Links are disabled because this is a static copy of a profile report

spontaneousORIGINAL (Calls: 1, Time: 142.298 s)
Generated 15-Jul-2024 15:16:58 using performance time.
script in file D:\Aalto\2324\BScThesis\OLD\spontaneousORIGINAL.m
Copy to new window for comparing multiple runs

Function listing

```
 time    Calls   line
                  28 close all
                  29 profile on
                  30
< 0.001      1    31 N = 1500;
< 0.001      1    32 Nr = 25;
< 0.001      1    33 w = 1;  %The qubit frequency (taken to be normalized to 1)
< 0.001      1    34 include_mutual = 1;
< 0.001      1    35 gamma = w/(5*sqrt(2));
                  36
< 0.001      1    37 te_results = zeros(1, N); %The array for collecting the results of long time evolution
< 0.001      1    38 gge_results = zeros(1, N+1); %The array for collecting the results of the GGE prediction
                  39 %%
< 0.001      1    40 dia1 = sort(2*w*rand(N,1));
< 0.001      1    41 for idx = 1:Nr
                  42
                  43 %{
                  44 %In this section we first generate the bath Hamiltonian H1
                  45 according to the rule Hij = [-gamma/sqrt(N),gamma/sqrt(N)] and Hii =
                  46 [0,2*Omega]. Then we expand it to include the initially exited site to form
                  47 the total Hamiltonian H.
                  48 %}
                  49
  1.172     25    50 a = -(gamma/sqrt(N)) + 2*(gamma/sqrt(N))*rand(N);
  0.381     25    51 a = include_mutual*triu(a);
  0.437     25    52 H1 = a+a';
  0.595     25    53 H1 = H1 - diag(diag(H1)) + diag(dia1);
                  54
                  55 %Diagonalize the bath Hamiltonian
 25.479     25    56 [vek1, ek1] = eig(H1);
                  57
                  58 %vek1 = [vek1; zeros(N+3,N)];
                  59 %vek2 = [zeros(N+3,N); vek2];
                  60 %vek = [vek1 vek2]
                  61
                  62 %Generate the couplings from the baths to the resonators
  0.004     25    63 lambda1 = -(gamma/sqrt(N)) + 2*(gamma/sqrt(N))*rand(N,1);
                  64
                  65 %Build the total Hamiltonian
  0.309     25    66 H = blkdiag(H1, w);
  0.001     25    67 H(1:N,N+1) = lambda1;
  0.003     25    68 H(N+1,1:N) = lambda1';
                  69 %%
                  70 %{
                  71 In this section, we diagonalize the total Hamiltonian H and set the intial state.
                  72 %}
                  73
 25.155     25    74 [vel, el] = eig(H); %Diagonalize the total Hamiltonian to eigenvectors vel and eigevalues el
  0.575     25    75 plot(diag(el), 'o')
  0.002     25    76 xi02 = zeros(N+1);
< 0.001     25    77 xi02(N+1, N+1) = 1;
  0.057     25    78 xi0 = xi02; %The initial state of the system
                  79
                  80 %%
                  81 %{
                  82 In this section, we time-evolve the intial density matrix and calculate the resulting populations.
                  83 %}
< 0.001     25    84 tmax = 8000000000; %The final time at which the populations are calculated
  0.005     25    85 t = linspace(0,tmax,2); %Vector for times for which to calculate the time evolution
                  86 %E1 =  zeros(1, N);
                  87
< 0.001     25    88 for i = 1:length(t)
  1.578     50    89     Ul = expm(1i*t(i)*el); %Time-evolution operator exp(iHt) in the eigenbasis of H (easy to calculate)
 36.118     50    90     Uj = vel*Ul*(vel'); %Convert it to the basis of the sites
 46.059     50    91     xi = (Uj')*xi0*Uj; %\rho(t) = exp(-iHt)\rho(t=0)exp(iHt)
  0.002     50    92 end
                  93
                  94 %{
                  95 for j = 1:N
                  96      Opj = zeros(N+1);
                  97      Opj(j,j) = 1; %Defining the number operator Opj of the j:th site
                  98      E1(j) = trace(Opj*xi); %Calculating the expectation value of it
                  99 end
```

```
                  %}
  0.005     25    e1 = diag(xi);
  0.001     25    E1 = e1(1:N);

  0.106     25    el = diag(el);
                  %%
                  %{
                  In this section, we calculate the numerical GGE prediction for the populations, which is to compared
                  the long-time evolution. Basically, we just implement a convolution
                  formula.
                  %}
  0.002     25    nau = zeros(1, N+1);
  0.003     25    ujt = abs(vel(N+1,:)).^2;

< 0.001     25    for k = 1:(N+1)
  1.171  37525        uki = abs(vel(k,:)).^2;
  0.052  37525        nau(k) = dot(ujt, uki);
  0.008  37525    end

  0.053     25    te_results = te_results+E1;
< 0.001     25    gge_results = gge_results+nau;


  0.001     25    end
                  %%
                  %{
                  Finally, we average over the set number of iterations.
                  %}
  0.006      1    te_result = te_results/Nr;
< 0.001      1    gge_result = gge_results/Nr;
< 0.001      1    epsilon = sort(dia1);
                  %%
                  %{
                  In this section, we plot the results of both the numerical long-time
                  evolution and the GGE prediction and compare an analytical formula.
                  %}

  0.005      1    omega = linspace(0,2*w,1000000);
< 0.001      1    gavg = (gamma^2)/(3*N);
< 0.001      1    Omega = w;
< 0.001      1    nu0 = N/(2*Omega);
< 0.001      1    g2 = (gamma^2)/(3*N);
< 0.001      1    rate = pi*nu0*gavg;
  0.002      1    nl = 2*gavg./((1-omega).^2+(2*rate)^2); %The analytical prediction for the populations

                  %-The final plotting of the results:
                  % (i) Numerical long-time evolution (ii) Numerical GGE (iii) Analytical
                  % -%
  1.643      1    a1 = semilogy(dia1, te_result, 'o', "Color", 'b');
  0.008      1    hold on

  0.004      1    a2 = plot(dia1, gge_result(1:N), 'x', "LineWidth", 1.1, "Color","g");
  0.004      1    a3 = plot(omega, nl, "LineWidth", 1.2, "Color", "r");

  0.020      1    xlabel("$\omega/\Omega$", 'Interpreter',"latex", 'FontSize',18)
  0.007      1    ylabel("$n$", 'Interpreter',"latex", 'FontSize',18)
  1.160      1    legend([a1(1), a2(1), a3(1)], 'Long-time evolution', 'Numerical GGE','Analytical GGE', 'location', "northwest")
  0.007      1    ylim([0.5*10^(-5),10^(-1)])
  0.003      1    hold off
                  %%
                  %{
                  Uncomment the line below and change the path to your desired location to save the resulting data.
                  %}

                  %save("mutual_off", "te_result", "gge_result", "dia1")

  0.090      1    profile viewer
```

Other subfunctions in this file are not included in this listing.