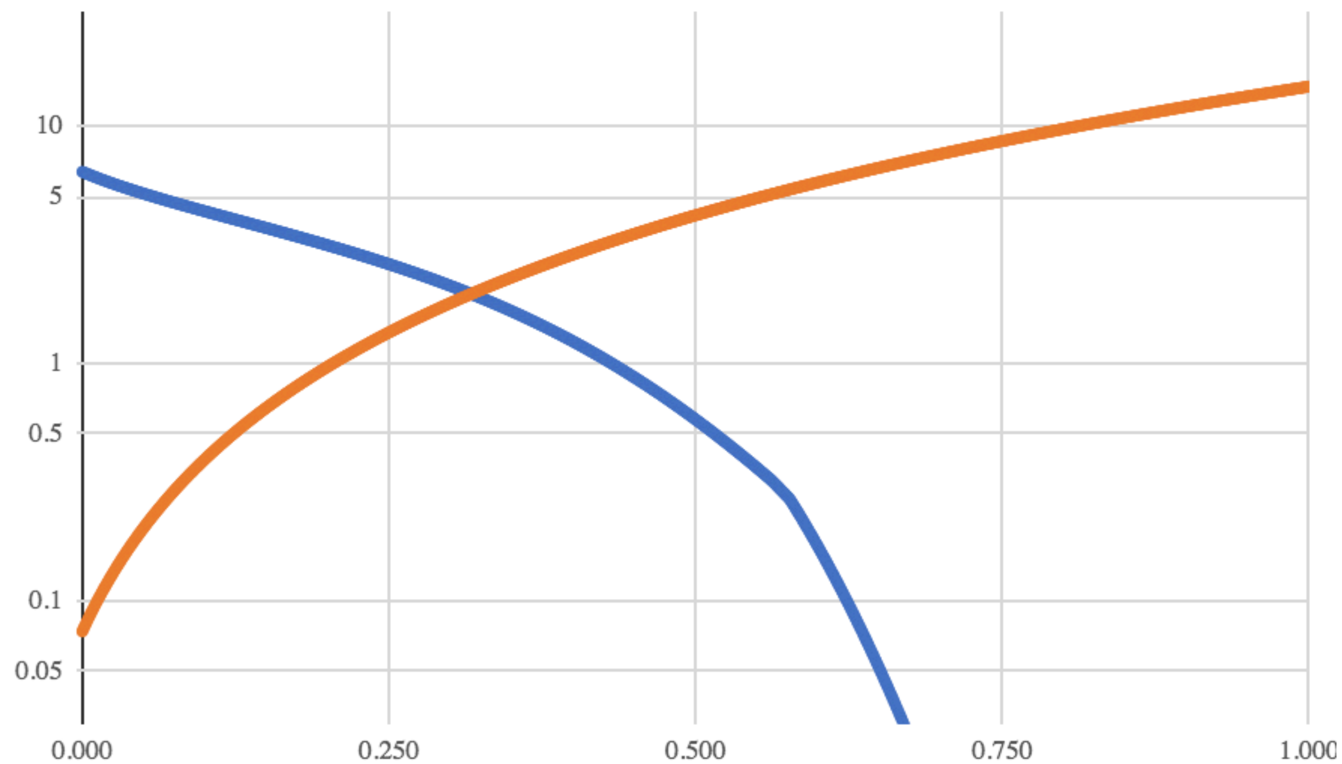
MATLAB CODE OF ADIABATIC MASTER EQUATION AND QUANTUM TRAJECTORIES

KA-WA YIP

$$\mathcal{H}_{ising} = -\frac{A(s)}{2}\left(\sum_i \hat{\sigma}_x^{(i)}\right) + \frac{B(s)}{2}\left(\sum_i h_i \hat{\sigma}_z^{(i)} + \sum_{i>j} J_{i,j}\hat{\sigma}_z^{(i)}\hat{\sigma}_z^{(j)}\right)$$

Form of Hamiltonian

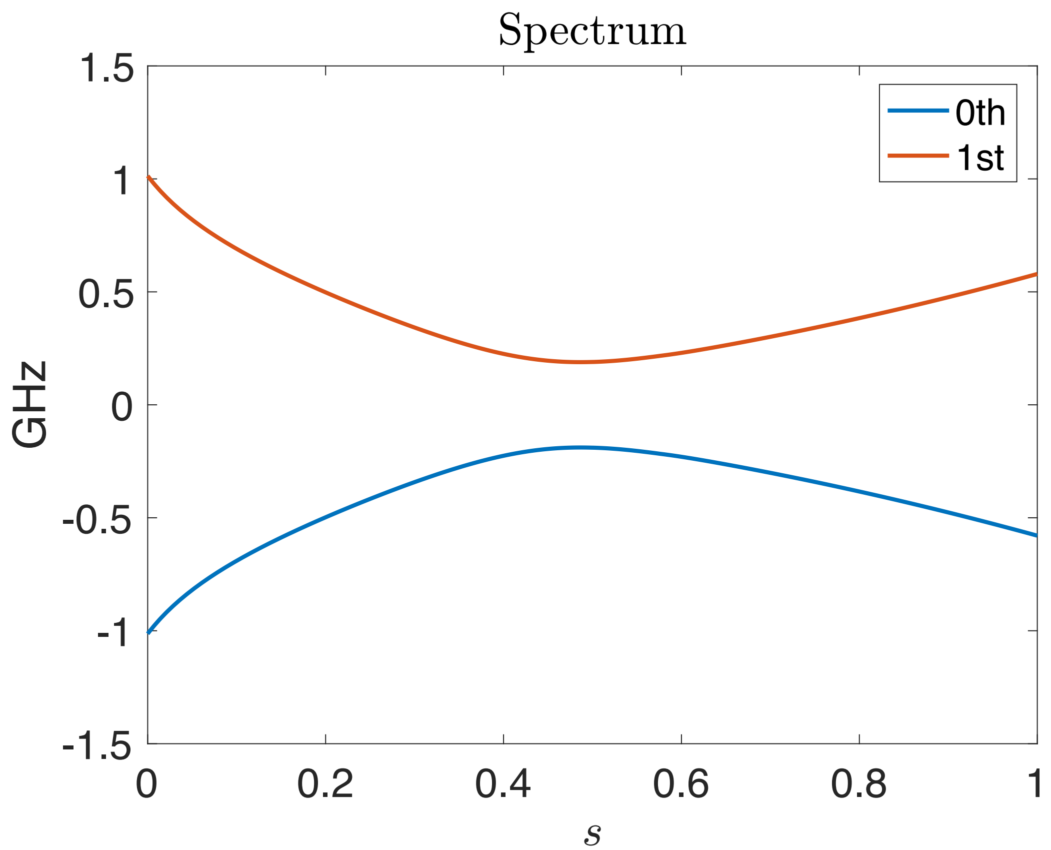


A(s) blue, B(s) orange

One-qubit example

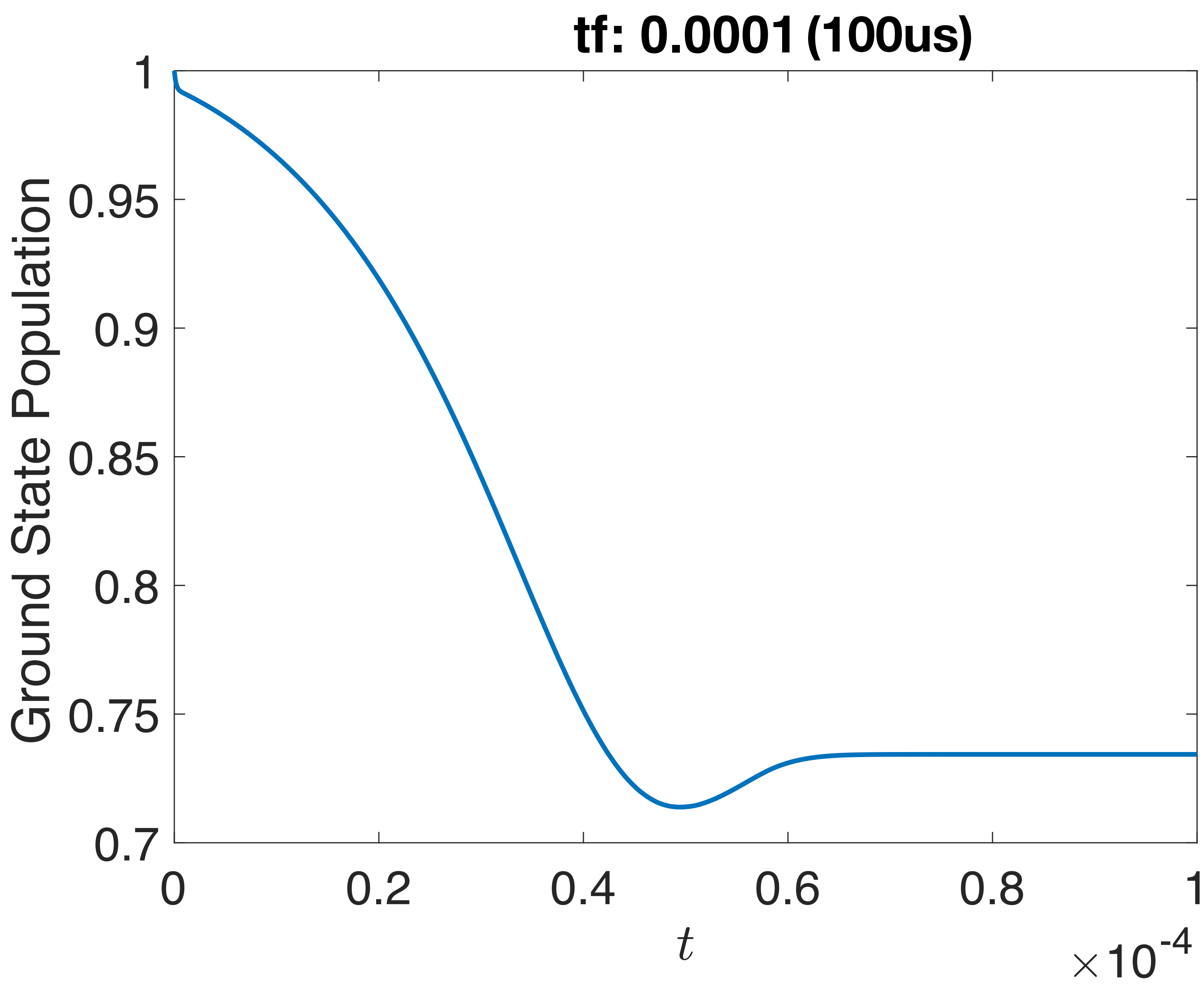
Ohmic bath,
DWave 2000Q schedule

$$T = 20\text{mK} \approx 2.6\text{GHz}$$
$$\eta g^2/(\hbar^2) = 1.2 \times 10^{-4}/(2\pi)$$



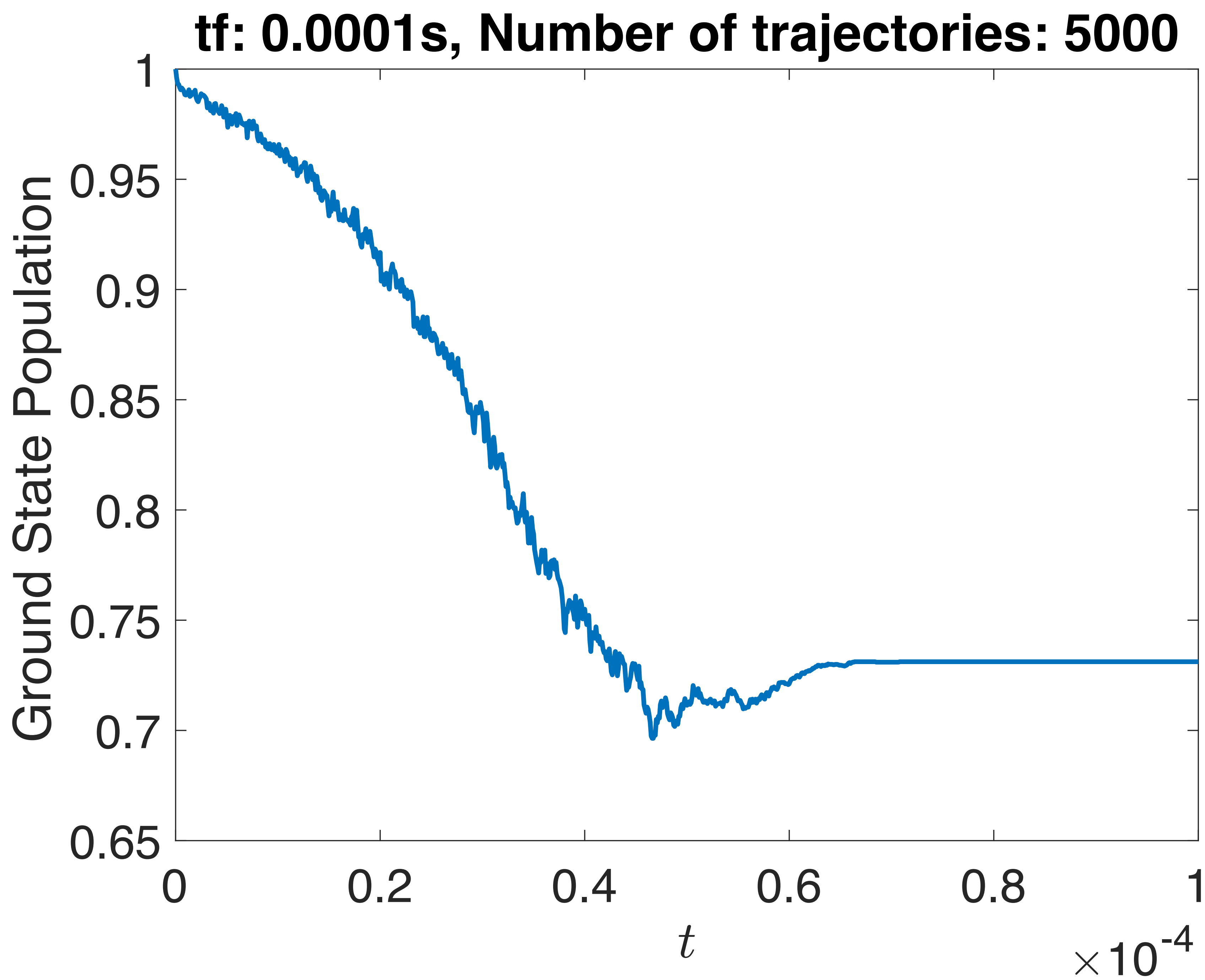
Spectrum

ame1qubit_demo.m



Master equation solution.

aqt1qubit_demo.m



Quantum traj. Solution

QT code feature

```
X = bsxfun(@minus, e.', e);   %matrix of \omega_{ba}
[sortedOutput,~,~] = uniquetol(full(reshape(X,[1 nevaltruc^2])),0.1,'DataScale',1);
length(sortedOutput(sortedOutput>0));
w_unique = length(sortedOutput);

dp = zeros(1, w_unique*4);
```

Sorting jump operators

```
    psi = v*psicb;
    %Change back to comp.basis
else % Rigorously should have implemented backtrack:
     % collapse has occured:
     % find collapse time to within specified tolerance
     % ------------------------------------------------
     % Rigorously should have implemented backtrack:
     % collapse has occured:
     % find collapse time to within specified tolerance
     % ------------------------------------------------
    t_prev = tstep_qt(index);
    t_final = tstep_qt(index) + dt_qt;
    %r1;
    ii = 0;
    while ii < 5
        ii = ii + 1;
        t_guess = t_prev + (log(norm2_prev/r1)/log(norm2_prev/norm2_unpsi)) *
        %t_guess - t_prev
        unpsi_guess = expm(-1i*(t_guess - t_prev)*H_eff/hbar)*unpsi_prev;
        %norm2_guess = norm(unpsi_prev)^2
        norm2_guess = norm(unpsi_guess)^2;
        if abs(r1 - norm2_guess) < 0.001*r1  %error tolerance
            break
        elseif (norm2_guess < r1)
            t_final = t_guess;
            norm2_unpsi = norm2_guess;
        else
```

Backtracking

ME code feature

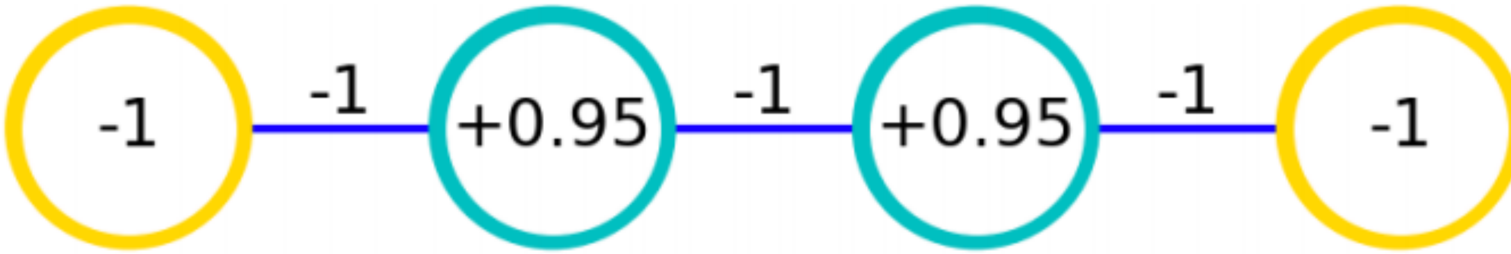
```
Lpcomponent1 = matrixelement1*sparse(a(s),b(s),1,nevaltruc,nevaltruc);
Lpcomponent2 = matrixelement2*sparse(a(s),b(s),1,nevaltruc,nevaltruc);
Lpcomponent3 = matrixelement3*sparse(a(s),b(s),1,nevaltruc,nevaltruc);
Lpcomponent4 = matrixelement4*sparse(a(s),b(s),1,nevaltruc,nevaltruc);
```

Sparse matrix

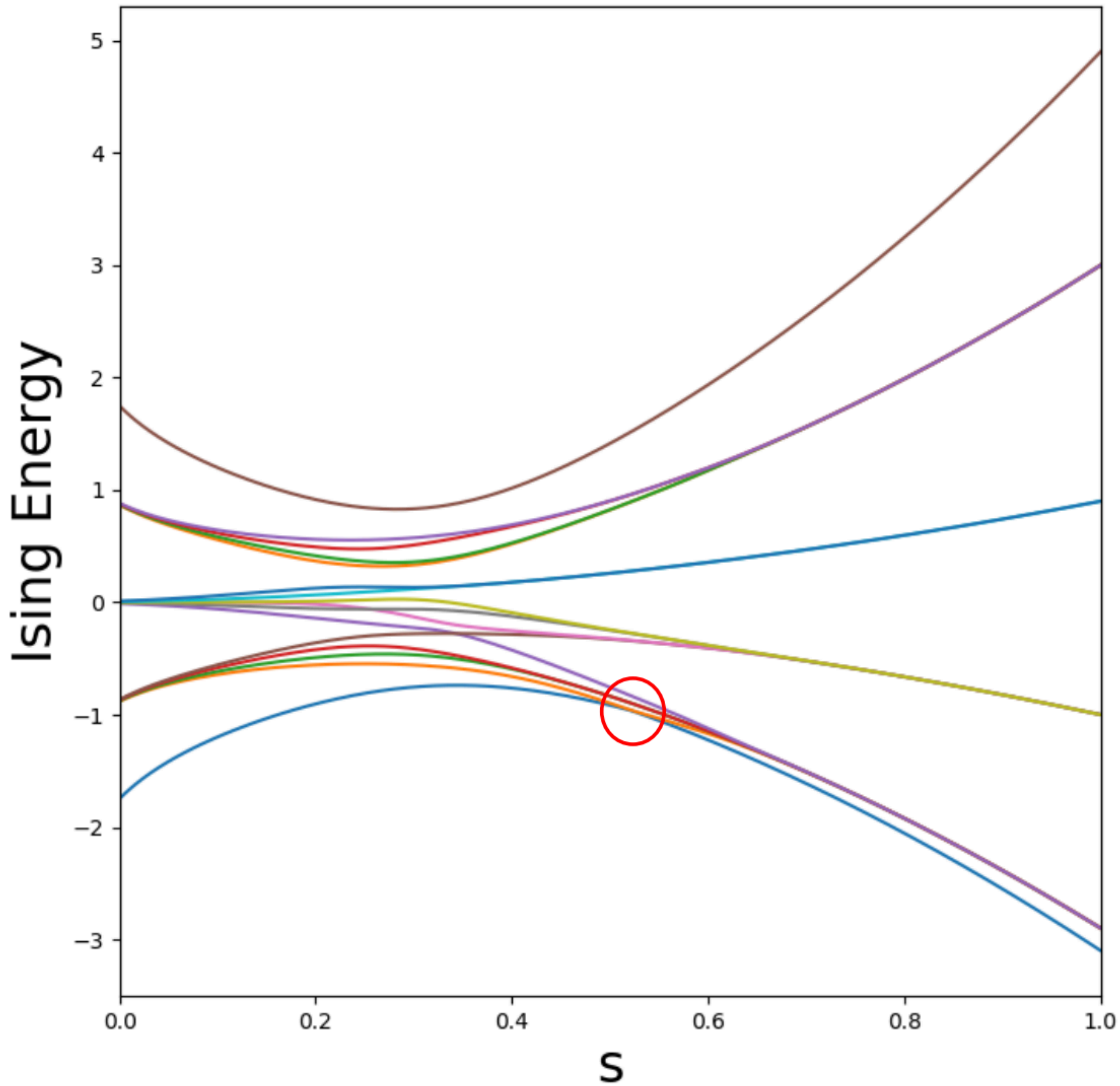
```
= v(:,1);
m = reshape(rho,[neval,neval]);
mcb = sparse(v'*rhom*v);


elity = rhomcb(1,1);
elitylist_me(1, index) = fidelity;

ndblad  = @(t, rho)lindblad(t, rho, Hsd, natom, gsq2pi, beta, betainv, wc, A_1,A
  = [tstep_me(index), tstep_me(index) + dt_me];
ions = odeset('RelTol',1e-3,'AbsTol',1e-6);
, RHO] = ode45(alindblad, t, rhomcb(:));
 RHO] = ode45(alindblad, t, rhomcb(:),options);
inal value of rho is initial value for next step:
 = RHO(end, :);
 = v*reshape(rho,[nevaltruc,nevaltruc])*v';   %in computational basis
 = rho(:);
```

Ode with rotations

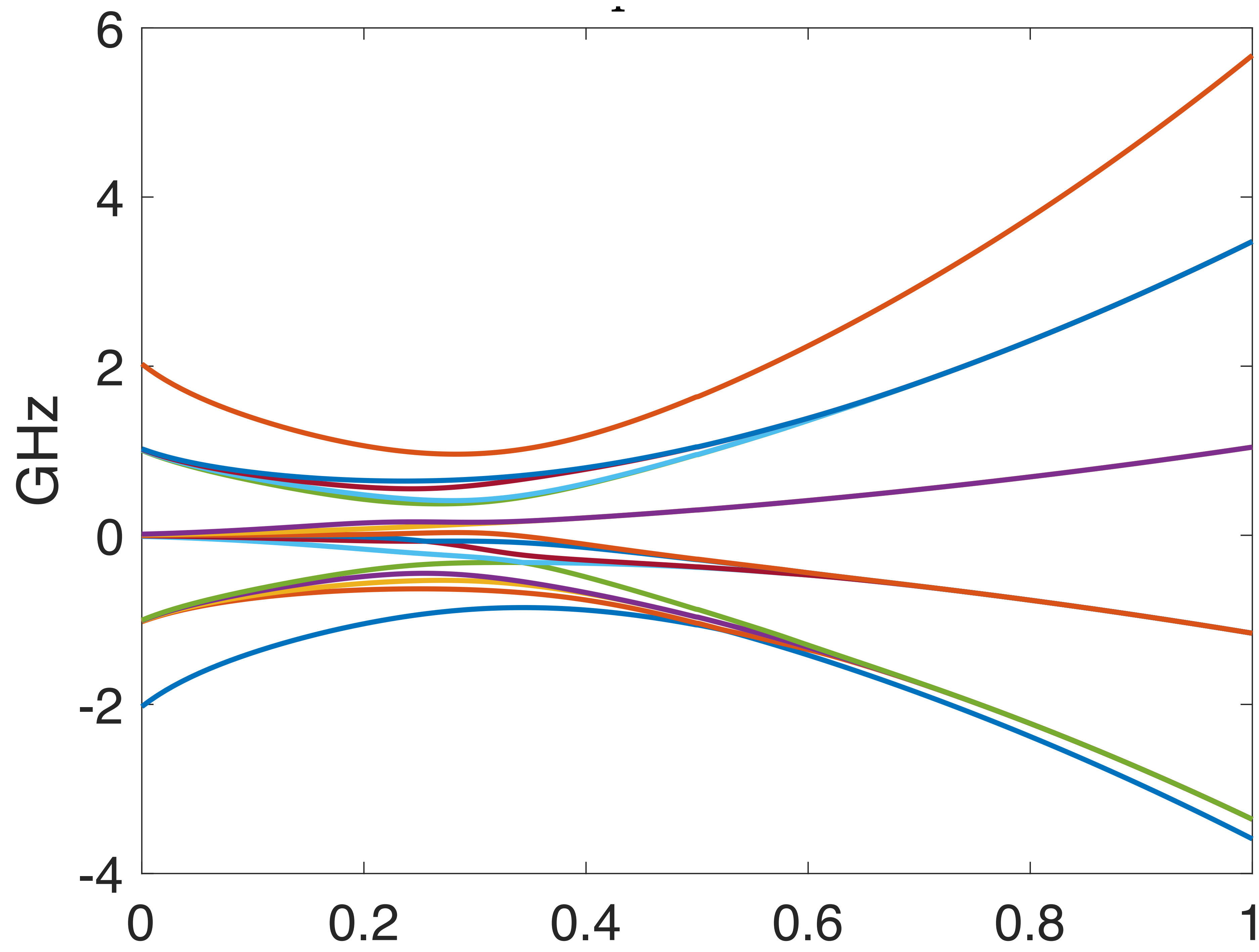
UCL 4-qubit gadget




```
Hs = -1e9.*A_sp1(inslist(index)).*(sX_1+sX_2+sX_3+sX_4) + 1e9.*B_sp1(inslist(index)).*(((-1).*sZ_1+(0.95).*sZ_2+(0.95).*sZ_3+(-1).*sZ_4) + ...
    ((-1).*sZsZII + (-1).*IsZsZI + (-1).*IIsZsZ));
```



ame_4qubit_reverse_annealing_spectrum.m

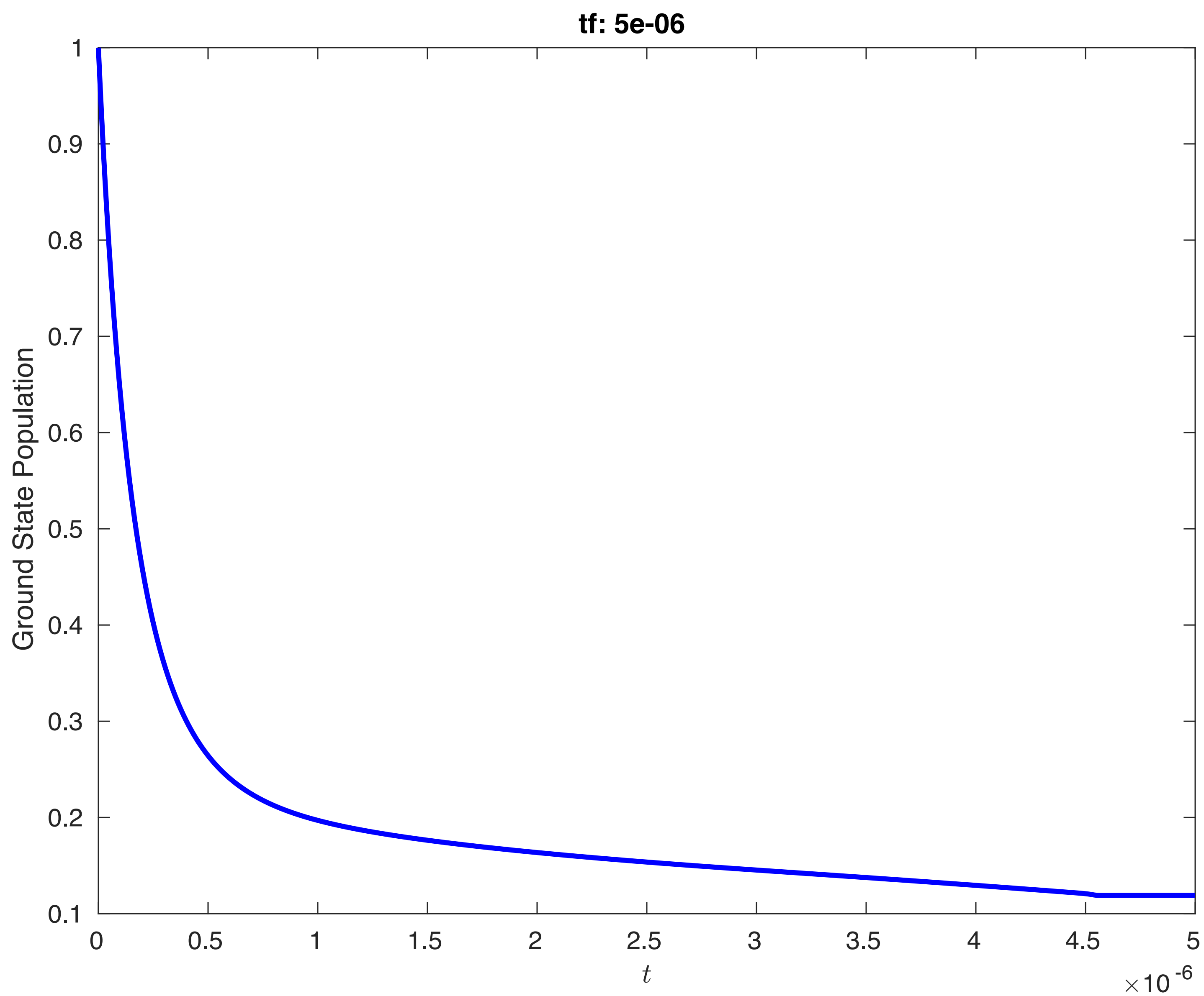


Reproducing the spectrum

Assume the following form and use linear schedule

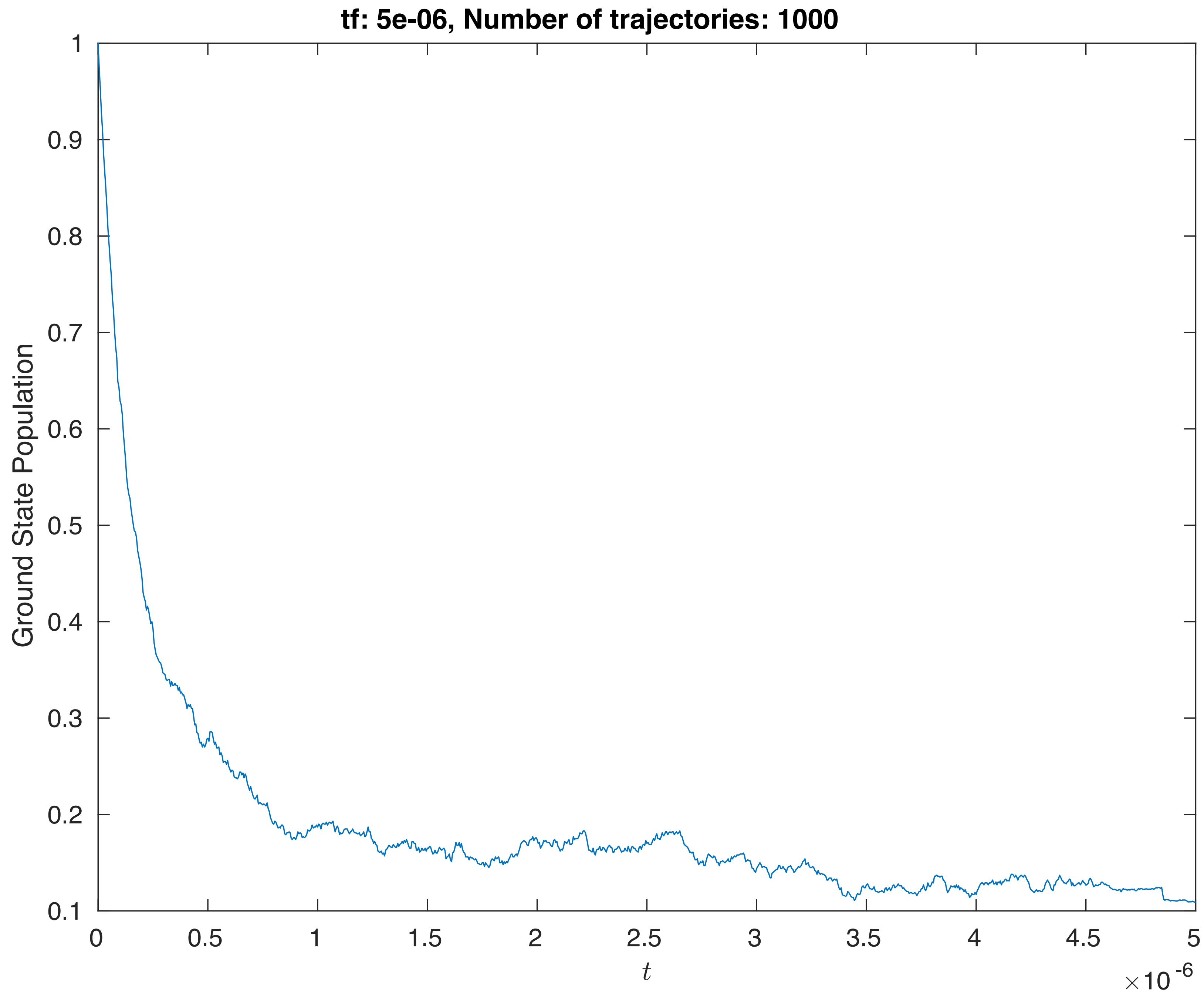
$$\mathcal{H}_{ising} = -\frac{A(s)}{2}\left(\sum_i \hat{\sigma}_x^{(i)}\right) + \frac{B(s)}{2}\left(\sum_i h_i \hat{\sigma}_z^{(i)} + \sum_{i>j} J_{i,j}\hat{\sigma}_z^{(i)}\hat{\sigma}_z^{(j)}\right)$$

linearschedule/ame_4qubit.m



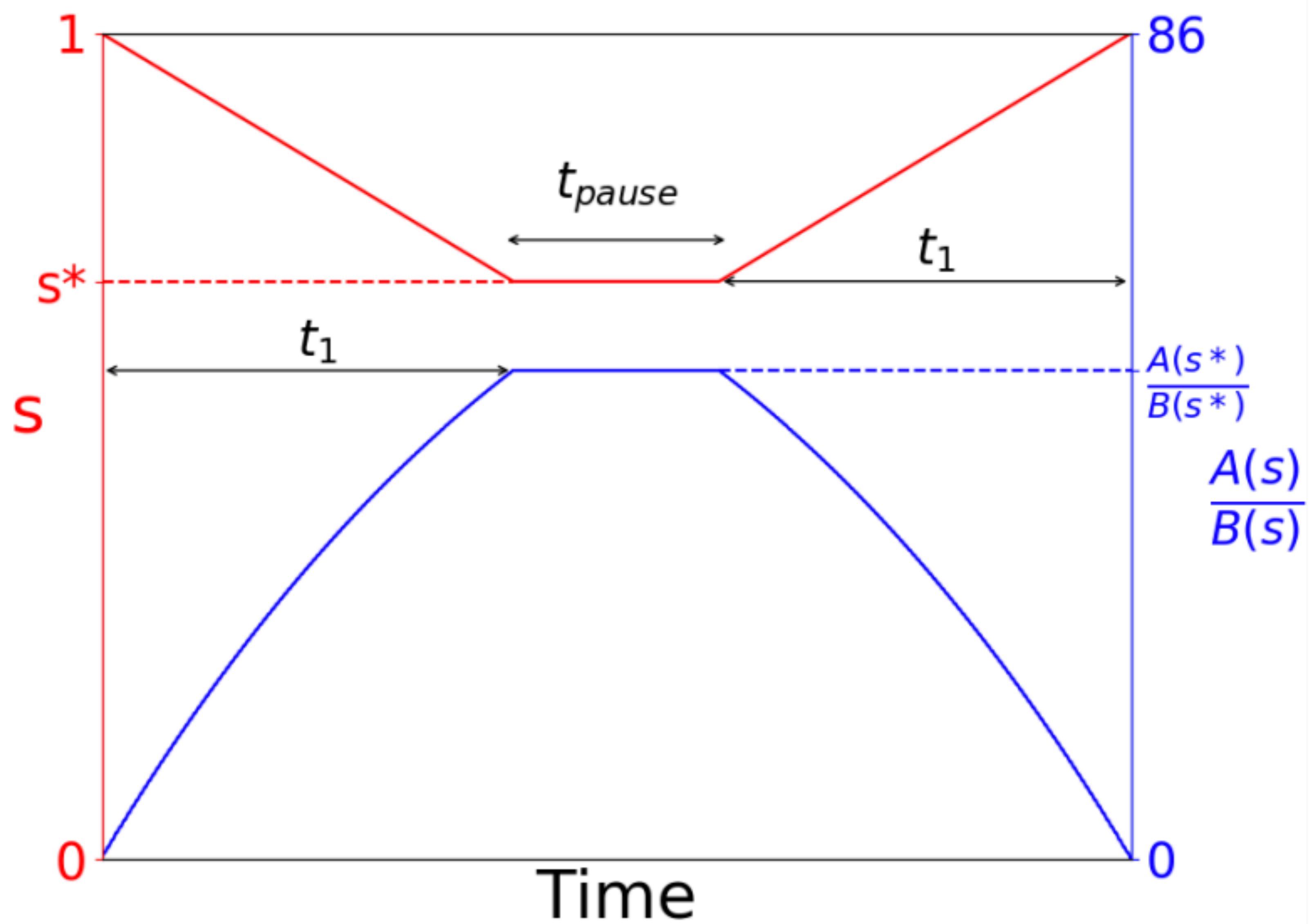
ME solution

linearschedule/aqt_4qubit.m



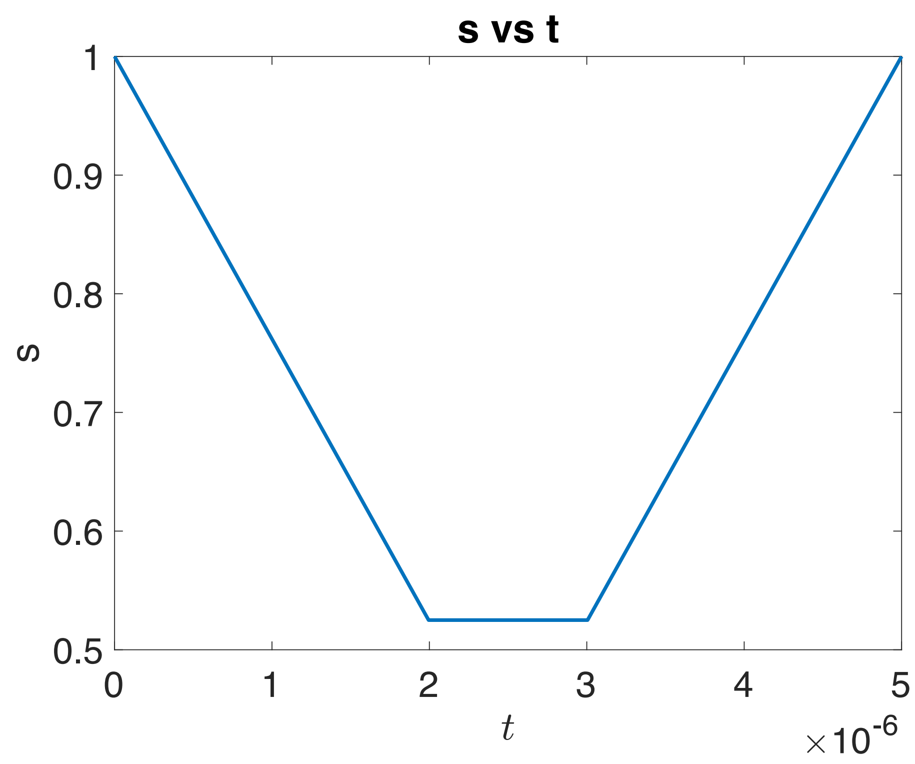
QT solution

Reverse Annealing

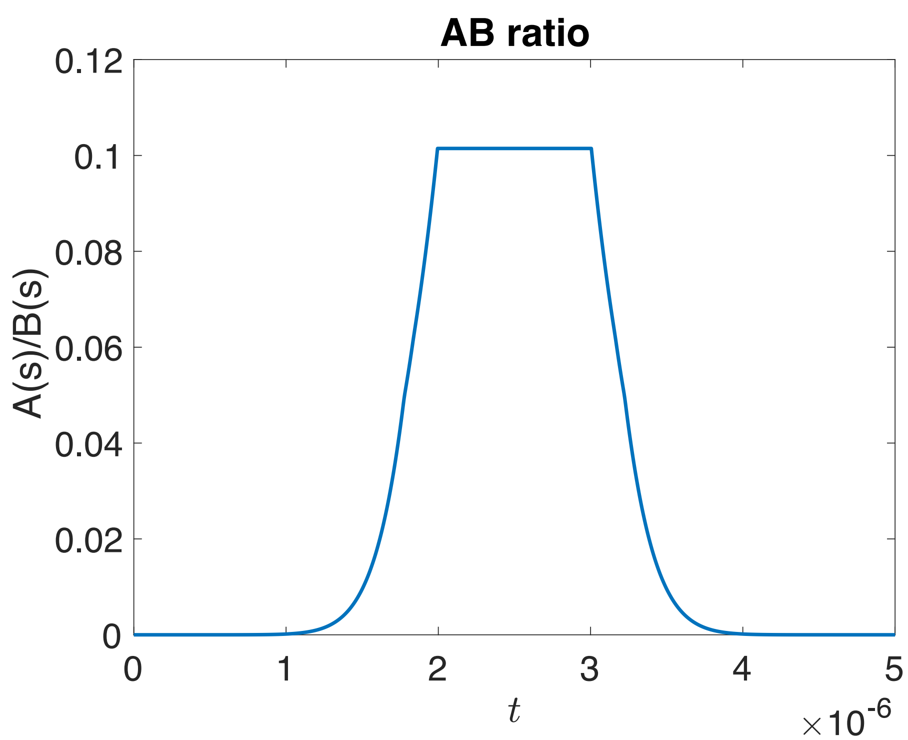


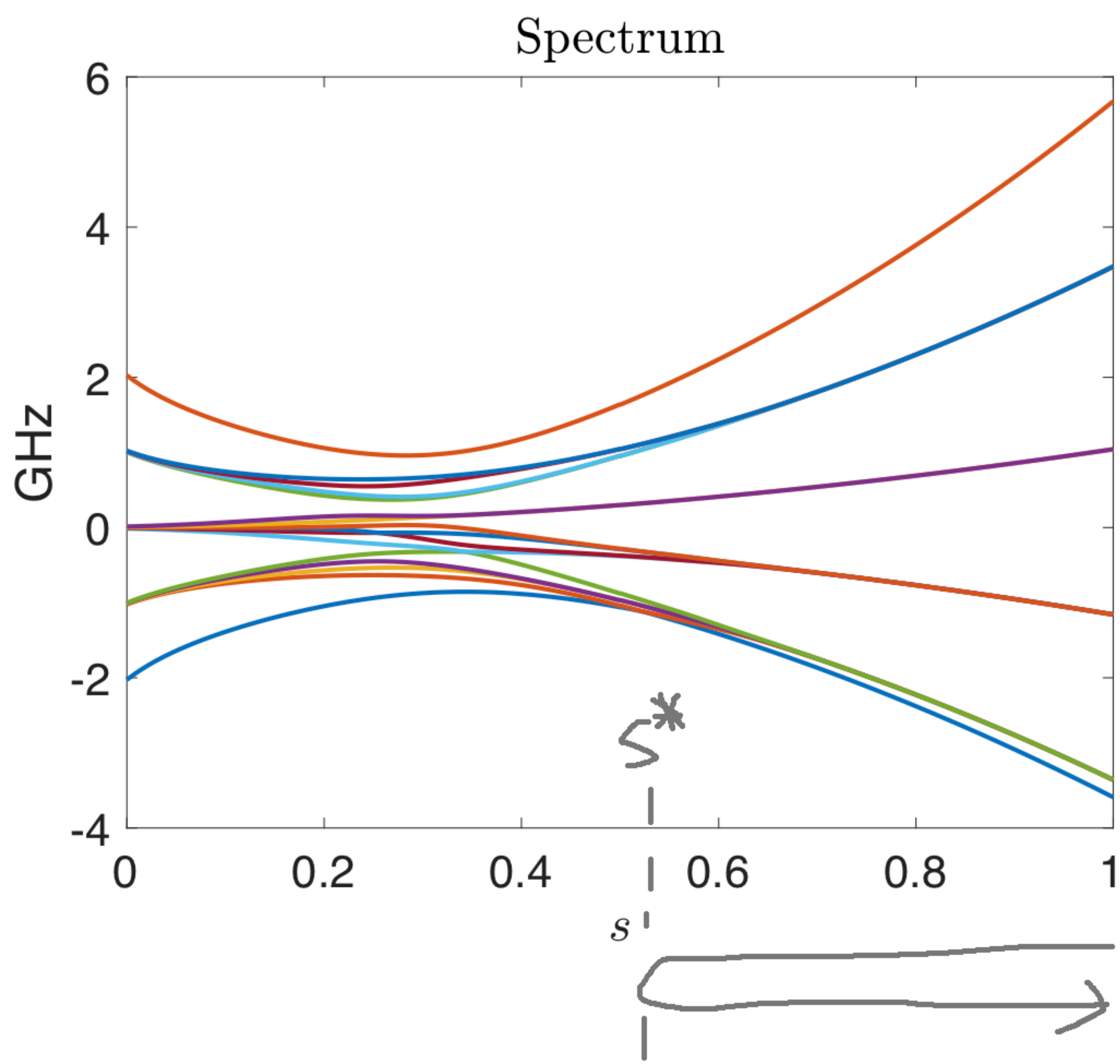
t1 = 2us. So total is around 5us.

Reproducing the previous graph



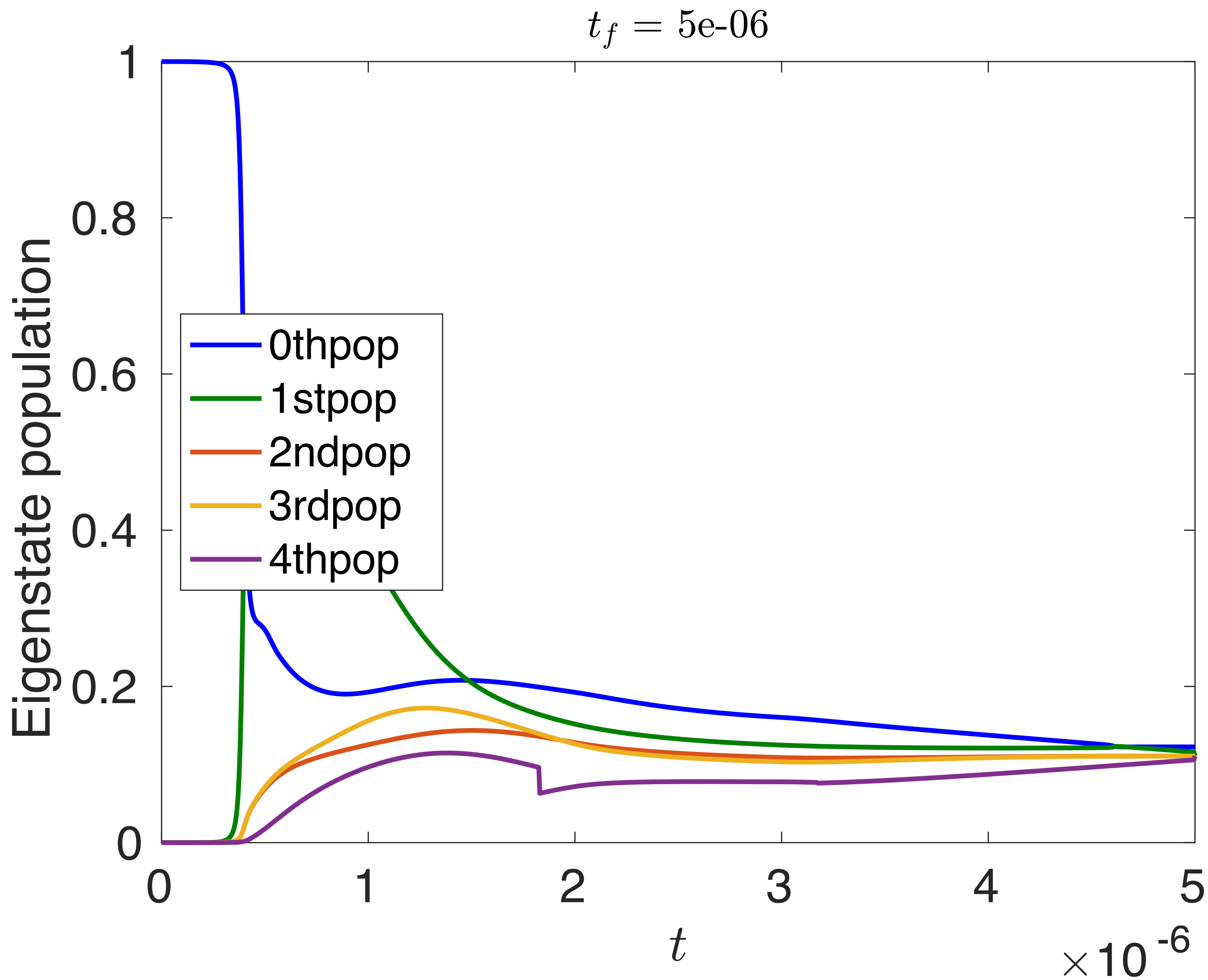

```
step = 1000;
sstar = 0.525;
inslist1 = linspace(1,sstar,step*2/5);
inslist2 = linspace(sstar,sstar,step*1/5+1);
inslist3 = linspace(sstar,1,step*2/5);
inslist = [inslist1, inslist2, inslist3];
```



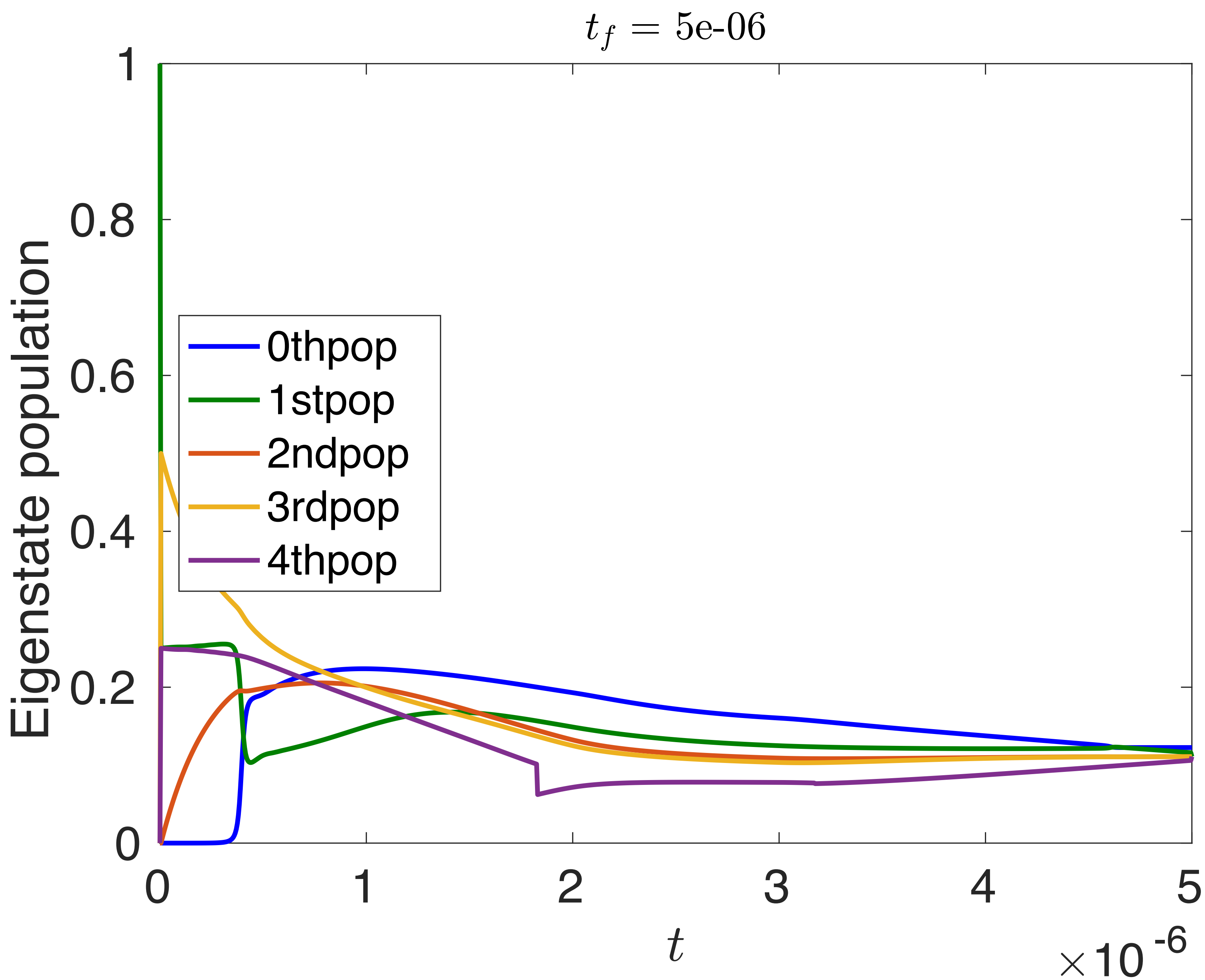


Spectrum
GHz
6
4
2
0
-2
-4
0
0.2
0.4
0.6
0.8
1
s*
s

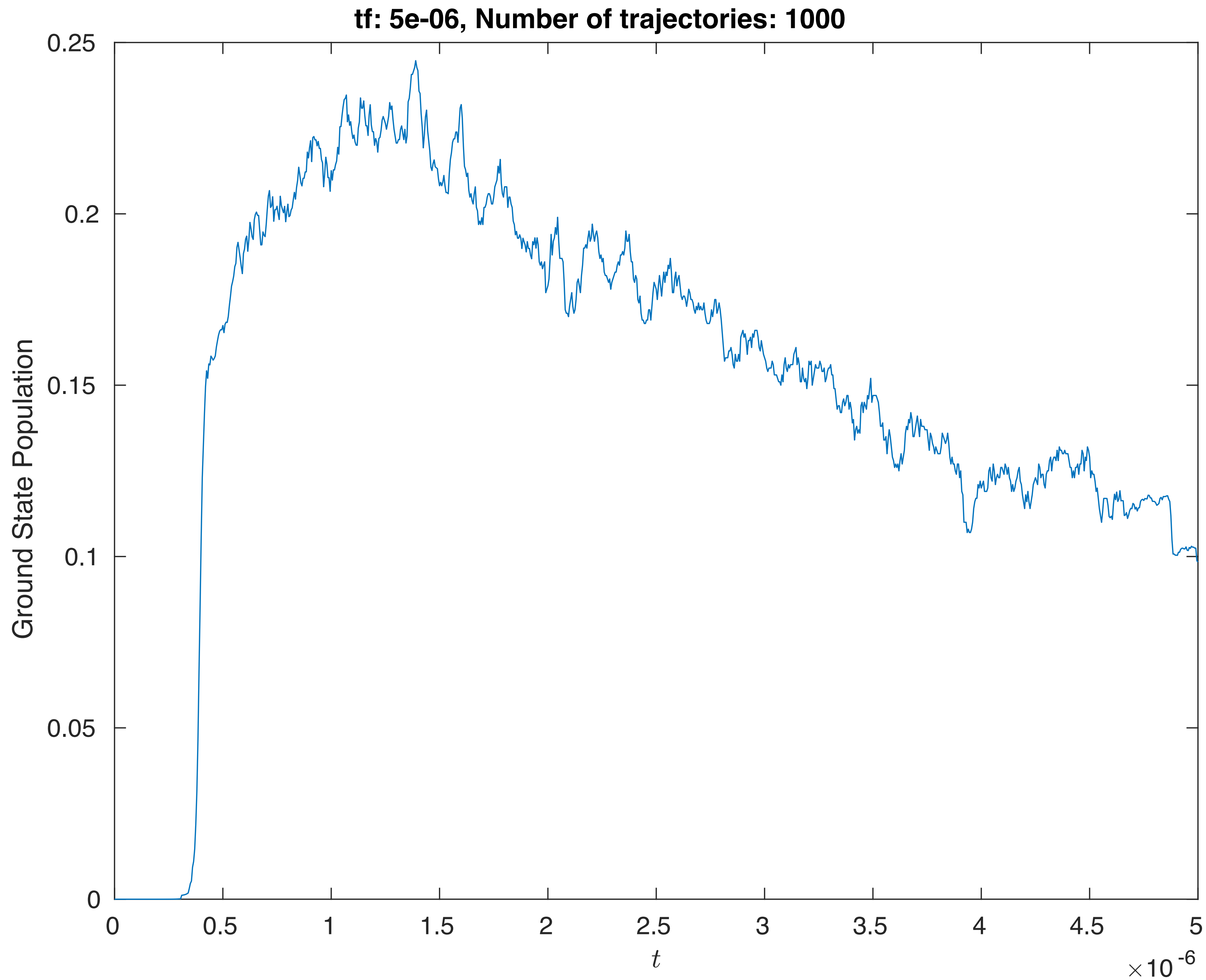
s*=0.525, tf = 50us, linear schedule, start at ground state of the Hp



s*=0.525, tf = 50us, linear schedule, start at first excited state of the Hp

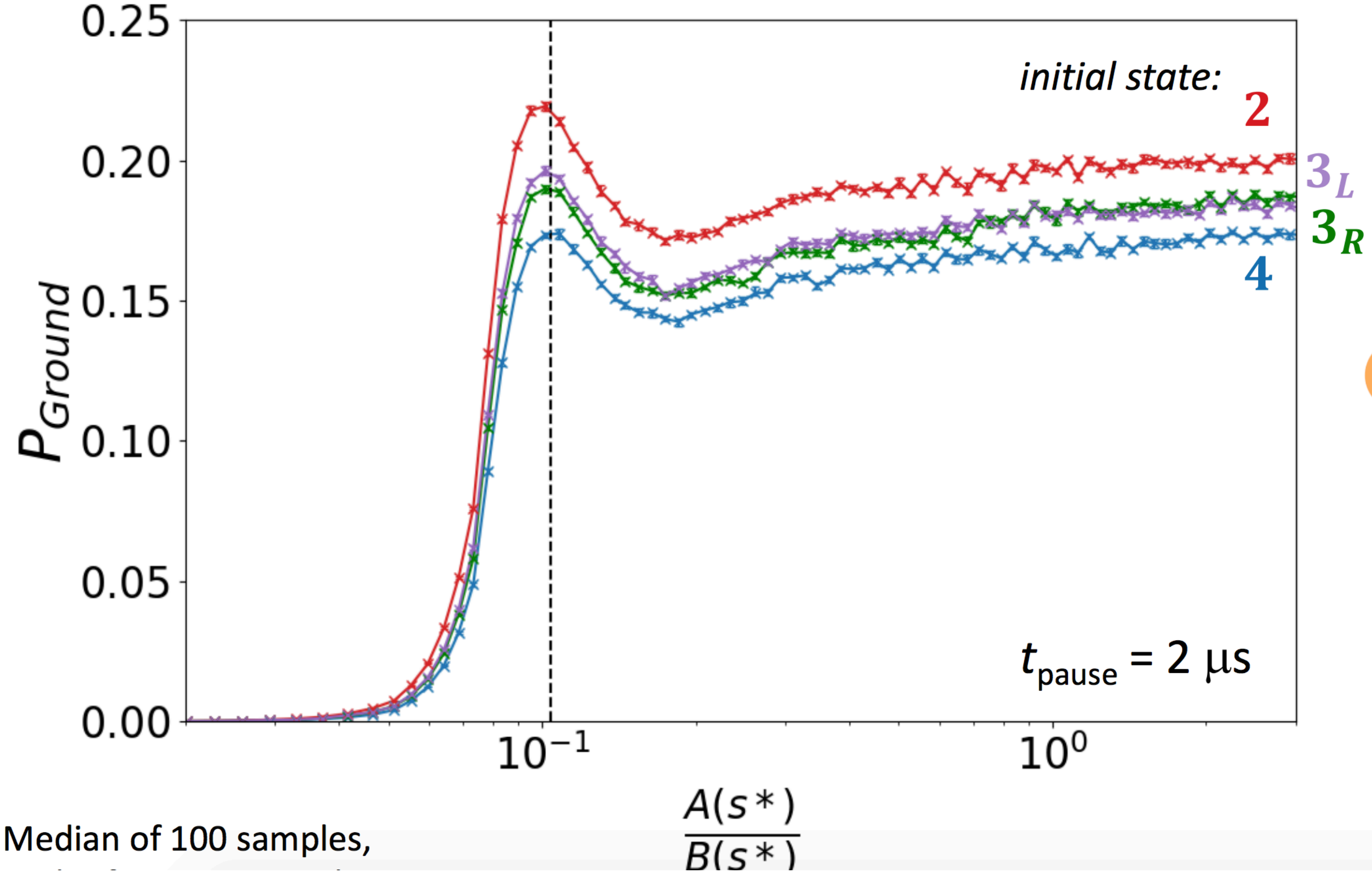


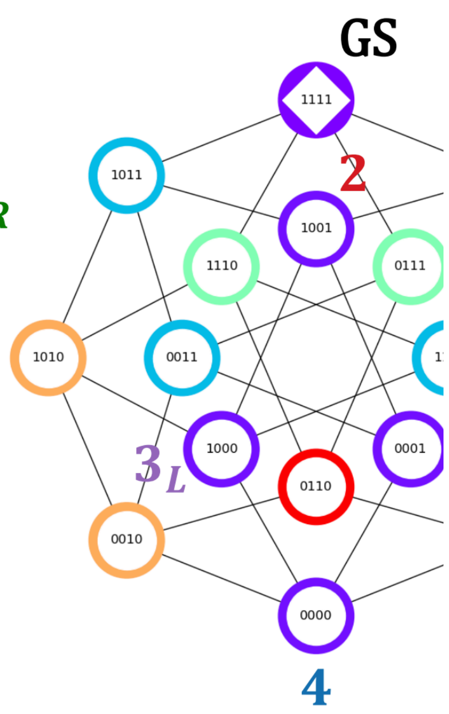
s*=0.525, tf = 50us, linear schedule, start at first excited state of the Hp



Ground state population from QT

Results from D-Wave 2000Q (with pause)





Median of 100 samples,

```matlab
function aqt_4qubit_reverse_annealing_linear
% cluster = parallel.cluster.Generic;
% set(cluster,'JobStorageLocation', '/home/rcf-proj2/ky/kawayip/research/fourqubitgadget_linear/qt');
% set(cluster,'HasSharedFilesystem', true);
% set(cluster,'IntegrationScriptsLocation','/usr/usc/matlab/R2018b/SlurmIntegrationScripts');
% cluster.AdditionalProperties.SlurmArgs='--time=23:59:59';

cluster = get_LOCAL_cluster('/home/rcf-proj2/ky/kawayip/research/fourqubitgadget_reverse_annealing/linear/qt');

pool=parpool(cluster,15)
```

Add the above lines for computing in the hpc cluster