DampingChannel>DampingChannel.apply_single (Calls: 80746, Time: 3496.084 s)

Generated 24-Feb-2021 16:49:27 using performance time.
Class method in file /Users/Arvid/Documents/MATLAB/QECSim/Operations/QuantumOperations/DampingChannel.m
Copy to new window for comparing multiple runs

Parents (calling functions)

Function Name	Function Type	Calls
DampingChannel>DampingChannel.apply	Class method	80746

Lines that take the most time

Line Number	Code	Calls	Total Time (s)	% Time	Time Plot
75	`res = res + op*r_tmp*op';`	161492	1962.565	56.1%	
79	`res = sparse(res);`	80746	826.386	23.6%	
72	`res = zeros(size(r_tmp));`	80746	364.217	10.4%	
78	`if nnz(res) <= size(res,1)^2/2`	80746	240.337	6.9%	
74	`op = obj.nbit_op_element(i, target, tot_bits);`	161492	89.334	2.6%	
All other lines			13.247	0.4%	
Totals			3496.084	100%	

Children (called functions)

Function Name	Function Type	Calls	Total Time (s)	% Time	Time Plot
DampingChannel>DampingChannel.nbit_op_element	Class method	161492	87.435	2.5%	
PhaseDamping>PhaseDamping.get.operation_elements	Class method	80746	6.649	0.2%	
NbitState>NbitState.NbitState	Class method	5824	0.688	0.0%	
NbitState>NbitState.copy_params	Class method	5824	0.314	0.0%	
NbitState>NbitState.get.nbits	Class method	5824	0.059	0.0%	
Self time (built-ins, overhead, etc.)			3400.940	97.3%	
Totals			3496.084	100%	

Code Analyzer results

No Code Analyzer messages.

Coverage results

Show coverage for parent folder

Total lines in function	31
Non-code lines (comments, blank lines)	4
Code lines (lines that can run)	27
Code lines that did run	26
Code lines that did not run	1
Coverage (did run/can run)	96.30 %

Function listing

Time	Calls	Line	
		58	`function rho = apply_single(obj, nstate, target)`
0.023	80746	59	`    if ~(isa(nstate,'NbitState')|| ismatrix(nstate))`
		60	`        error('nstate must be an NbitState, a subclass thereof or a matrix')`
0.021	80746	61	`    end`
0.011	80746	62	`    return_state = 0;`
0.023	80746	63	`    if isa(nstate,'NbitState')`
0.043	5824	64	`        r_tmp = nstate.rho;`
0.360	5824	65	`        tot_bits = nstate.nbits;`
0.001	5824	66	`        return_state = 1;`
0.022	74922	67	`    else`

0.010	74922	68	r_tmp = nstate;
0.061	74922	69	tot_bits = log2(size(nstate,1));
0.010	80746	70	end
		71	
364.217	80746	72	res = zeros(size(r_tmp));
8.013	80746	73	for i =1:size(obj.operation_elements,3)
89.334	161492	74	op = obj.nbit_op_element(i, target, tot_bits);
1962.565	161492	75	res = res + op*r_tmp*op';
0.159	161492	76	end
		77	
240.337	80746	78	if nnz(res) <= size(res,1)^2/2
826.386	80746	79	res = sparse(res);
0.024	80746	80	end
		81	
0.090	80746	82	if return_state
0.841	5824	83	rho = NbitState(res);
0.337	5824	84	rho.copy_params(nstate);
0.012	74922	85	else
0.016	74922	86	rho = res;
0.014	80746	87	end
2.061	80746	88	end

Local functions in this file are not included in this listing.