

December 16, 2014

Notes on the 12 qubit PPS

Dawei Lu

Notes about the problems in the 12 qubit PPS preparation, including Matlab codes and Experiments.

DEC 12, 2014

Calculating the state to state GRAPE on Ordi2. In pulsefinder folder. paramsfile is 'twqubit_subS2S.m', and the output file is 'twqubit_7zto12z'.

The GRAPE is to evolve ZZZZZZIIII to ZZZZZZZZZZ. As the couplings between nearest-neighbored C and H are about 150Hz. I set the GRAPE

```

1 % Number of timesteps
2 params.plength = 400;
3
4 % Length of each time step
5 params.timestep = 10e-6;
6
7 params.subsystem{1} = [1 2 3 9 10 11];
8 params.subsystem{2} = [4 5 6 7 8 12];
9 params.subsys_weight = [6 6];
10
11 % Input and goal states for state to state
12 params.rhoin = mkstate('+1ZZZZZZIIII',1);
13 params.rhogoal = mkstate('+1ZZZZZZZZZZ',1);
14
15 % Allow Zfreedom or not
16 params.Zfreedomflag = 1;

```

The fidelity keeps 0 all the time. Guess the reason is 'Zfreedom'. Set 'params.Zfreedomflag = 0;'. However, still 0.

Annie said maybe due to the length. Her SWAP gate requires 8ms, so I changed 'params.plength = 800;'. But for with or without Zfreedom, fidelity is still 0.

Check if some of my GRAPE settings are wrong. try to repeat Annie's SWAP gate calculation.

```

1 % Number of timesteps
2 params.plength = 800;
3
4 % Length of each time step
5 params.timestep = 10e-6;
6
7 params.subsystem{1} = [1 2 3 9 10 11];
8 params.subsystem{2} = [4 5 6 7 8 12];
9 params.subsys_weight = [6 6];
10
11 % Input and goal states for state to state
12 params.rhoin = mkstate('+1IIIIIIIZIIII+1IIIIIIIZIII+1IIIIIIIZII+1IIIIIIIZI
    +1IIIIIIIZ',1);
13 params.rhogoal = mkstate('+1IIIIIIIZIIII+1IIIIIIIZIII+1IIIIIIIZII+1
    IIIIIIIIZI+1IIIIIIIZIIIZ',1);
14
15 % Allow Zfreedom or not
16 params.Zfreedomflag = 0;

```

The outputfile is 'twqubit_SWAPC7H5'. And the fidelity is already over 98%. Then I changed 'params.Zfreedomflag = 1;', and the fidelity is over 95% after 30 iterations. Much slower than the no Zfreedom case. Maybe due to different initial guesses.

DEC 15, 2014

Generate all $\pi/2$ and π pulses for the 7 Carbons, with the Calibration = 25KHz. $\pi/2$ pulses are 1ms length and 100 steps, and π pulses are 2ms length and 200 steps. Generating Code in 'twqubit_shape.m'

```

1 for ii = 1:7
2 loadfile = ['twqubit_C', num2str(ii), '180', '.mat'];
3 eval(['load ', loadfile]);
4 filename1 = ['twqubit_C', num2str(ii), '180_C_25000.txt'];
5 filename2 = ['twqubit_C', num2str(ii), '180_H_25000.txt'];
6 make_brucker_shape(pulses{1}, 25000, filename1,1);
7 make_brucker_shape(pulses{1}, 25000, filename2,2);
8 end

```

The pulses are saved in Ordi2 '\pulsefinder\12 Qubit\' with the names such as 'twqubit_C590_C_25000.txt'.

I checked all the fidelities of the $\pi/2$ pulses in the folder '\pulseexam_12qubit\C_rotations\check_grape.m'. The code is

```

1 load Para.mat
2 load twpauliX_full.mat
3 load twpauliY_full.mat
4
5 %% Check all 90 rotations
6 %% Parameters for the GRAPE pulse
7 for spin_number = 1:7
8 Name1 = ['twqubit_C', num2str(spin_number), '90_C_25000.txt'];
9 Name2 = ['twqubit_C', num2str(spin_number), '90_H_25000.txt'];
10 Amplitude = 25000;
11 Time = 1e-3;
12 Length = 100;
13 dt = Time/Length;
14 FirstLine = 19; % the first line which contains the information of power and
    phase
15
16 Output1 = 'test1';
17 Output2 = 'test2';
18
19 [power1,phase1]=dataout(Name1,Output1,FirstLine,Length);
20 [power2,phase2]=dataout(Name2,Output2,FirstLine,Length);
21 %% Check
22 X_C = 0; Y_C = 0;
23 for jj = 1:7
24     X_C = X_C + KIx{jj};
25     Y_C = Y_C + KIy{jj};
26 end
27
28 X_H = 0; Y_H = 0;
29 for jj = 8:12
30     X_H = X_H + KIx{jj};
31     Y_H = Y_H + KIy{jj};
32 end
33
34
35 U = eye(2^12);
36 U = U*expm(-i*H*4e-6);
37 for ii = 1:Length

```

```

38     Hext = 2*pi*(Amplitude*power1(ii)/100)*(X_C*cos(phase1(ii)/360*2*pi)-Y_C*sin
39         (phase1(ii)/360*2*pi))+2*pi*(Amplitude*power2(ii)/100)*(X_H*cos(phase2(ii)
40         )/360*2*pi)-Y_H*sin(phase2(ii)/360*2*pi));
41     U = expm(-i*(Hext+H)*dt)*U;
42 end
43 U = U*expm(-i*H*4e-6);
44
45 Utar = expm(-i*KIx{spin_number}*pi/2);
46
47 % Fidelity = ['Fidelity_C', num2str(spin_number), '90'];
48 % eval(['Fidelity_C', num2str(spin_number), '90 = abs(trace(U*Utar'))/2^12']);
49 Fidelity = abs(trace(U*Utar))/2^12
50
51 savefile = ['twqubit_C', num2str(spin_number), '90_Ufid.mat'];
52 save (savefile, 'U', 'Fidelity');
53 end

```

Unitaries and Fidelities of the pulses will both be saved in 'twqubit_C590_Ufid.mat', so they can be called for further calculations in the PPS simulation. Wait for the results.

DEC 16, 2014

Combine pulses in the PPS preparation into big shape files, which should be easy for calibrations and pulsefixing.

The code is in the SVN server for Matlab named '`\Twqubit\pulse_combine.m`'.

First read all the powers and phases for the $\pi/2$ and π rotations.

```

1 for spin_number = 1:7
2     Name1 = ['twqubit_C', num2str(spin_number), '90_C_25000.txt'];
3     Name2 = ['twqubit_C', num2str(spin_number), '90_H_25000.txt'];
4     [power1,phase1]=dataout(Name1,Output1,FirstLine,Length_90);
5     [power2,phase2]=dataout(Name2,Output2,FirstLine,Length_90);
6     eval(['power_C', num2str(spin_number), '90_C = power1;']); eval(['phase_C
7         ', num2str(spin_number), '90_C = phase1;']);
8     eval(['power_H', num2str(spin_number), '90_H = power2;']); eval(['phase_H
9         ', num2str(spin_number), '90_H = phase2;']);
10 end
11
12 for spin_number = 1:7
13     Name1 = ['twqubit_C', num2str(spin_number), '180_C_25000.txt'];
14     Name2 = ['twqubit_C', num2str(spin_number), '180_H_25000.txt'];
15     [power1,phase1]=dataout(Name1,Output1,FirstLine,Length_180);
16     [power2,phase2]=dataout(Name2,Output2,FirstLine,Length_180);
17     eval(['power_C', num2str(spin_number), '180_C = power1;']); eval(['
18         phase_C', num2str(spin_number), '180_C = phase1;']);
19     eval(['power_H', num2str(spin_number), '180_H = power2;']); eval(['
20         phase_H', num2str(spin_number), '180_H = phase2;']);
21 end

```

Then combine them with the free evolutions. Here I set the time step $dt = 10\mu s$.

```

1 %% From Z7 to Z24567
2 step_27 = round(1/(4*Para(2,7))/dt);
3 step_67_27 = round((1/(4*Para(6,7))-1/(4*Para(2,7)))/dt);
4 step_47_67 = round((1/(4*Para(4,7))-1/(4*Para(6,7)))/dt);
5 step_57_47 = round((1/(4*Para(5,7))-1/(4*Para(4,7)))/dt);
6 step_57 = round((1/(4*Para(5,7)))/dt);
7
8 power_encoding1_C = [power_C790_C; zeros(step_27,1);power_C2180_C; zeros(
9     step_67_27,1); power_C6180_C; zeros(step_47_67,1);power_C4180_C; zeros(
10    step_57_47,1);...
11                                power_C5180_C; power_C7180_C; zeros(step_57,1)
12                                ;power_C790_C]*Calibration/Calibration_old;
13 phase_encoding1_C = [phase_C790_C; zeros(step_27,1);phase_C2180_C; zeros(
14    step_67_27,1); phase_C6180_C; zeros(step_47_67,1);phase_C4180_C; zeros(
15    step_57_47,1);...
16                                phase_C5180_C; phase_C7180_C; zeros(step_57,1)
17                                ;mod((phase_C790_C+90),360)];
18 power_encoding1_H = [power_H790_H; zeros(step_27,1);power_H2180_H; zeros(
19    step_67_27,1); power_H6180_H; zeros(step_47_67,1);power_H4180_H; zeros(
20    step_57_47,1);...
21                                power_H5180_H; power_H7180_H; zeros(step_57,1)
22                                ;power_H790_H]*Calibration/Calibration_old;
23 phase_encoding1_H = [phase_H790_H; zeros(step_27,1);phase_H2180_H; zeros(
24    step_67_27,1); phase_H6180_H; zeros(step_47_67,1);phase_H4180_H; zeros(
25    step_57_47,1);...

```

```

15         phase_H5180_H; phase_H7180_H; zeros(step_57,1)
16         ;mod((phase_H790_H+90),360)];
17 total_time_encoding1 = length(power_encoding1_C)*dt;
18
19 outputfile = 'twqubit_encoding1_C';
20 shpfile = fopen(outputfile,'w');
21 fprintf(shpfile, '##TITLE= %s\n',outputfile);
22 fprintf(shpfile, '##JCAMP-DX= 5.00 Bruker JCAMP library\n');
23 fprintf(shpfile, '##DATA TYPE= Shape Data\n');
24 fprintf(shpfile, '##ORIGIN= Dawei's GRAPE Pulses \n');
25 fprintf(shpfile, '##OWNER= Dawei\n');
26 fprintf(shpfile, '##DATE= %s\n',date);
27 time = clock;
28 fprintf(shpfile, '##TIME= %d:%d\n',fix(time(4)),fix(time(5)));
29 fprintf(shpfile, '##MINX= %7.6e\n',min(power_encoding1_C));
30 fprintf(shpfile, '##MAXX= %7.6e\n',max(power_encoding1_C));
31 fprintf(shpfile, '##MINY= %7.6e\n',min(phase_encoding1_C));
32 fprintf(shpfile, '##MAXY= %7.6e\n',max(phase_encoding1_C));
33 fprintf(shpfile, '##$SHAPE_EXMODE= None\n');
34 fprintf(shpfile, '##$SHAPE_TOTROT= %7.6e\n',90);
35 fprintf(shpfile, '##$SHAPE_BWFAC= %7.6e\n',1);
36 fprintf(shpfile, '##$SHAPE_INTEGFAC= %7.6e\n',1);
37 fprintf(shpfile, '##$SHAPE_MODE= 1\n');
38 fprintf(shpfile, '##PULSE_WIDTH= %d\n',total_time_encoding1);
39 fprintf(shpfile, '##Calibration_Power= %d\n',Calibration);
40 fprintf(shpfile, '##NPOINTS= %d\n',length(power_encoding1_C));
41 fprintf(shpfile, '##XYPOINTS= (XY..XY)\n');
42
43 for ii = 1:length(power_encoding1_C)
44     fprintf(shpfile, ' %7.6e, %7.6e\n',power_encoding1_C(ii),phase_encoding1_C(
45         ii));
46
47 end
48
49 fprintf(shpfile, '##END=\n');
50
51 outputfile = 'twqubit_encoding1_H';
52 shpfile = fopen(outputfile,'w');
53 fprintf(shpfile, '##TITLE= %s\n',outputfile);
54 fprintf(shpfile, '##JCAMP-DX= 5.00 Bruker JCAMP library\n');
55 fprintf(shpfile, '##DATA TYPE= Shape Data\n');
56 fprintf(shpfile, '##ORIGIN= Dawei's GRAPE Pulses \n');
57 fprintf(shpfile, '##OWNER= Dawei\n');
58 fprintf(shpfile, '##DATE= %s\n',date);
59 time = clock;
60 fprintf(shpfile, '##TIME= %d:%d\n',fix(time(4)),fix(time(5)));
61 fprintf(shpfile, '##MINX= %7.6e\n',min(power_encoding1_H));
62 fprintf(shpfile, '##MAXX= %7.6e\n',max(power_encoding1_H));
63 fprintf(shpfile, '##MINY= %7.6e\n',min(phase_encoding1_H));
64 fprintf(shpfile, '##MAXY= %7.6e\n',max(phase_encoding1_H));
65 fprintf(shpfile, '##$SHAPE_EXMODE= None\n');
66 fprintf(shpfile, '##$SHAPE_TOTROT= %7.6e\n',90);
67 fprintf(shpfile, '##$SHAPE_BWFAC= %7.6e\n',1);
68 fprintf(shpfile, '##$SHAPE_INTEGFAC= %7.6e\n',1);
69 fprintf(shpfile, '##$SHAPE_MODE= 1\n');
70 fprintf(shpfile, '##PULSE_WIDTH= %d\n',total_time_encoding1);

```

```

69     fprintf(shpfile, '##Calibration_Power= %d\n', Calibration);
70     fprintf(shpfile, '##NPOINTS= %d\n', length(power_encoding1_H));
71     fprintf(shpfile, '##XYPPOINTS= (XY..XY)\n');
72
73     for ii = 1:length(power_encoding1_H)
74         fprintf(shpfile, '    %7.6e,    %7.6e\n', power_encoding1_H(ii), phase_encoding1_H(
75             ii));
76     end
77     fprintf(shpfile, '##END=\n');

```

The two output files are 'twqubit_encoding1.C' and 'twqubit_encoding1.H'. The calibrations are 25000Hz.
