
// matrix initialization
cout << endl << "Matrix initialization timing." << endl;</pre>
// start the timer, automatic tic() in the constructor
Timer t, total;
cmat randcmat = cmat::Random(N, N);
t.toc(); // read the time
cout << "Took " << t << " seconds." << endl;
// lazv matrix product
```

```
cout << endl << "Lazy matrix product timing." << endl;</pre>
auto lazyprod = randcmat * randcmat; // lazyprod has type GenMatProduct
t.toc(); // read the time
cout << "Took " << t << " seconds." << endl;
// ptrace1 timing
cout << endl << "ptrace1 timing." << endl;</pre>
t.tic(); // reset the chronometer
// trace away half of the qubits
ptrace1(randcmat,
{ (std::size_t) std::sqrt(N), (std::size_t) std::sqrt(N) }); t.toc(); // read the time cout << "Took " << t << " seconds." << endl;
// ptrace2 timing
cout << endl << "ptrace2 timing." << endl;</pre>
t.tic(); // reset the chronometer // trace away half of the qubits
ptrace2(randcmat,
{ (std::size_t) std::sqrt(N), (std::size_t) std::sqrt(N) });
t.toc(); // read the time
cout << "Took " << t << " seconds." << endl;
// ptrace
cout << endl << "ptrace timing." << endl;</pre>
vector<std::size_t> subsys_ptrace = { 0 };
cout << "Subsytem(s): ";
displn(subsys_ptrace, ", ");</pre>
t.tic();
ptrace(randcmat, subsys ptrace, dims);
t.toc();
cout << "Took " << t << " seconds." << endl;
// ptranspose
cout << endl << "ptranspose timing." << endl;</pre>
vector<std::size_t> subsys_ptranspose; // partially transpose n-1 subsystems
for (std::size_t i = 0; i < n - 1; i++)</pre>
     subsys_ptranspose.push_back(i);
cout << "Subsytem(s): ";
displn(subsys_ptranspose, ", ");
t.tic();
ptranspose (randcmat, subsys ptranspose, dims);
t.toc();
cout << "Took " << t << " seconds." << endl;
// syspermute
cout << endl << "syspermute timing." << endl;</pre>
fout < end < syspenmete timing. < end;
vector(std::size_t) perm; // left-shift all subsystems by 1
for (std::size_t i = 0; i < n; i++)
    perm.push_back((i + 1) % n);</pre>
cout << "Subsytem(s): ";</pre>
displn(perm, ", ");
t.tic();
syspermute(randcmat, perm, dims);
t.toc();
cout << "Took " << t << " seconds." << endl;
      // matrix product
cout << endl << "Matrix product timing." << endl;</pre>
11
      t.tic(); // reset the chronometer
11
//
      cmat prodmat = randcmat * randcmat; // explicit cmat now
      t.toc(); // read the time
cout << "Took " << t << " seconds." << endl;
// END TIMING
total.toc(); // read the total running time
cout << endl << "Total time: " << total.seconds() << " seconds.";
cout << endl << "Exiting qpp..." << endl;</pre>
```

# Namespace Index

	2.1	<b>Names</b>	pace	List
--	-----	--------------	------	------

Here is	a list of	all	nar	ne	spa	ace	es	wit	:h I	bri	ef	de	esc	rip	otic	ons	s:												
qpp																													13
qpp:	::interna	d.																											68

6 Namespace Index

## **Hierarchical Index**

### 3.1 Class Hierarchy

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

qpp::DiscreteDistribution	 71
qpp::DiscreteDistributionAbsSquare	 72
exception	
qpp::Exception	 73
qpp::NormalDistribution	 82
qpp::Qudit	 83
qpp::Singleton $<$ T $>$	 86
qpp::Gates	 76
qpp::RandomDevices	 85
qpp::Singleton < const Gates >	 86
qpp::Singleton < const States >	 86
qpp::States	 87
qpp::Singleton < RandomDevices >	 86
qpp::Timer	 90
qpp::UniformIntDistribution	 91
app::UniformRealDistribution	 91

8 **Hierarchical Index** 

## **Class Index**

### 4.1 Class List

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

p::DiscreteDistribution	71
p::DiscreteDistributionAbsSquare	72
p::Exception	73
p::Gates	76
p::NormalDistribution	82
p::Qudit	83
p::RandomDevices	
p::Singleton< T >	86
p::States	87
p::Timer	90
p::UniformIntDistribution	91
n::IniformRealDistribution	91

10 Class Index

## File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

include/channels.h
include/constants.h
include/entanglement.h
include/entropies.h
include/functions.h
include/internal.h
include/io.h
include/matlab.h
include/qpp.h
include/random.h
include/types.h
include/classes/exception.h
include/classes/gates.h
include/classes/qudit.h
include/classes/randevs.h
include/classes/singleton.h
include/classes/stat.h
include/classes/states.h
include/classes/timer.h

12 File Index

### **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**

Adjoint.

```
    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)
```

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

    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 evects (const Eigen::MatrixBase< Derived > &A)
     Eigenvectors.

    template<typename Derived >

  dmat hevals (const Eigen::MatrixBase< Derived > &A)
     Hermitian eigenvalues.
• template<typename Derived >
  cmat hevects (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 >

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 >

Partial transpose.

• template<typename Derived1 , typename Derived2 >

 $\frac{\text{DynMat}<\text{typename Derived1::Scalar}>\text{comm}\text{ (const Eigen::MatrixBase}<\text{Derived1}>\text{\&A, const Eigen::}\leftarrow\text{MatrixBase}<\text{Derived2}>\text{\&B)}$ 

Commutator.

• template<typename Derived1 , typename Derived2 >

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)  $\bullet \ \ \mathsf{template} \mathord{<} \mathsf{typename} \ \mathsf{Derived} >$ 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

constexpr double ee = 2.718281828459045235360287471352662497

Base of natural logarithm, e.

RandomDevices & rdevs = RandomDevices::get\_instance()

gpp::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.

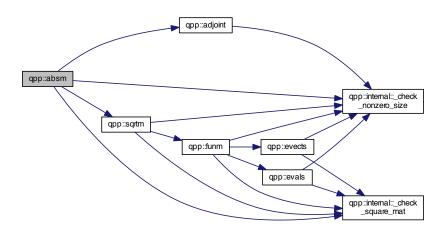
**Parameters** 

A | 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.

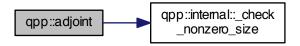
#### **Parameters**

Α	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

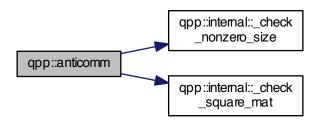
#### **Parameters**

Α	Eigen expression
В	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*.

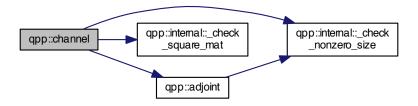
#### **Parameters**

rho	Eigen expression
Ks	std::vector 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*.

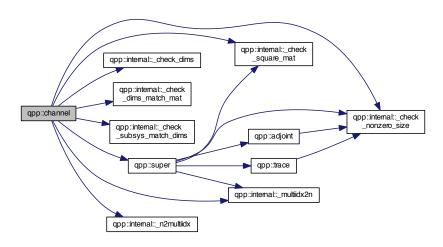
#### **Parameters**

rho	Eigen expression
Ks	std::vector of Eigen expressions representing the set of Kraus operators
subsys	Subsystems' indexes
dims	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  $\mathit{Ks}$  in the standard operator basis  $\{|i\rangle\langle j|\}$  ordered in lexicographical order, i.e.  $|0\rangle\langle 0|,\,|0\rangle\langle 1|$  etc.

#### Note

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

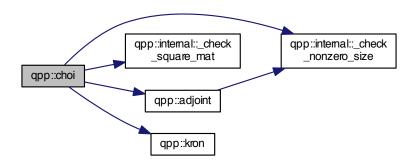
#### **Parameters**

Ks	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

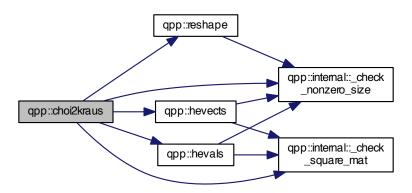
The Kraus operators satisfy  $Tr(K_i^\dagger K_j) = \delta_{ij}$  for all i 
eq j

#### Parameters

A Choi ma	atrix
-----------	-------

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< Derived2 > & B )

# Commutator.

Commutator [A,B] = AB - BA

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

### Parameters

Α	Eigen expression
В	Eigen expression

# Returns

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



Compose permutations.

perm	Permutation
sigma	Permutation

# Returns

Composition of the permutations *perm* o *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.

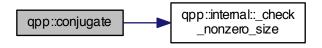
## **Parameters**

A Ligent expression
---------------------

# Returns

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

Here is the call graph for this function:



 $6.1.2.11 \quad template < typename \ Derived > cmat \ qpp::cosm \ ( \ const \ Eigen::MatrixBase < Derived > \& \ \textit{A} \ )$ 

Matrix cos.

Α	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**

Α	Eigen expression
f	Pointer-to-function from scalars of A to OutputScalar

# 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.

Α	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 )



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 )



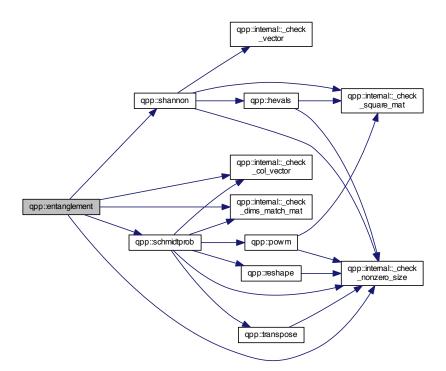
6.1.2.21 void qpp::displn (const cplx c, double chop = chop, std::ostream & os = std::cout)

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 )

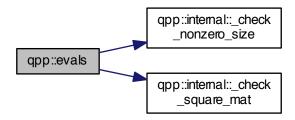
Eigenvalues.

Α	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::evects ( const Eigen::MatrixBase< Derived > & A )

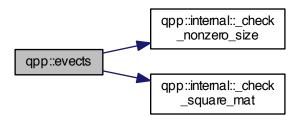
# Eigenvectors.

# **Parameters**

Α	Eigen expression

## Returns

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



# Expand out.

Expand out A as a matrix in a multi-partite system Faster than using <a href="mailto:qpp::kron(I, I, ..., I, A, I, ..., I">qpp::kron(I, I, ..., I, A, I, ..., I)</a>

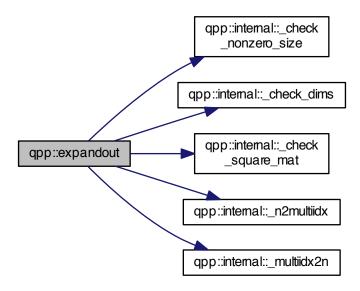
## **Parameters**

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

# Returns

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

Here is the call graph for this function:



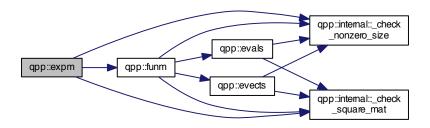
6.1.2.26 template<typename Derived > cmat qpp::expm ( const Eigen::MatrixBase< Derived > & A )

## Matrix exponential.

Α	Eigen expression

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)

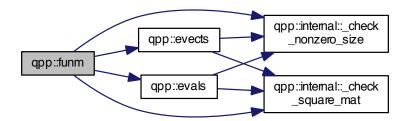
# Functional calculus f(A)

## **Parameters**

Α	Eigen expression
f	Pointer-to-function from complex to complex

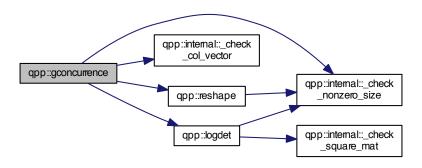
# Returns

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



6.1.2.28 template < typename Derived > double qpp::gconcurrence ( 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)

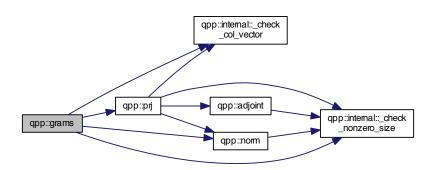
**Parameters** 

Vs	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 )

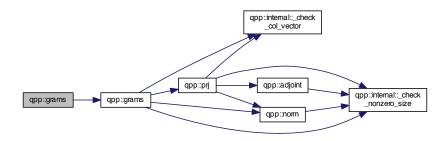
Gram-Schmidt orthogonalization (std::initializer list overload)

V/c	std::initializer_list of Eigen expressions as column vectors
vo	Starillitializat ilot of Fiach 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:



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

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

# **Parameters**

Α	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 )

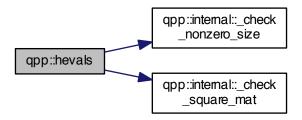
Hermitian eigenvalues.

Λ	Figure overcooien
A	Elderi expression
	9 1

# 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 )

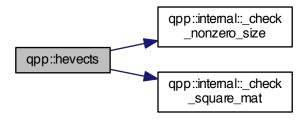
Hermitian eigenvectors.

# **Parameters**

Α	Eigen expression

## Returns

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



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

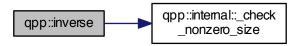
Inverse.

Α	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**

perm	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 <a href="mailto:qpp::kron()">qpp::kron()</a>

head	Eigen expression

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)

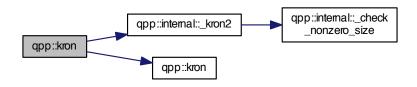
## **Parameters**

head	Eigen expression
tail	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)

	As	std::vector of Eigen expressions
--	----	----------------------------------

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,, Ak}
AS	stdinitializer_list of Ligeri expressions, such as {A1, A2, ,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:



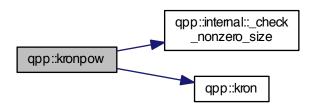
6.1.2.40 template<typename Derived > DynMat<typename Derived::Scalar> qpp::kronpow ( const Eigen::MatrixBase < Derived > & A, std::size\_t n)

Kronecker power.

Α	Eigen expression
n	Non-negative integer

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 )

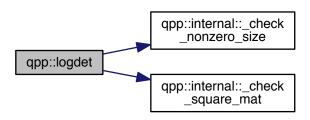
Logarithm of the determinant.

Especially useful when the determinant overflows/underflows

Α	Eigen expression

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.

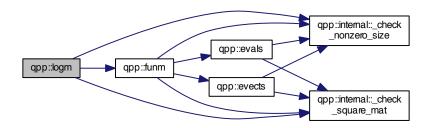
### **Parameters**

Α	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.

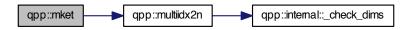
Constructs the multi-partite qubit ket  $|mask\rangle$ , where mask is a std::vector of 0's and 1's

mask	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  $|mask\rangle$ , 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**

mask	std::vector of non-negative integers
dims	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)

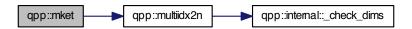
Constructs the multi-partite qudit ket  $|mask\rangle$  in a multi-partite system, all subsystem having equal dimension d mask is a std::vector of non-negative integers, and each element in mask has to be strictly smaller than d

mask	std::vector of non-negative integers
d	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**

midx	Multi-index
dims	Dimensions of the multi-partite system

# Returns

Non-negative integer index

Here is the call graph for this function:



 $6.1.2.51 \quad std::vector < std::size\_t > qpp::n2multiidx ( \ std::size\_t \ \textit{n, } const \ std::vector < std::size\_t > \& \ \textit{dims } )$ 

Non-negative integer index to multi-index.

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

n	Non-negative integer index
dims	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**

Α	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.

D	Non-negative integer

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

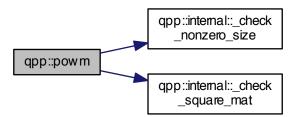
By convention  $A^0 = I$ 

**Parameters** 

Α	Eigen expression
n	Non-negative integer

## Returns

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



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

# Projector.

Normalized projector onto state vector

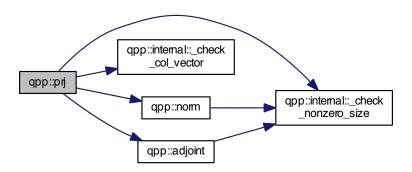
### **Parameters**

V	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 )

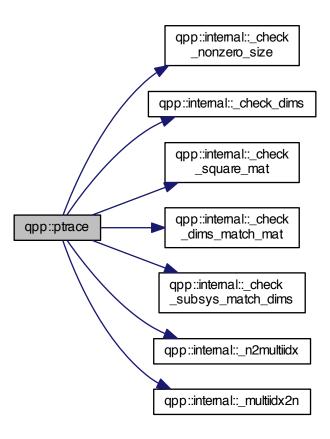
# Partial trace.

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

Α	Eigen expression
subsys	Subsystems' indexes
dims	Dimensions of the multi-partite system

Partial trace  $Tr_{subsys}(\cdot)$  over the subsytems *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

Α	Eigen expression
dims	Dimensions of bi-partite system (must be a std::vector with 2 elements)

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

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**

Α	Eigen expression
dims	Dimensions of bi-partite system (must be a std::vector 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:



# Partial transpose.

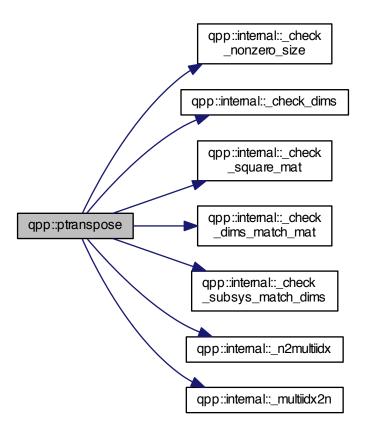
Partial transpose of the multi-partite density matrix over a list of subsystems

### **Parameters**

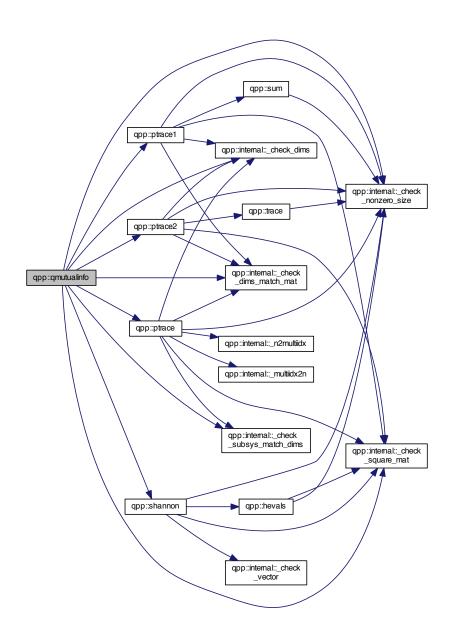
Α	Eigen expression
subsys	Subsystems' indexes
dims	Dimensions of the multi-partite system

# Returns

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



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 )



- 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:

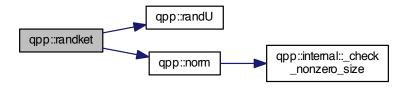


6.1.2.68 long long qpp::randint ( long long a, long long b )



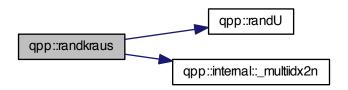
# 6.1.2.69 ket qpp::randket ( std::size\_t D )

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 )



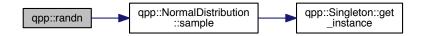
6.1.2.73 template<> cmat qpp::randn ( std::size\_t rows, std::size\_t cols, double mean, double sigma )

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:

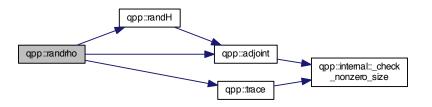


6.1.2.75 std::vector<std::size\_t> qpp::randperm ( std::size\_t n )



# 6.1.2.76 cmat qpp::randrho ( std::size\_t D )

Here is the call graph for this function:



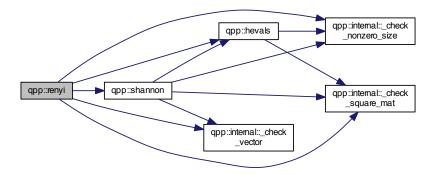
# 6.1.2.77 cmat qpp::randU ( std::size\_t D )

# 6.1.2.78 cmat qpp::randV ( std::size\_t Din, std::size\_t Dout )

Here is the call graph for this function:

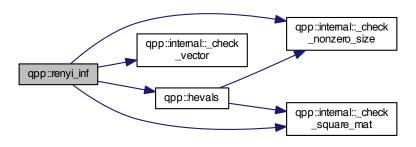


# 6.1.2.79 template<typename Derived > double qpp::renyi ( const double alpha, const Eigen::MatrixBase< Derived > & A )



6.1.2.80 template < typename Derived > double qpp::renyi\_inf ( const Eigen::MatrixBase < Derived > & A )

Here is the call graph for this function:



## Reshape.

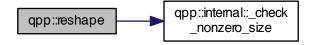
Uses column-major order when reshaping (same as MATLAB)

### **Parameters**

Α	Eigen expression
rows	Number of rows of the reshaped matrix
cols	Number of columns of the reshaped matrix

## Returns

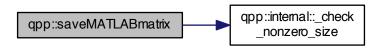
Reshaped matrix with rows rows and cols columns, as a dynamic matrix over the same scalar field



- 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 & mode )

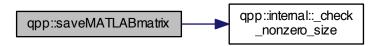
6.1.2.84 template<> void qpp::saveMATLABmatrix ( const Eigen::MatrixBase< dmat > & A, const std::string & mat\_file, const std::string & var\_name, const std::string & mode )

Here is the call graph for this function:

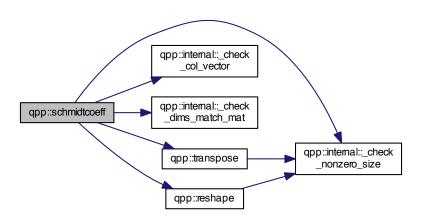


6.1.2.85 template<> void qpp::saveMATLABmatrix ( const Eigen::MatrixBase< cmat > & A, const std::string & mat\_file, const std::string & var\_name, const std::string & mode )

Here is the call graph for this function:

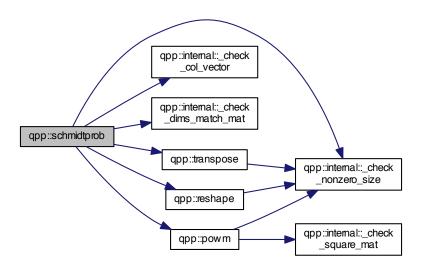


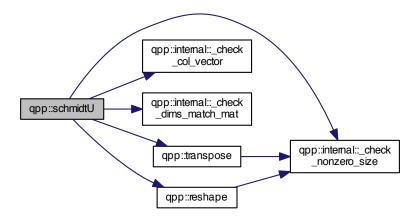
6.1.2.86 template < typename Derived > cmat qpp::schmidtcoeff ( const Eigen::MatrixBase < Derived > & A, const std::vector < std::size\_t > & dims )



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

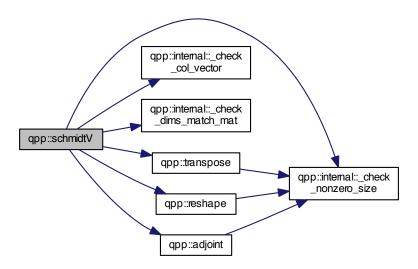
Here is the call graph for this function:





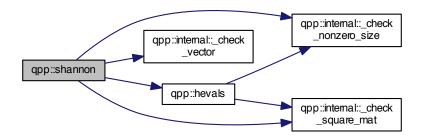
6.1.2.89 template < typename Derived > cmat qpp::schmidtV ( const Eigen::MatrixBase < Derived > & A, const std::vector < std::size\_t > & dims )

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.

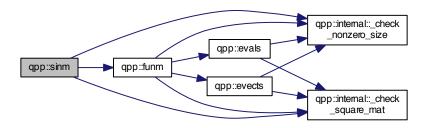
#### **Parameters**

Α	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  $\emph{A}$  to compute the matrix power By convention  $\emph{A}^0 = \emph{I}$ 

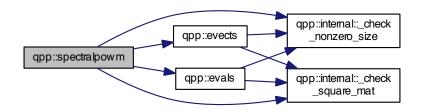
## **Parameters**

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.

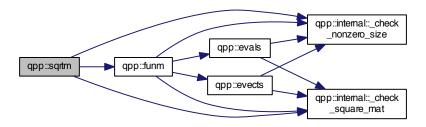
#### **Parameters**

Α	l 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 Ks in the standard operator basis  $\{|i\rangle\langle j|\}$  ordered in lexicographical order, i.e.  $|0\rangle\langle 0|$ ,  $|0\rangle\langle 1|$  etc.

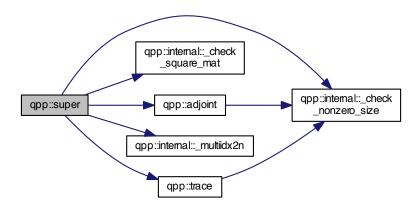
#### **Parameters**

Ks	std::vector 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:



System permutation.

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

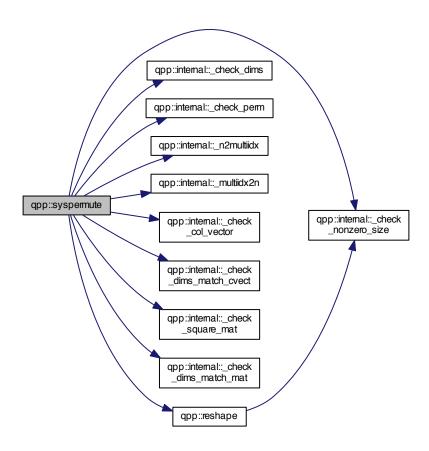
## Parameters

Α	Eigen expression
perm	Permutation
dims	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** 

Α	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

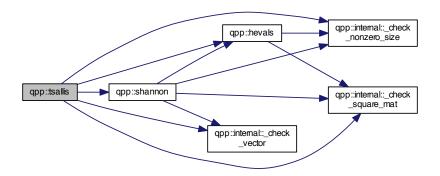
Transpose of A, as a dynamic matrix over the same scalar field

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</pre>
```

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 <u>\_check\_square\_mat</u> (const Eigen::MatrixBase< Derived > &A)

template<typename Derived >

bool <u>\_check\_vector</u> (const Eigen::MatrixBase< Derived > &A)

template<typename Derived >

bool <u>\_check\_row\_vector</u> (const Eigen::MatrixBase< Derived > &A)

• template<typename Derived >

bool <u>\_check\_col\_vector</u> (const Eigen::MatrixBase< Derived > &A)

template<typename T >

bool <u>\_check\_nonzero\_size</u> (const T &x)

- bool <u>\_check\_dims</u> (const std::vector < std::size\_t > &dims)
- template<typename Derived >

 $\label{local_bool_check_dims_match_mat} \mbox{ (const std::vector} < \mbox{ std::size\_t} > \mbox{\&dims, const Eigen::MatrixBase} < \mbox{ Derived} > \mbox{\&A})$ 

template<typename Derived >

bool <u>\_check\_dims\_match\_cvect</u> (const std::vector< std::size\_t > &dims, const Eigen::MatrixBase< Derived > &V)

• template<typename Derived >

 $\label{local_check_dims_match_rvect} bool\_check\_dims\_match\_rvect \ (const \ std::vector < std::size\_t > \&dims, \ const \ Eigen::MatrixBase < Derived > \&V)$ 

- 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 )
- $6.2.2.12 \quad template < typename \ Derived > bool \ qpp::internal::\_check\_vector \ ( \ const \ Eigen::MatrixBase < Derived > \& \ A \ )$
- $\begin{array}{ll} \textbf{6.2.2.13} & \textbf{template} < \textbf{typename Derived1} \ , \ \textbf{typename Derived2} > \textbf{DynMat} < \textbf{typename Derived1::Scalar} > \textbf{qpp::internal::\_kron2} \ ( \\ \textbf{const Eigen::MatrixBase} < \textbf{Derived1} > \& \textit{A}, \ \textbf{const Eigen::MatrixBase} < \textbf{Derived2} > \& \textit{B} \ ) \end{array}$

Here is the call graph for this function:



- 6.2.2.14 std::size\_t app::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)
- Discrete Distribution (std::initializer\_list< double > weights)
- Discrete Distribution (std::vector< double > weights)
- std::size\_t sample ()
- std::vector< double > probabilities () const

#### **Protected Attributes**

```
std::discrete_distributionstd::size_t > _d
```

### 7.1.1 Constructor & Destructor Documentation

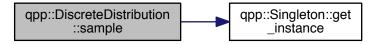
- 7.1.1.1 template < typename InputIterator > qpp::DiscreteDistribution::DiscreteDistribution ( InputIterator first, InputIterator last ) [inline]
- 7.1.1.2 qpp::DiscreteDistribution::DiscreteDistribution ( std::initializer\_list< double > weights ) [inline]
- 7.1.1.3 qpp::DiscreteDistribution::DiscreteDistribution ( 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\_distributionstd::size\_t > \_d

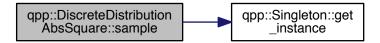
## 7.2.1 Constructor & Destructor Documentation

- 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]

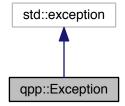
The documentation for this class was generated from the following file:

• include/classes/stat.h

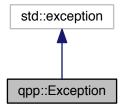
## 7.3 qpp::Exception Class Reference

#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\_C↔ VECTOR, Type::MATRIX\_NOT\_SQUARE\_OR\_RVECTOR,

Type::MATRIX\_NOT\_SQUARE\_OR\_VECTOR, Type::DIMS\_INVALID, Type::DIMS\_NOT\_EQUAL, Type::D↔ IMS\_MISMATCH\_MATRIX,

 $\label{type::DIMS_MISMATCH_CVECTOR} Type::DIMS\_MISMATCH\_RVECTOR, Type::DIMS\_MISMATCH\_VE \leftarrow CTOR, Type::SUBSYS\_MISMATCH\_DIMS,$ 

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\_EXCEPT → ION }

### **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 noexceptoverride

#### **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 dimenisons' 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 dimenisons' 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 dimenisons' 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 dimenisons' 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 duplicatates, 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]

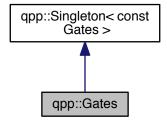
The documentation for this class was generated from the following file:

• include/classes/exception.h

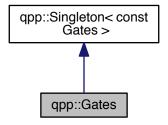
## 7.4 qpp::Gates Class Reference

#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 Id (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 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 CX { 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 >

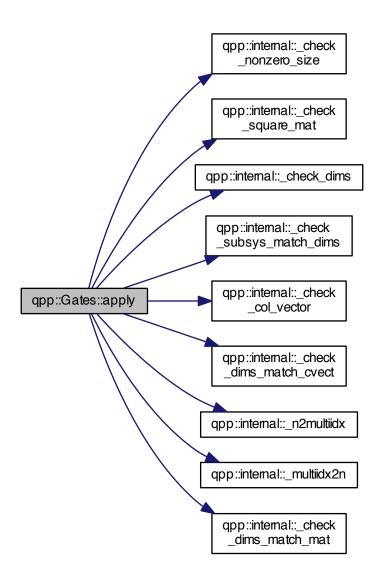
## **Additional Inherited Members**

## 7.4.1 Constructor & Destructor Documentation

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

## 7.4.2 Member Function Documentation

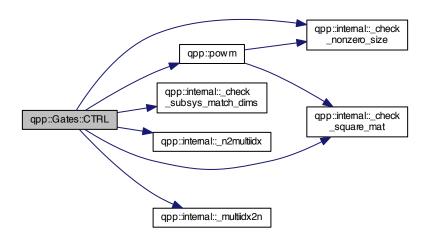
Here is the call graph for this function:



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 n, 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]

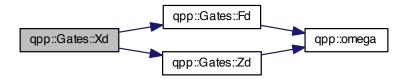
Here is the call graph for this function:



- 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::ld2 { cmat::ldentity(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

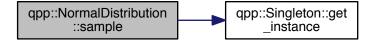
## 7.5.1 Constructor & Destructor Documentation

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

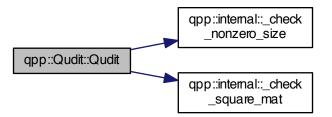
## **Private Attributes**

- 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:

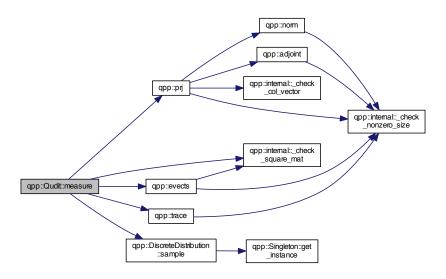


## 7.6.2 Member Function Documentation

- 7.6.2.1 std::size\_t qpp::Qudit::getD() const [inline]
- 7.6.2.2 cmat qpp::Qudit::getRho() const [inline]

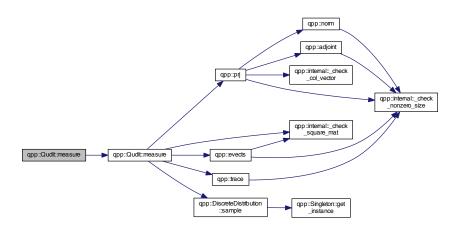
7.6.2.3 std::size\_t qpp::Qudit::measure ( const cmat & U, bool destructive = false ) [inline]

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]

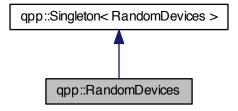
The documentation for this class was generated from the following file:

• include/classes/qudit.h

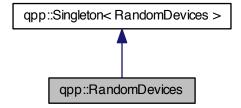
# 7.7 qpp::RandomDevices Class Reference

#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**

 $\bullet \ \ {\it class Singleton}{<} \ {\it RandomDevices} >$ 

## **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 < Random Devices > [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

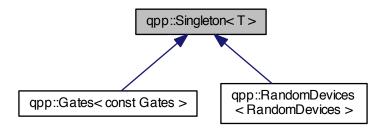
The documentation for this class was generated from the following file:

• 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>:: $\sim$ Singleton ( ) [inline], [protected], [virtual]

## 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]

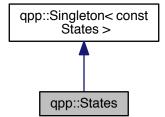
The documentation for this class was generated from the following file:

• include/classes/singleton.h

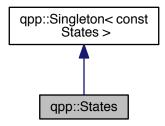
# 7.9 qpp::States Class Reference

#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**

```
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]
        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) }
        ket qpp::States::b10 { ket::Zero(4) }
7.9.3.3
        ket qpp::States::b11 { ket::Zero(4) }
        ket qpp::States::GHZ { ket::Zero(8) }
        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)</li>

#### 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.
- $\bullet \ \ {\it template}{<} {\it typename Derived}>$ 
  - 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.

 $\bullet \ \ \mathsf{template} \mathord{<} \mathsf{typename} \ \mathsf{Derived} >$ 

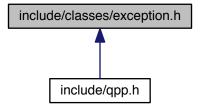
cmat qpp::channel (const Eigen::MatrixBase< Derived > &rho, const std::vector< cmat > &Ks, const std $\leftrightarrow$  ::vector< std::size\_t > &subsys, const std::vector< std::size\_t > &dims)

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

94 File Documentation

# 8.2 include/classes/exception.h File Reference

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



### Classes

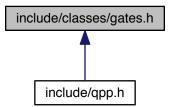
· class qpp::Exception

## **Namespaces**

• qpp

# 8.3 include/classes/gates.h File Reference

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



# Classes

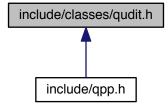
class qpp::Gates

## **Namespaces**

qpp

# 8.4 include/classes/qudit.h File Reference

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



### Classes

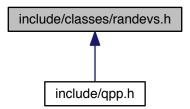
· class qpp::Qudit

## **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

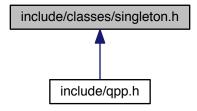
## **Namespaces**

qpp

96 File Documentation

# 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:

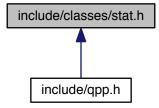
```
class Foo: public Singleton<const Foo>\
{\
          friend class Singleton<const Foo>;
```

## 8.6.1.2 #define CLASS\_SINGLETON( Foo )

#### Value:

# 8.7 include/classes/stat.h File Reference

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



# **Classes**

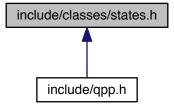
- class qpp::NormalDistribution
- class qpp::UniformRealDistribution
- class qpp::UniformIntDistribution
- class qpp::DiscreteDistribution
- class qpp::DiscreteDistributionAbsSquare

# **Namespaces**

• qpp

# 8.8 include/classes/states.h File Reference

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



## **Classes**

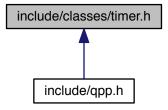
• class qpp::States

# **Namespaces**

• qpp

# 8.9 include/classes/timer.h File Reference

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



#### Classes

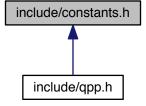
class qpp::Timer

# **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

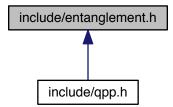
 $\pi$ 

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**

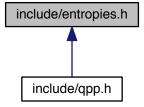
qpp

- 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:



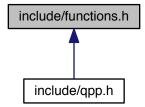
#### **Namespaces**

• qpp

- 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 <a href="mailto:qpp::det">qpp::det</a> (const Eigen::MatrixBase</a> 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 <a href="mailto:qpp::norm">qpp::norm</a> (const Eigen::MatrixBase</a> 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 <a href="mailto:qpp::hevects">qpp::hevects</a> (const Eigen::MatrixBase</a> 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 <a href="mailto:qpp::logm">qpp::logm</a> (const Eigen::MatrixBase</a> 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 >

 $\label{lem:def:def:DynMat} \mbox{DynMat} < \mbox{typename Derived::Scalar} > \mbox{qpp::ptrace1} \mbox{ (const Eigen::MatrixBase} < \mbox{Derived} > \&\mbox{A, const std} \\ \mbox{::vector} < \mbox{std::size\_t} > \&\mbox{dims})$ 

Partial trace.

template<typename Derived >

 $\label{lem:def:DynMat} \mbox{DynMat} < \mbox{typename Derived::Scalar} > \mbox{qpp::ptrace2} \mbox{ (const Eigen::MatrixBase} < \mbox{Derived} > \&\mbox{A, const std} \\ \mbox{::vector} < \mbox{std::size\_t} > \&\mbox{dims})$ 

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 >

 $\label{lem:def:def:DynMat} \mbox{ Derived1::Scalar } > \mbox{qpp::comm (const Eigen::MatrixBase} < \mbox{ Derived1 } > \mbox{\&A, const Eigen::MatrixBase} < \mbox{ Derived2 } > \mbox{\&B)}$ 

Commutator.

template<typename Derived1 , typename Derived2 >

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

Anti-commutator.

 $\bullet \ \ {\sf template}{<} {\sf typename \ Derived} >$ 

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)

 $\bullet \ \ \text{template}{<} \text{typename Derived} >$ 

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)

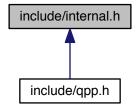
Multi-partite qudit ket (same dimensions overload)

- std::vector< std::size\_t > qpp::invperm (const std::vector< std::size\_t > &perm)
   Inverse permutation.
- std::vector< std::size\_t > app::compperm (const std::vector< std::size\_t > aperm, const std::vector< std
   ::size\_t > aperm, const std::vector< std
   ::size\_t

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)

 $\bullet \ \ \text{template}{<} \text{typename Derived} >$ 

bool qpp::internal::\_check\_vector (const Eigen::MatrixBase< Derived > &A)

• template<typename Derived >

bool qpp::internal:: check row vector (const Eigen::MatrixBase< Derived > &A)

 $\bullet \ \ \mathsf{template}{<} \mathsf{typename} \ \mathsf{Derived} >$ 

bool qpp::internal::\_check\_col\_vector (const Eigen::MatrixBase< Derived > &A)

• template<typename T >

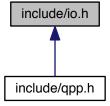
bool qpp::internal::\_check\_nonzero\_size (const T &x)

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::Matrix
   Base< Derived > &A)
- template<typename Derived >
   bool qpp::internal::\_check\_dims\_match\_cvect (const std::vector< std::size\_t > &dims, const Eigen::Matrix
   Base< Derived > &V)
- template<typename Derived >
   bool qpp::internal::\_check\_dims\_match\_rvect (const std::vector< std::size\_t > &dims, const Eigen::Matrix
   Base< 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**

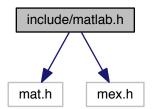
qpp

- 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::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 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

- 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 &mat\_file, 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**

dbb

#### **Variables**

```
    RandomDevices & qpp::rdevs = RandomDevices::get_instance()
    qpp::RandomDevices Singleton
```

const Gates & qpp::gt = Gates::get\_instance()

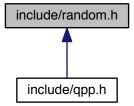
qpp::Gates const Singleton

• 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)
- $\bullet \ \ \mathsf{template} \mathord{<} \mathsf{typename} \ \mathsf{Derived} >$

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/>

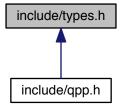
cmat qpp::randn (std::size\_t rows, std::size\_t cols, double mean, double sigma)

- double qpp::randn (double mean=0, double sigma=1)
- cmat qpp::randU (std::size\_t D)
- cmat qpp::randV (std::size\_t Din, std::size\_t Dout)
- std::vector< cmat > qpp::randkraus (std::size\_t n, std::size\_t D)
- cmat 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.

 $\bullet \ \ \text{template}{<} \text{typename Scalar} >$ 

 $using \ qpp::DynMat = Eigen::Matrix < Scalar, Eigen::Dynamic, Eigen::Dynamic > \\$ 

Dynamic Eigen matrix over the field specified by Scalar.

# Index

abass	
absm	qpp, 29
qpp, 19	displn
adjoint	qpp, 29, 30
qpp, 20	dmat
anticomm	qpp, 19
qpp, 20	ee
bra	qpp, 67
gpp, 19	entanglement
<b>Ч</b> РР, 10	qpp, 31
CUSTOM EXCEPTION	eps
qpp::Exception, 75	qpp, 67
channel	evals
qpp, 21	qpp, 31
choi	evects
qpp, 23	qpp, 32
choi2kraus	expandout
qpp, 24	qpp, 32
chop	expm
qpp, 67	qpp, 33
cmat	
qpp, 19	funm
comm	qpp, <mark>34</mark>
qpp, 25	
compperm	gconcurrence
qpp, 25	qpp, 34
conjugate	grams
qpp, 27	qpp, 35, 36
cosm	gt
qpp, 27	qpp, 67
cplx	hevals
qpp, 19	qpp, 36
cwise	hevects
qpp, 28	qpp, 37
DIME INIVALID	
DIMS_INVALID qpp::Exception, 75	inverse
DIMS_MISMATCH_CVECTOR	qpp, 37
qpp::Exception, 75	invperm
DIMS MISMATCH MATRIX	qpp, 39
qpp::Exception, 75	lead
DIMS MISMATCH RVECTOR	ket
qpp::Exception, 75	qpp, 19 kron
DIMS MISMATCH VECTOR	qpp, 39–41
qpp::Exception, 75	
DIMS_NOT_EQUAL	kronpow qpp, 41
qpp::Exception, 75	4pp, -1
det	load
qpp, 28	qpp, 42
disp	logdet
•	<u> </u>

INDEX 111

40	50
qpp, 42 logm	qpp, 52 qpp, 13
qpp, 43	absm, 19
۹۳۴, ۱۵	adjoint, 20
MATRIX_NOT_CVECTOR	anticomm, 20
qpp::Exception, 75	bra, 19
MATRIX_NOT_RVECTOR	channel, 21
qpp::Exception, 75	choi, 23
MATRIX_NOT_SQUARE	choi2kraus, 24
qpp::Exception, 75 MATRIX_NOT_SQUARE_OR_CVECTOR	chop, 67
qpp::Exception, 75	cmat, 19
MATRIX_NOT_SQUARE_OR_RVECTOR	comm, 25 compperm, 25
qpp::Exception, 75	conjugate, 27
MATRIX_NOT_SQUARE_OR_VECTOR	cosm, 27
qpp::Exception, 75	cplx, 19
MATRIX_NOT_VECTOR	cwise, 28
qpp::Exception, 75	det, 28
maxn	disp, 29
qpp, 67 mket	displn, 29, 30
qpp, 43, 44	dmat, 19
multiidx2n	ee, 67
qpp, 45	entanglement, 31 eps, 67
	evals, 31
n2multiidx	evects, 32
qpp, 45	expandout, 32
NOT_BIPARTITE	expm, 33
qpp::Exception, 75 NOT QUBIT GATE	funm, 34
qpp::Exception, 75	gconcurrence, 34
NOT QUBIT SUBSYS	grams, 35, 36
qpp::Exception, 75	gt, 67
norm	hevals, 36 hevects, 37
qpp, 46	inverse, 37
OUT OF PANCE	invperm, 39
OUT_OF_RANGE qpp::Exception, 75	ket, 19
omega	kron, 39-41
qpp, 46	kronpow, 41
TIPE : 1	load, 42
PERM_INVALID	logdet, 42
qpp::Exception, 75	logm, 43
pi 0.7	maxn, 67 mket, 43, 44
qpp, 67	multiidx2n, 45
powm qpp, 47	n2multiidx, 45
prj	norm, 46
qpp, 47	omega, 46
ptrace	pi, <mark>67</mark>
qpp, 48	powm, 47
ptrace1	prj, 47
qpp, 49	ptrace, 48 ptrace1, 49
ptrace2 qpp, 50	ptrace2, 50
ptranspose	ptranspose, 51
qpp, 51	qmutualinfo, 52
· ·	rand, 53, 54
qmutualinfo	randint, 54

112 INDEX

randket, 55	qpp, 56
randkraus, 55	randrho
randn, 55, 56	qpp, 56
randperm, 56	rdevs
randrho, 56	qpp, 67
rdevs, 67	renyi
renyi, 57	qpp, 57
reshape, 58	reshape
save, 58	qpp, 58
schmidtcoeff, 59	
schmidtprob, 60	SUBSYS_MISMATCH_DIMS
shannon, 61	qpp::Exception, 75
sinm, 61	save
spectralpowm, 62	qpp, 58
sqrtm, 62	schmidtcoeff
st, 68	qpp, 59
sum, 63	schmidtprob
super, 63	qpp, 60
syspermute, 64	shannon
trace, 65	qpp, 61
transpose, 66	sinm
tsallis, 66	qpp, 61
qpp::Exception	spectralpowm
CUSTOM EXCEPTION, 75	qpp, 62
DIMS INVALID, 75	sqrtm
DIMS MISMATCH CVECTOR, 75	qpp, 62
DIMS MISMATCH MATRIX, 75	st
DIMS MISMATCH RVECTOR, 75	qpp, 68
DIMS MISMATCH VECTOR, 75	sum
	qpp, 63
DIMS_NOT_EQUAL, 75	super
MATRIX_NOT_CVECTOR, 75	qpp, 63
MATRIX_NOT_RVECTOR, 75	syspermute
MATRIX_NOT_SQUARE, 75	qpp, 64
MATRIX_NOT_SQUARE_OR_CVECTOR, 75	чрр, от
MATRIX_NOT_SQUARE_OR_RVECTOR, 75	TYPE MISMATCH
MATRIX_NOT_SQUARE_OR_VECTOR, 75	qpp::Exception, 75
MATRIX_NOT_VECTOR, 75	trace
NOT_BIPARTITE, 75	qpp, 65
NOT_QUBIT_GATE, 75	transpose
NOT_QUBIT_SUBSYS, 75	qpp, 66
OUT_OF_RANGE, 75	tsallis
PERM_INVALID, 75	qpp, 66
SUBSYS_MISMATCH_DIMS, 75	<b>Ч</b> рр, <b>υ</b> υ
TYPE_MISMATCH, 75	UNDEFINED TYPE
UNDEFINED_TYPE, 75	qpp::Exception, 75
UNKNOWN_EXCEPTION, 75	UNKNOWN EXCEPTION
ZERO_SIZE, 75	qpp::Exception, 75
rand	-11-11 , -
qpp, 53, 54	ZERO_SIZE
randint	qpp::Exception, 75
qpp, 54	
randket	
qpp, 55	
randkraus	
qpp, 55	
randn	
qpp, 55, 56	
randperm	