Practical aspects of simulating MAS NMR

Guinevere Mathies

Emmy Noether-Programm
DFG Deutsche Forschungsgemeinschaft

Universität Konstanz

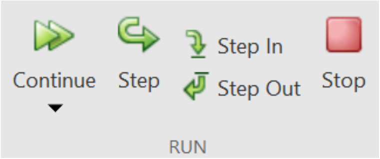
SFB 1527 HyPERiON

Introduction to Spinach

https://spindynamics.org

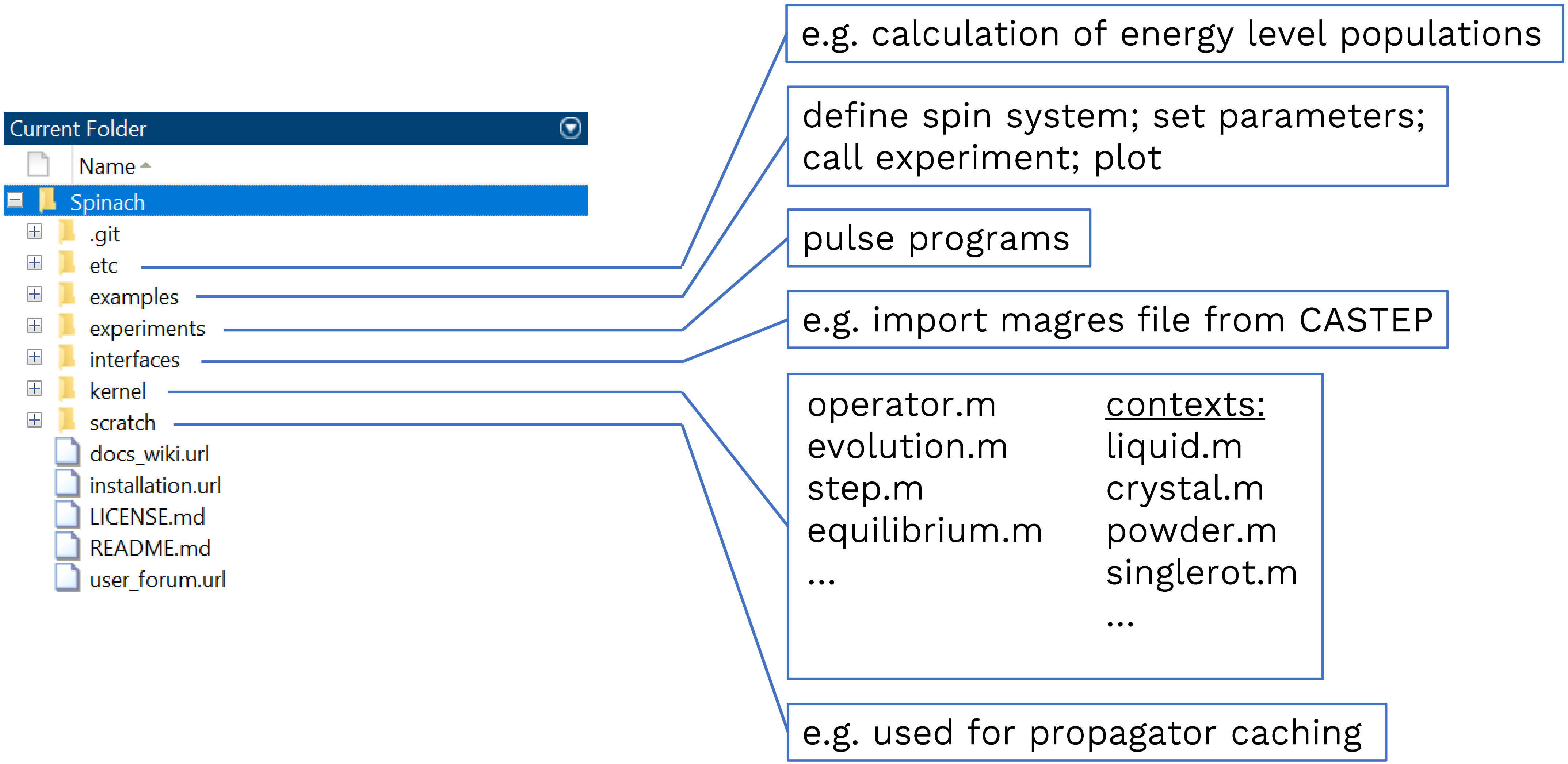
- open-source spin dynamics simulation library covering magnetic resonance in width and in depth
- developed and maintained by Ilya Kuprov (University of Southampton) and his team
- Hogben, ..., Kuprov *JMR* 2011
- for many magnetic resonance systems, reduced basis sets are an excellent approximation
- polynomial, as opposed to exponential, complexity scaling makes it possible to simulate the dynamics of large(r) spin systems

- version 2.9 (May 2024)
- developer version on GitHub (https://github.com/IlyaKuprov/Spinach)
- runs under Matlab (Mathworks)
- open source and platform independent
- installation: download Spinach and add to the Matlab path
- Spinach consists of functions; troubleshoot in debug mode



- documentation Wiki
- functions are annotated

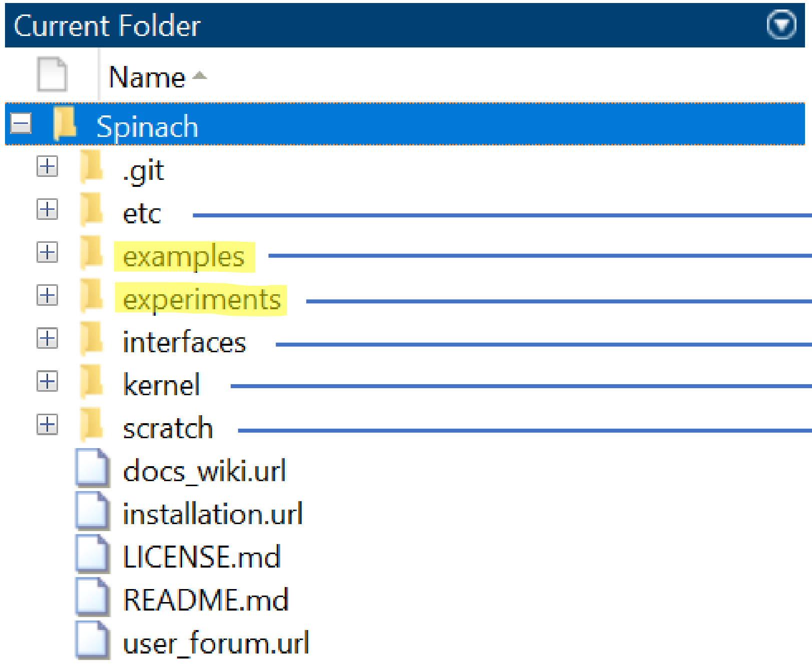
Overview



Current Folder
Name
Spinach
.git
etc
examples
experiments
interfaces
kernel
scratch
docs_wiki.url
installation.url
LICENSE.md
README.md
user_forum.url
e.g. calculation of energy level populations
define spin system; set parameters; call experiment; plot
pulse programs
e.g. import magres file from CASTEP
operator.m
evolution.m
step.m
equilibrium.m
...
contexts:
liquid.m
crystal.m
powder.m
singlerot.m
...
e.g. used for propagator caching

Overview

pulse programming for MAS NMR



e.g. calculation of energy level populations

define spin system; set parameters; call experiment; plot

pulse programs

e.g. import magres file from CASTEP

operator.m	<u>contexts:</u>
evolution.m	liquid.m
step.m	crystal.m
equilibrium.m	powder.m
…	singlerot.m
	…

e.g. used for propagator caching

Setting the magic angle with KBr

example

initialize the spin system

```
function setting_ma()

    % Specify the spin system
    sys.isotopes={'79Br'};
    inter.coupling.matrix{1,1}=eeqq2nqi(-92.4e3,-0.79,3/2,[0 0 0]);
    inter.zeeman.scalar=60.0933;

    % Magnetic field
    sys.magnet=9.4;

    % Basis set
    bas.formalism='sphten-liouv';
    bas.approximation='none';

    % Spinach housekeeping
    spin_system=create(sys,inter);
    spin_system=basis(spin_system,bas);

    %...%

end
```

convert C_q and η_q quadrupolar interaction parameters into interaction matrix

main component; determined from fit to experimental KBr spectrum

Liouville space; fundamental operators are single-spin irreducible spherical tensors

creates the spin system object that the library requires to run

update spin-system with basis set and related information

Setting the magic angle with KBr

set up the experimental parameters

```
function setting_ma()

    %...%

    % Experiment setup
    parameters.spins={'79Br'};
    parameters.rate=4000; % spinning freq in Hz
    parameters.max_rank=21; % maximum harmonic rank to retain
    parameters.grid='rep_2ang_400pts_sph'; % choice of powder grid
    parameters.sweep=1e5; % in Hz, dw = 10 us
    parameters.npoints=2048; % acquisition time = 20.48 ms
    basefrq=-spin(parameters.spins{1})*spin_system.inter.magnet/(2*pi);
    parameters.offset=-60.0933*basefrq/1e6; % in Hz
    parameters.zerofill=4096;
    parameters.offset=0; % carrier freq offset in Hz
    parameters.rho0=state(spin_system,'L+','79Br'); % start in x,y-plane
    parameters.coil=state(spin_system,'L+','79Br'); % note L+=Lx+iLy

    %...%

end
```

Setting the magic angle with KBr

slightly change the spinning axis and obtain the FIDs

```
function setting_ma()

    %...%

    for m=1:numel(z_axis)
        parameters.axis=[1 1 z_axis(m)];


        % Simulation
        fid=singlerot(spin_system,@acquire,parameters,'nmr');

        % Apodization and Fourier transform
        fid=apodization(fid,'exp-1d',6);
        spectrum=fftshift(fft(fid,parameters.zerofill));

        %...plotting...%

    end

end
```

array of z-heights; to vary the angle

call the pulse sequence via the context using a function handle

rotating frame, secular terms

possibility to adjust to experimental line width

Setting the magic angle with KBr

experiment

what goes on inside acquire.m

```
function fid=acquire(spin_system,parameters,H,R,K)

    %...%

    % Run the evolution and watch the coil state
    fid=evolution(spin_system,L,parameters.coil,parameters.rho0,
    1/parameters.sweep,parameters.npoints-1,'observable');

    %...%

end
```

Setting the magic angle with KBr

slightly change the spinning axis and obtain the FID

```
function setting_ma()

    %...%

    for m=1:numel(z_axis)
        parameters.axis=[1 1 z_axis(m)];


        % Simulation
        fid=singlerot(spin_system,@acquire,parameters,'nmr');

        % Apodization and Fourier transform
        fid=apodization(fid,'exp-1d',6);
        spectrum=fftshift(fft(fid,parameters.zerofill));

        %...plotting...%

    end

end
```

array of z-heights; to vary the angle

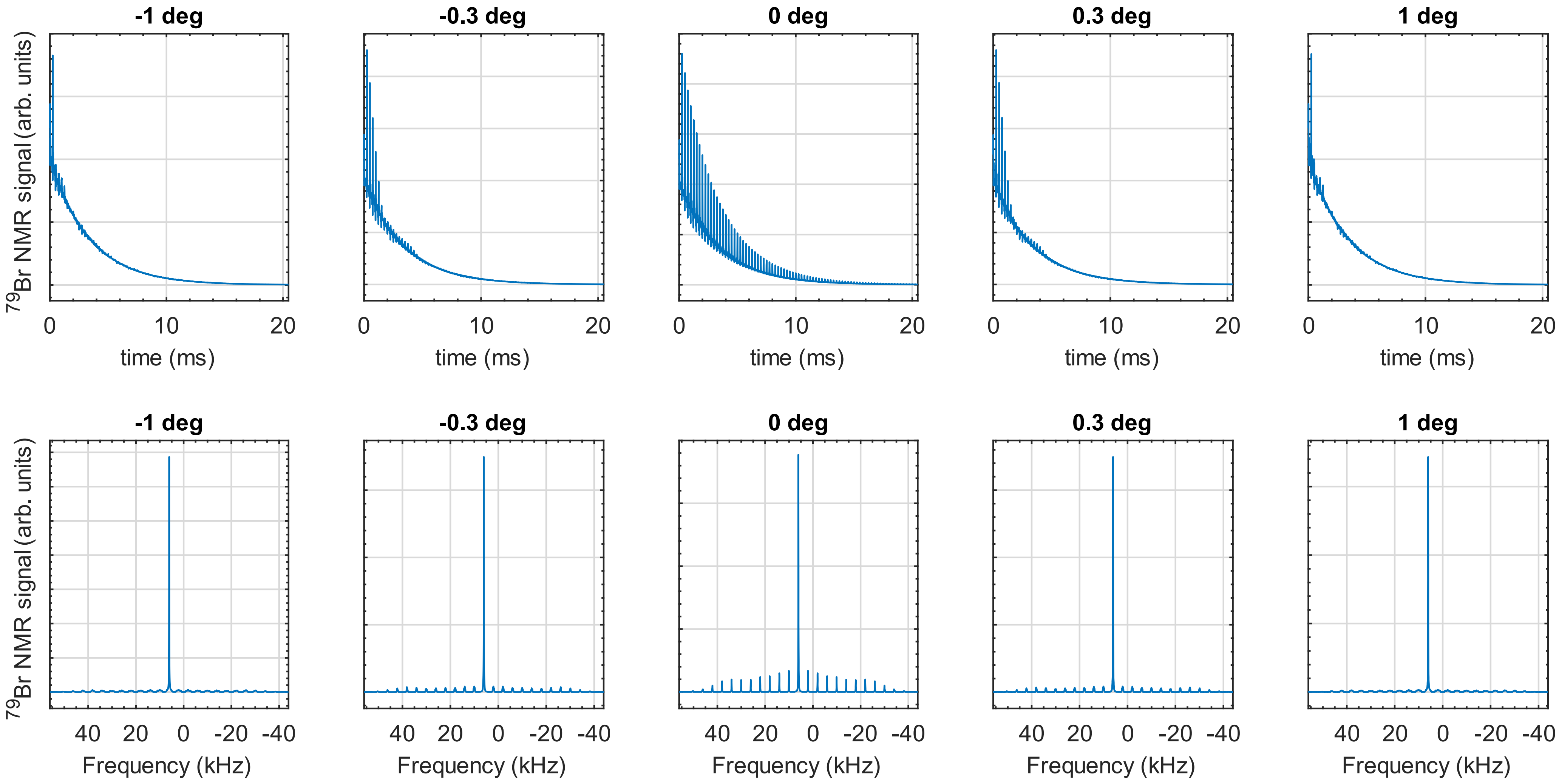
call the pulse sequence via the context using a function handle

rotating frame, secular terms

possibility to adjust to experimental line width

Setting the magic angle with KBr

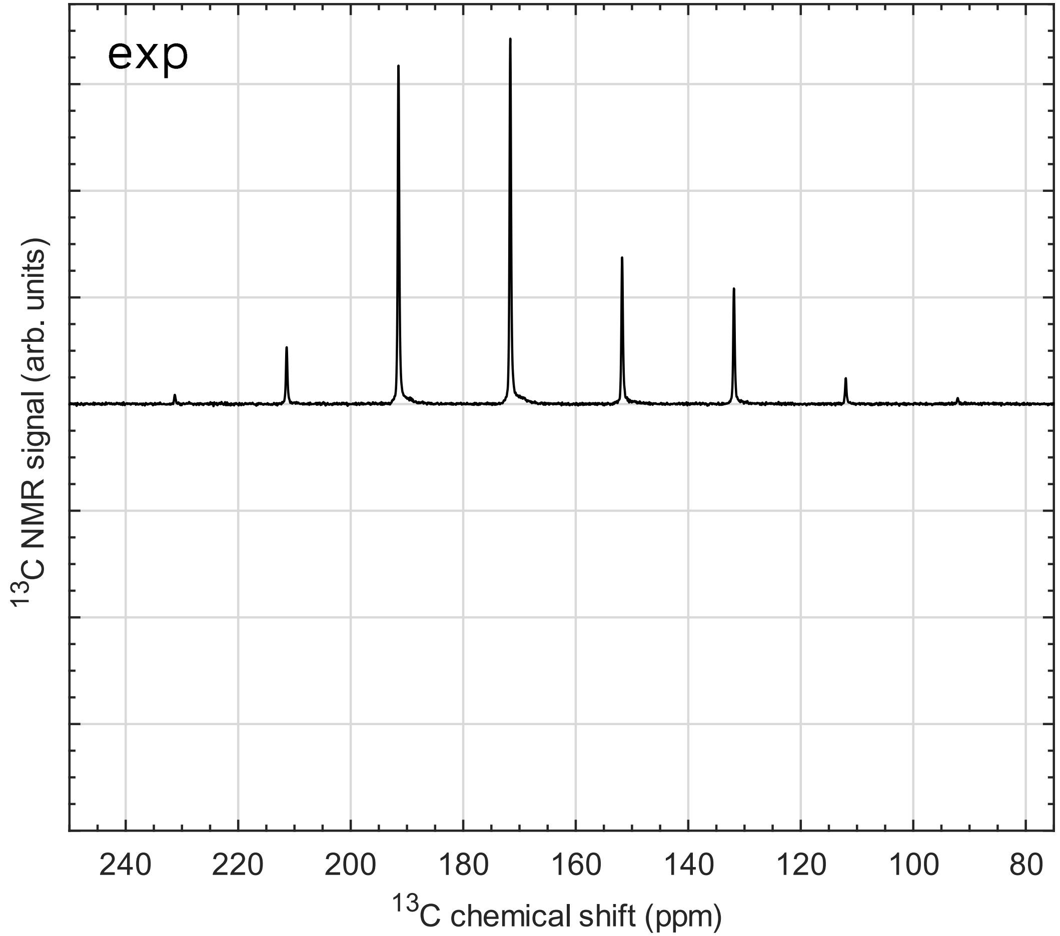
simulated FIDs and spectra



^{13}C chemical shift anisotropy

spin slowly and analyze the side bands

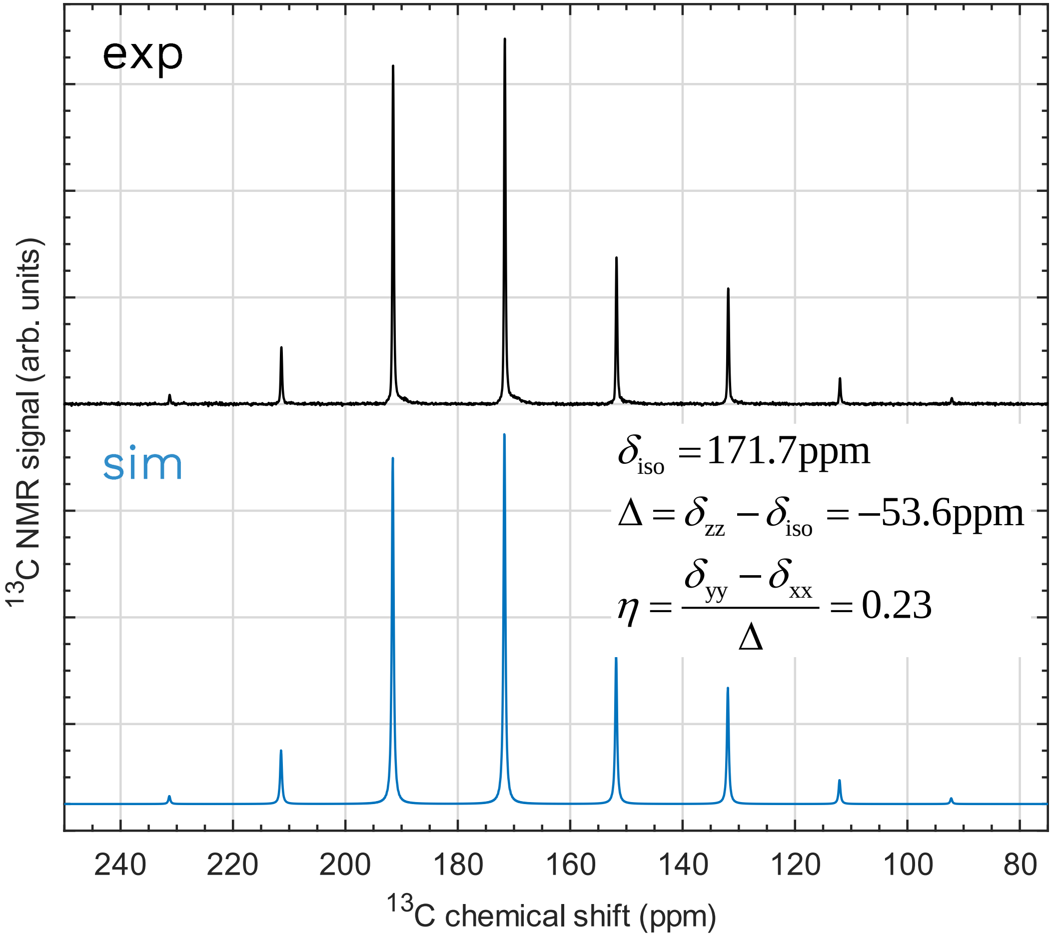
^{13}C-carbonate in monohydrocalcite ($CaCO_3 \cdot H_2O$), 400 MHz, 2 kHz spinning



^{13}C chemical shift anisotropy

reproduce the sideband pattern with a simulation

^{13}C-carbonate in monohydrocalcite ($CaCO_3 \cdot H_2O$), 400 MHz, 2 kHz spinning



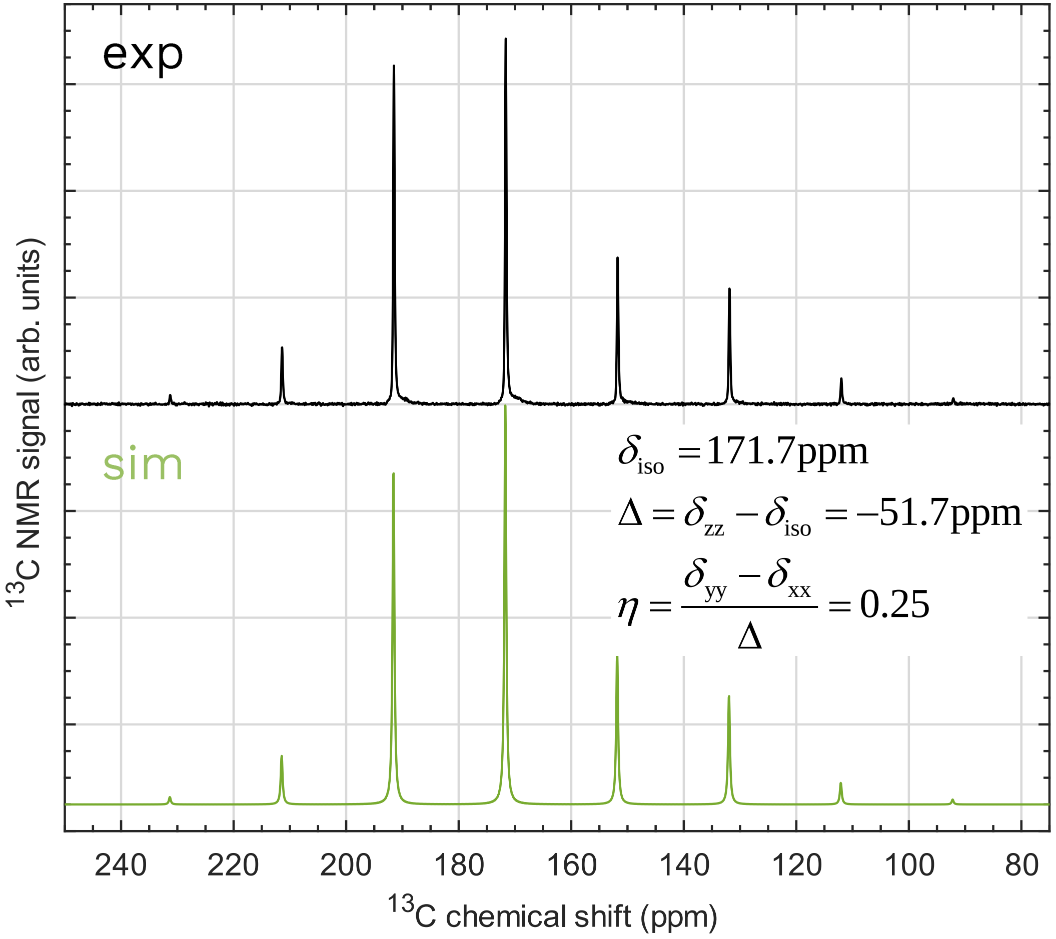

```
% Specify the spin system
delta_iso = 171.7; Delta = -53.6; eta = 0.23;

delta_zz = delta_iso+Delta;
delta_yy = (eta*Delta+3*delta_iso-delta_zz)/2;
delta_xx = 3*delta_iso-delta_zz-delta_yy;

inter.zeeman.eigs={[delta_xx delta_yy delta_zz]};
inter.zeeman.euler={[0 0 0]};

%...%

% Simulation
fid=singlerot(spin_system,@acquire,parameters,'nmr');
```

Vinod-Kumar et al. *In preparation*

^{13}C chemical shift anisotropy

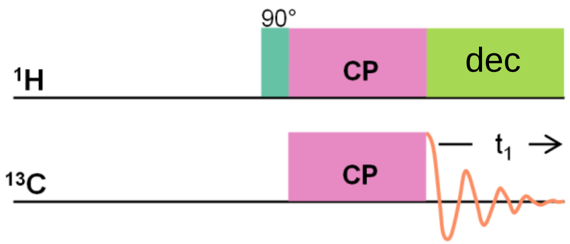
import calculated magnetic properties into Spinach

^{13}C-carbonate in monohydrocalcite ($CaCO_3 \cdot H_2O$), 400 MHz, 2 kHz spinning




```
% Read CASTEP file
props=c2spinach('mhc.magres');

sys.isotopes{1}='13C';

% Convert shielding tensors into shift using
% parametrisation of Huang et al. ACIE 2021
inter.zeeman.matrix{1}=169.86*eye(3)-props.cst{19};

% Cartesian coordinates
inter.coordinates={props.std_geom(19,:)};

%...%

% Simulation
fid=singlerot(spin_system,@acquire,parameters,'nmr');
```

^1H-^{13}C cross polarization



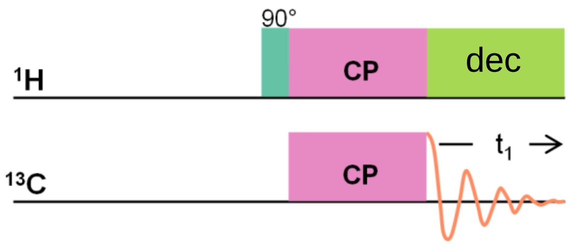
use cp_acquire_soft.m (instead of acquire.m)

```
function fid=cp_acquire_soft(spin_system,parameters,H,R,K)

    %...%

    % Build and project 1H and 13C control operators
    Hx=operator(spin_system,'Lx',parameters.spins{1});
    Hy=operator(spin_system,'Ly',parameters.spins{1});
    Cx=operator(spin_system,'Lx',parameters.spins{2});
    Hx=kron(speye(parameters.spc_dim),Hx);
    Hy=kron(speye(parameters.spc_dim),Hy);
    Cx=kron(speye(parameters.spc_dim),Cx);

    % Apply the 90-degree pulse on 1H along +X
    rho=step(spin_system,L+2*pi*parameters.hi_pwr*Hx,...
                         rho,1/(4*parameters.hi_pwr));

    %...%

end
```

include spatial degrees of freedom for Fokker-Planck formalism

alternative algorithm for propagation with short, one-off pulses

^1H-^{13}C cross polarization



use cp_acquire_soft.m (instead of acquire.m)

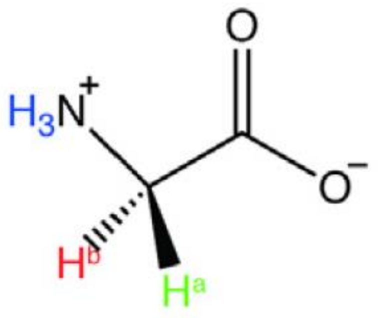
```
function fid=cp_acquire_soft(spin_system,parameters,H,R,K)

    %...%

    % Run the CP contact time evolution: irradiation
    % of 1H along -Y, and of 13C along +X
    rho=evolution(spin_system,L-2*pi*parameters.cp_pwr(1)*Hy...
                               +2*pi*parameters.cp_pwr(2)*Cx,...
                  [],rho,parameters.cp_dur,1,'final');

    % Wipe the state of 1H and apply 1H decoupling
    [L,rho]=decouple(spin_system,L,rho,parameters.spins(1));

    % Run the acquisition
    fid=evolution(spin_system,L,parameters.coil,rho,...
                  1/parameters.sweep,parameters.npoints-1,'observable')

end
```

^{1}H-^{13}C cross polarization

set up the spin system for alpha-Gly and run

contains calculated magnetic properties of example molecules incl. all amino acids

```
function cp_acquire_mas_gly()

    % Spin system properties (PCM DFT calculation)
    [sys,inter]=g2spinach(gparse('..\..\examples\standard_systems\glycine.log'),...
                          {{'H','1H'},{'C','13C'}},[31.8 182.1],[]);

    % Isotropic alpha-glycine chemical shifts
    inter.zeeman.matrix=shift_iso(inter.zeeman.matrix,1,176.4); % CO
    inter.zeeman.matrix=shift_iso(inter.zeeman.matrix,2, 43.6); % CA
    inter.zeeman.matrix=shift_iso(inter.zeeman.matrix,3,  2.6); % H_CA
    inter.zeeman.matrix=shift_iso(inter.zeeman.matrix,4,  3.8); % H_CA
    inter.zeeman.matrix=shift_iso(inter.zeeman.matrix,5,  8.0); % H_N
    inter.zeeman.matrix=shift_iso(inter.zeeman.matrix,6,  8.0); % H_N
    inter.zeeman.matrix=shift_iso(inter.zeeman.matrix,7,  8.0); % H_N

    %...%

end
```



Glycine

^{1}H-^{13}C cross polarization

set up the spin system for alpha-Gly and run

```
function cp_acquire_mas_gly()

    %...%

    % Experiment setup
    parameters.spins={'1H','13C'};
    parameters.rate=10000;
    parameters.axis=[1 1 1];
    parameters.max_rank=5;
    parameters.grid='rep_2ang_100pts_sph';
    parameters.offset=[2e3 10e3];  % Hz
    parameters.hi_pwr=83e3;        % Hz
    parameters.cp_pwr=[60e3 50e3]; % Hz
    parameters.cp_dur=50e-5;       % seconds
    %...%

end
```

^{1}H-^{13}C cross polarization

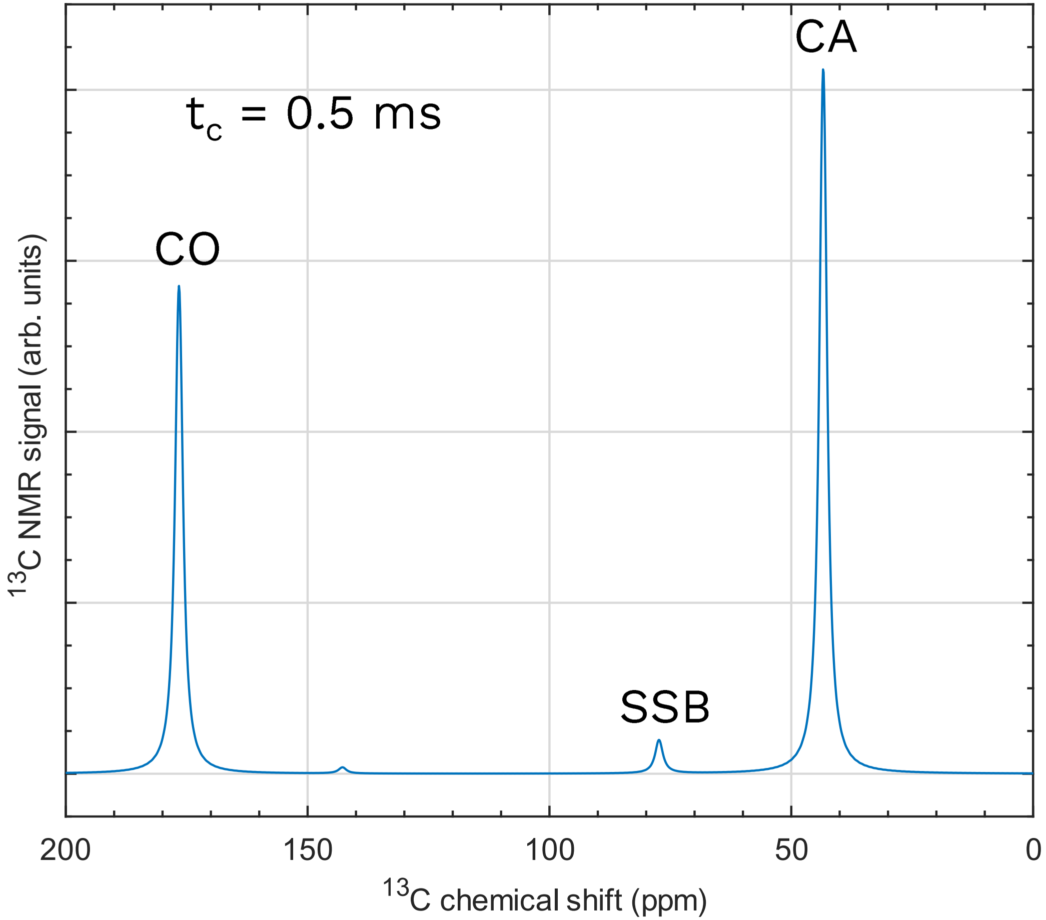
set up the spin system for alpha-Gly and run

```
function cp_acquire_mas_gly()

    %...%
    parameters.sweep=5e4;           % Hz, dw = 20 us
    parameters.npoints=512;
    parameters.zerofill=4096;
    parameters.needs={'iso_eq'};    % initial condition is thermal equilibrium

    % Detection state
    parameters.coil=state(spin_system,'L+','13C');

    % Simulation
    fid=singlerot(spin_system,@cp_acquire_soft,parameters,'nmr');

end
```

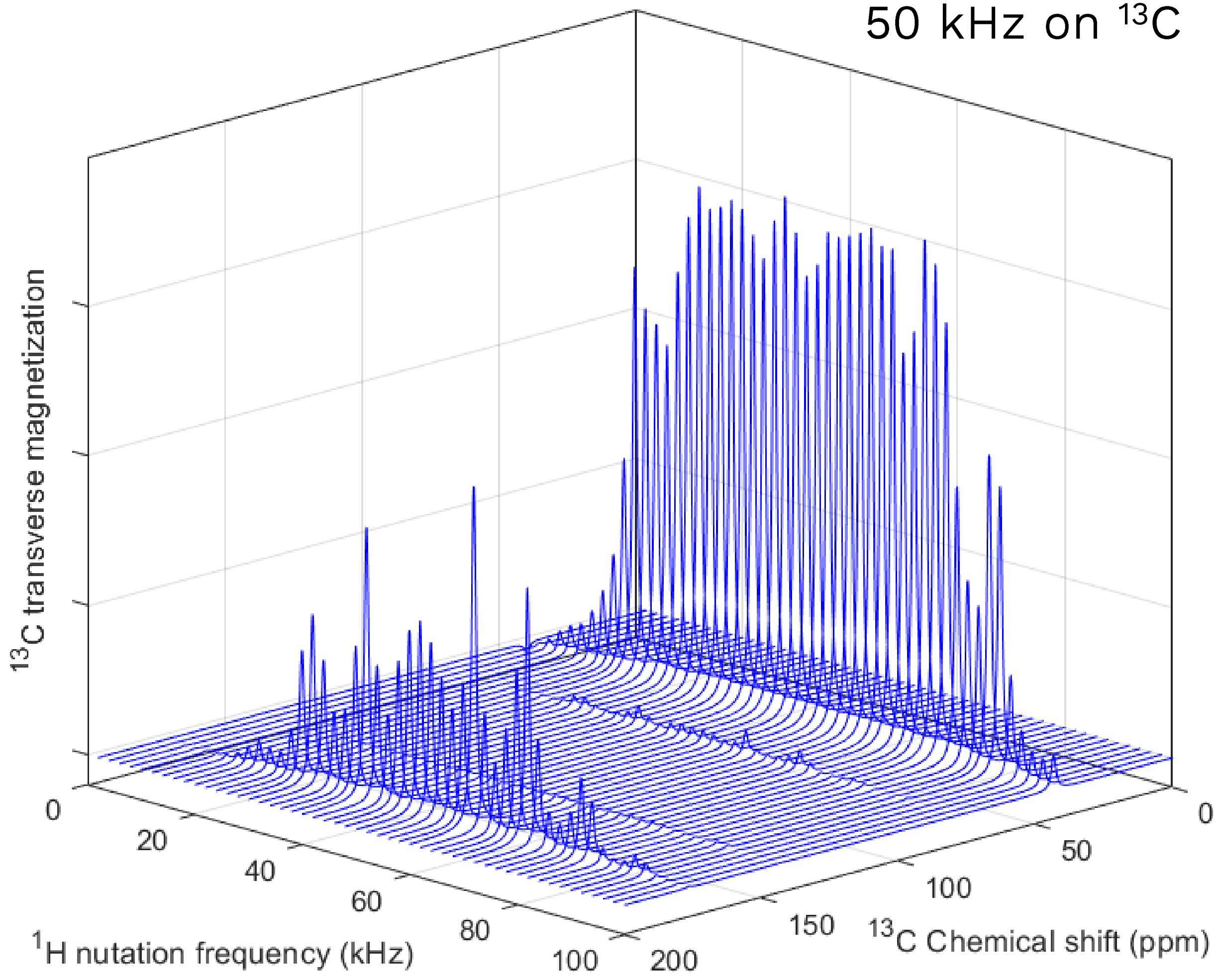
^1H-^{13}C cross polarization

optimize the CP conditions

alpha-Gly, 400 MHz, 10 kHz spinning

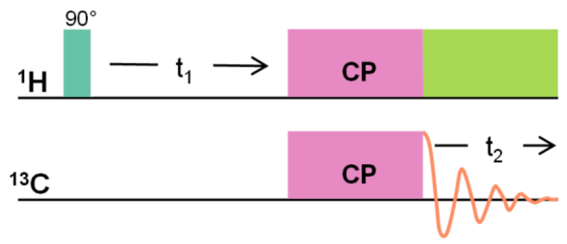


50 kHz on ^{13}C



^{1}H-^{13}C cross correlation

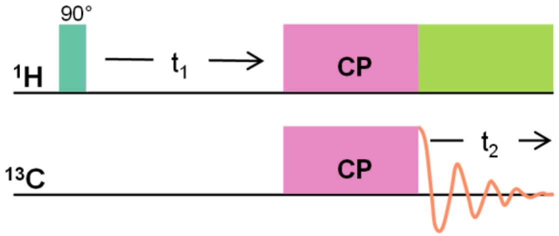
wide-line separation (WISE): wise.m




```
function fid=wise(spin_system,parameters,H,R,K)

    %...%

    % Build and project 1H and 13C control operators
    Hx=operator(spin_system,'Lx',parameters.spins{1});
    Hy=operator(spin_system,'Ly',parameters.spins{1});
    Cx=operator(spin_system,'Lx',parameters.spins{2});
    Hx=kron(speye(parameters.spc_dim),Hx);
    Hy=kron(speye(parameters.spc_dim),Hy);
    Cx=kron(speye(parameters.spc_dim),Cx);

    % High-power 90-degree pulses on 1H along X (cos) and Y (sin)
    L_hp_cos=L+2*pi*parameters.hi_pwr*Hx; L_hp_sin=L+2*pi*parameters.hi_pwr*Hy;
    rho_cos=step(spin_system,L_hp_cos,parameters.rho0,1/(4*parameters.hi_pwr));
    rho_sin=step(spin_system,L_hp_sin,parameters.rho0,1/(4*parameters.hi_pwr));

    %...%

end
```

^{1}H-^{13}C cross correlation

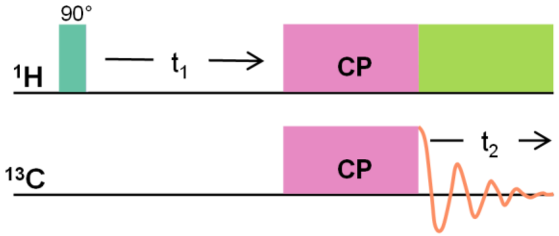
wide-line separation (WISE): wise.m




```
function fid=wise(spin_system,parameters,H,R,K)

    %...%

    % Get dwell times
    dw=1./parameters.sweep;

    % Run the F1 evolution
    rho_stack_cos=evolution(spin_system,L,[],rho_cos,dw(1),parameters.npoints(1)-1,'trajectory');
    rho_stack_sin=evolution(spin_system,L,[],rho_sin,dw(1),parameters.npoints(1)-1,'trajectory');

    % CP contact time evolution generator (-Y on 1H, +X on 13C)
    L_cp=L-2*pi*parameters.cp_pwr(1)*Hy+2*pi*parameters.cp_pwr(2)*Cx;

    % Run CP contact time evolution
    rho_stack_cos=evolution(spin_system,L_cp,[],rho_stack_cos,parameters.cp_dur,1,'final');
    rho_stack_sin=evolution(spin_system,L_cp,[],rho_stack_sin,parameters.cp_dur,1,'final');

    %...%

end
```

^1H-^{13}C cross correlation

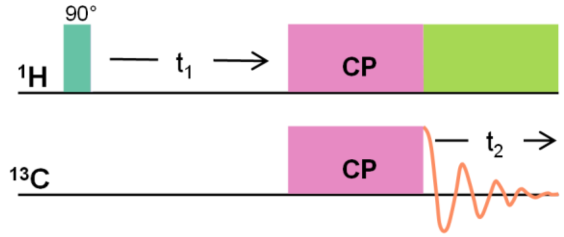
wide-line separation (WISE): wise.m




```
function fid=wise(spin_system,parameters,H,R,K)

    %...%

    % Wipe and decouple protons for acquisition
    [L_dec,rho_stack_cos]=decouple(spin_system,L,rho_stack_cos,parameters.spins(1));
    [~,rho_stack_sin]=decouple(spin_system,[],rho_stack_sin,parameters.spins(1));

    % Run the F2 evolution
    fid.cos=evolution(spin_system,L_dec,parameters.coil,rho_stack_cos,...
                      dw(2),parameters.npoints(2)-1,'observable');
    fid.sin=evolution(spin_system,L_dec,parameters.coil,rho_stack_sin,...
                      dw(2),parameters.npoints(2)-1,'observable');

end
```

^{1}H-^{13}C cross correlation

wide-line separation (WISE)




```
function wise_mas_gly()

    %...%

    % Basis set
    bas.formalism='sphten-liouv';
    bas.approximation='IK-0';
    bas.level=4;

    % Ignore interactions below 200 Hz
    sys.tols.inter_cutoff=2*pi*200;

    %...%

    % Simulation
    fid=singlerot(spin_system,@wise,parameters,'nmr');

    %...%

end
```

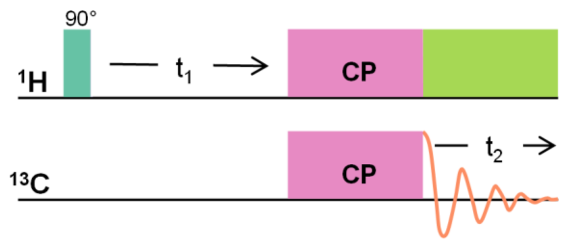
include all product states between up to (and including) bas.level spins located anywhere within the system

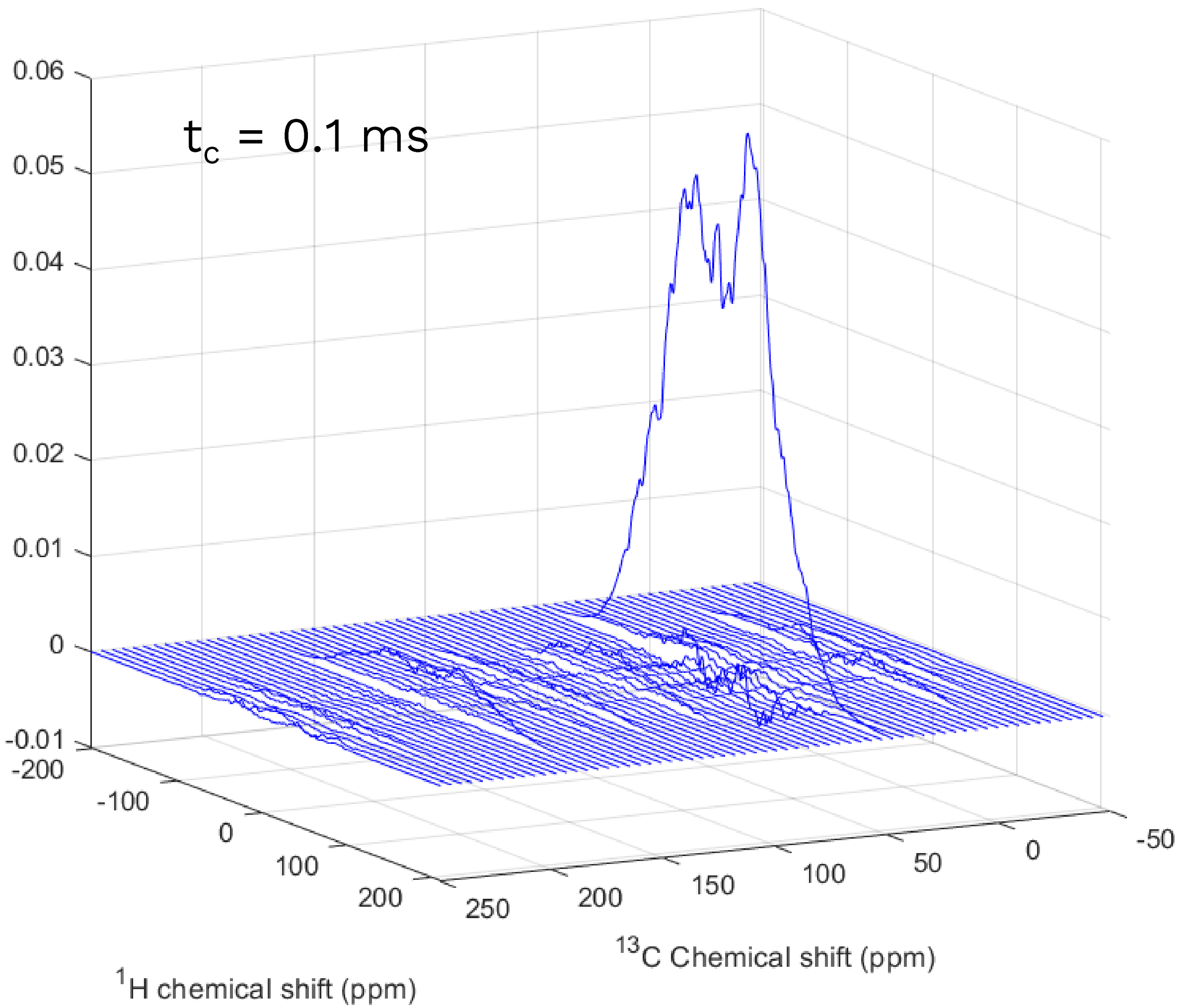
discard product states between spins that have weaker coupling

... 6 hours on the server...

^{1}H-^{13}C cross correlation

wide-line separation (WISE)

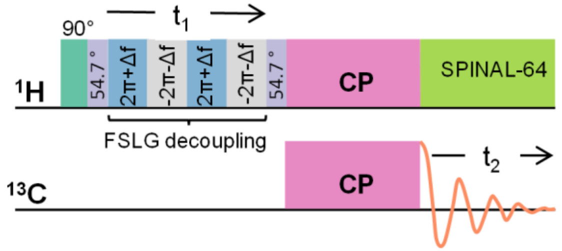




alpha-Gly, 400 MHz, 5 kHz spinning

^1H-^{13}C cross correlation

include frequency-switched Lee-Goldburg ^1H-^1H decoupling: fslghetcor.m



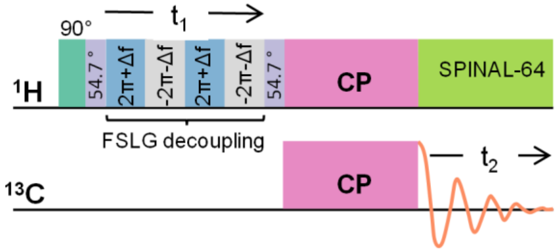

```
function fid=fslghetcor(spin_system,parameters,H,R,K)

    %...%

    % FSLG pulse evolution generators
    L_FSLG_p=L-2*pi*parameters.offset(1)*Hz ...
              +2*pi*(parameters.hi_pwr/sqrt(2))*Hz;
    L_FSLG_m=L-2*pi*parameters.offset(1)*Hz ...
              -2*pi*(parameters.hi_pwr/sqrt(2))*Hz;
    L1_cos=L_FSLG_m+2*pi*parameters.hi_pwr*Hy;
    L2_cos=L_FSLG_p-2*pi*parameters.hi_pwr*Hy;
    L1_sin=L_FSLG_p+2*pi*parameters.hi_pwr*Hx;
    L2_sin=L_FSLG_m-2*pi*parameters.hi_pwr*Hx;

    %...%

end
```

carrier at 0 ppm offset during F1 evolution so the artifact does not interfere with the spectrum

set resonance offset for FSLG decoupling

^1H-^{13}C cross correlation

include frequency-switched Lee-Goldburg ^1H-^1H decoupling: fslghetcor.m

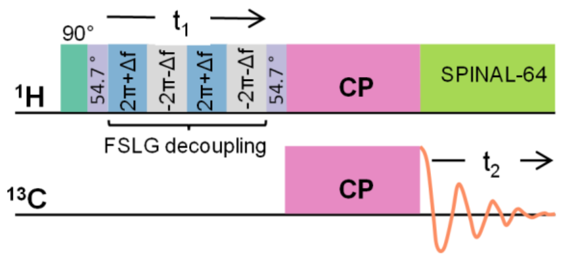



```
function fid=fslghetcor(spin_system,parameters,H,R,K)

    %...%

    for k=1:parameters.npoints(1) % Run the bulk of the sequence

        if k==1
            rho_cos_ev=rho_cos; rho_sin_ev=rho_sin; % First F1 step gets rho from above
        else
            for n=1:parameters.nblocks % Subsequent F1 steps keep going
                rho_cos_ev=step(spin_system,L1_cos,rho_cos_ev,sqrt(2/3)/parameters.hi_pwr);
                rho_cos_ev=step(spin_system,L2_cos,rho_cos_ev,sqrt(2/3)/parameters.hi_pwr);
                rho_sin_ev=step(spin_system,L1_sin,rho_sin_ev,sqrt(2/3)/parameters.hi_pwr);
                rho_sin_ev=step(spin_system,L2_sin,rho_sin_ev,sqrt(2/3)/parameters.hi_pwr);
            end
        end

        %...continue in for loop: propagate each increment in F1 to the end of the sequence...%

end
```

^{1}H-^{13}C cross correlation

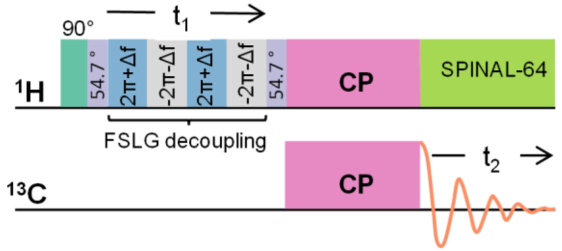
include frequency-switched Lee-Goldburg ^{1}H-^{1}H decoupling: fslghetcor.m

```
function fid=fslghetcor(spin_system,parameters,H,R,K)

       %...continue in for loop...%

       % Grab the current F1 point and divert it to F2
       rho_cos_cont=rho_cos_ev; rho_sin_cont=rho_sin_ev;

       % Flip the 1H back from the magic angle to the x,y-plane
       rho_cos_cont=step(spin_system,L_backtoXY_cos,rho_cos_cont,...
                         acos(1/sqrt(3))/(2*pi)/parameters.hi_pwr);
       rho_sin_cont=step(spin_system,L_backtoXY_sin,rho_sin_cont,...
                         acos(1/sqrt(3))/(2*pi)/parameters.hi_pwr);

       % Save work by stacking cos and sin parts
       rho_cont=[rho_cos_cont rho_sin_cont];

       %...continue in for loop...%

end
```



^{1}H-^{13}C cross correlation

include frequency-switched Lee-Goldburg ^{1}H-^{1}H decoupling: fslghetcor.m

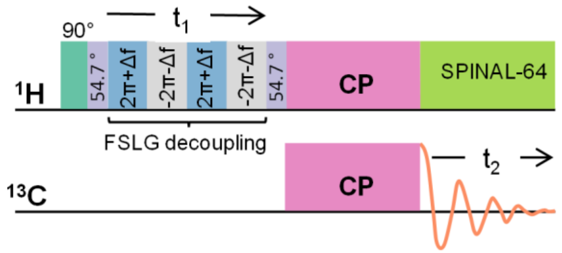



```
function fid=fslghetcor(spin_system,parameters,H,R,K)

        %...continue in for loop...%

        % Run the CP contact period
        rho_cont=step(spin_system,L_c,rho_cont,parameters.cp_dur);

        % Decouple 1H for the acquisition period
        [L_dec,rho_cont]=decouple(spin_system,L,rho_cont,parameters.spins(1));

        % Run the F2 evolution
        traj=evolution(spin_system,L_dec,parameters.coil,rho_cont,...
                       dwell_times(2),parameters.npoints(2)-1,'observable');
        fid.cos(:,k)=traj(:,1); fid.sin(:,k)=traj(:,2);

    end

end
```

^{1}H-^{13}C cross correlation

include frequency-switched Lee-Goldburg ^{1}H-^{1}H decoupling: fslghetcor.m

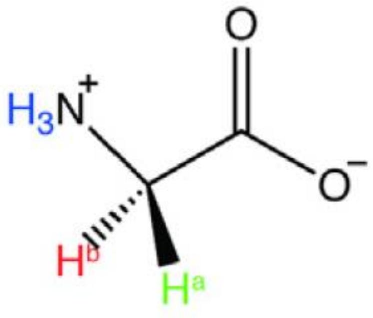
```
function fslghetcor_mas_gly()

    %...%

    % Basis set
    bas.formalism='sphten-liouv';
    bas.approximation='IK-0';
    bas.level=4;

    % Ignore interactions below 200 Hz
    sys.tols.inter_cutoff=2*pi*200;

    %...%

    % Simulation
    fid=singlerot(spin_system,@fslghetcor,parameters,'nmr');

    %...%

end
```



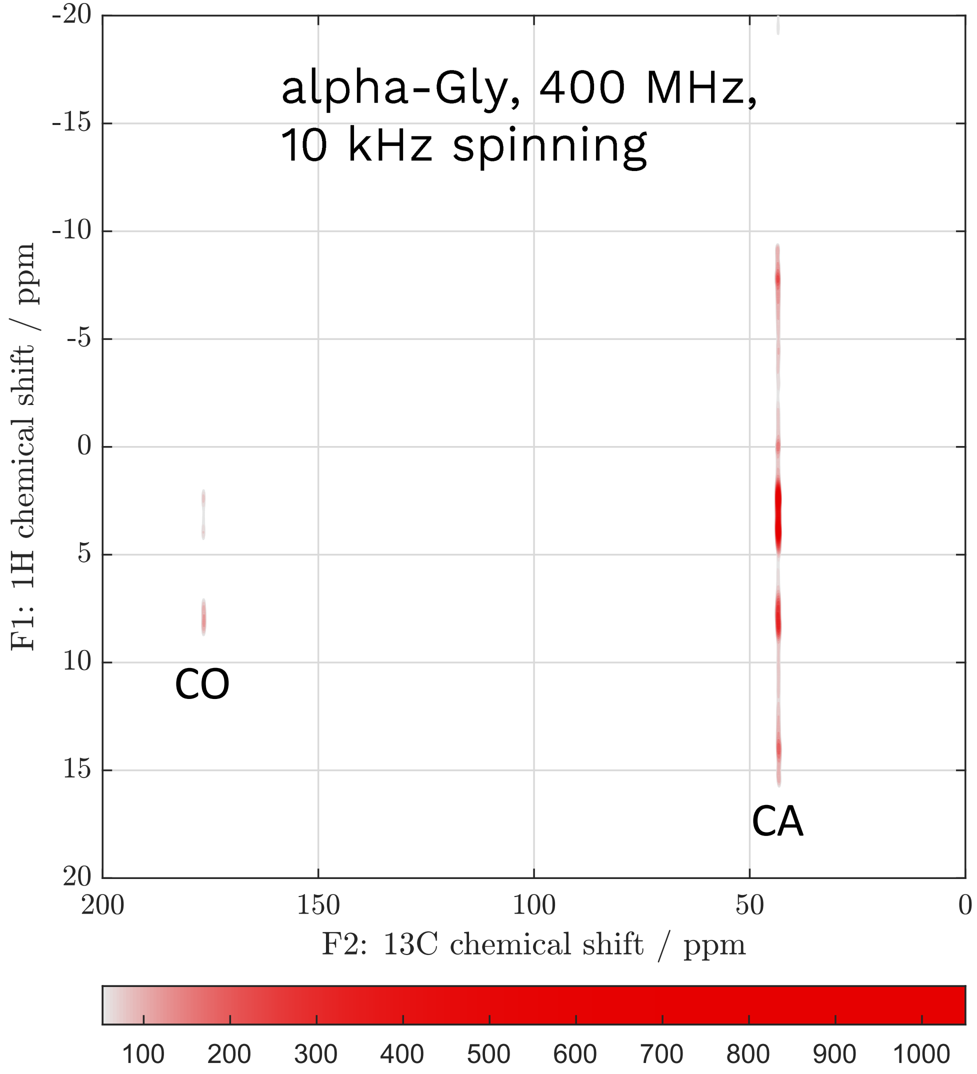
... 18 hours on the server...

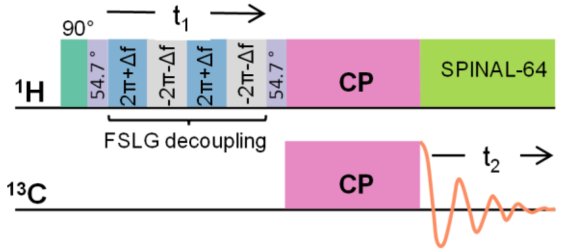
^{1}H-^{13}C cross correlation

include frequency-switched Lee-Goldburg ^{1}H-^{1}H decoupling: fslghetcor.m



Glycine

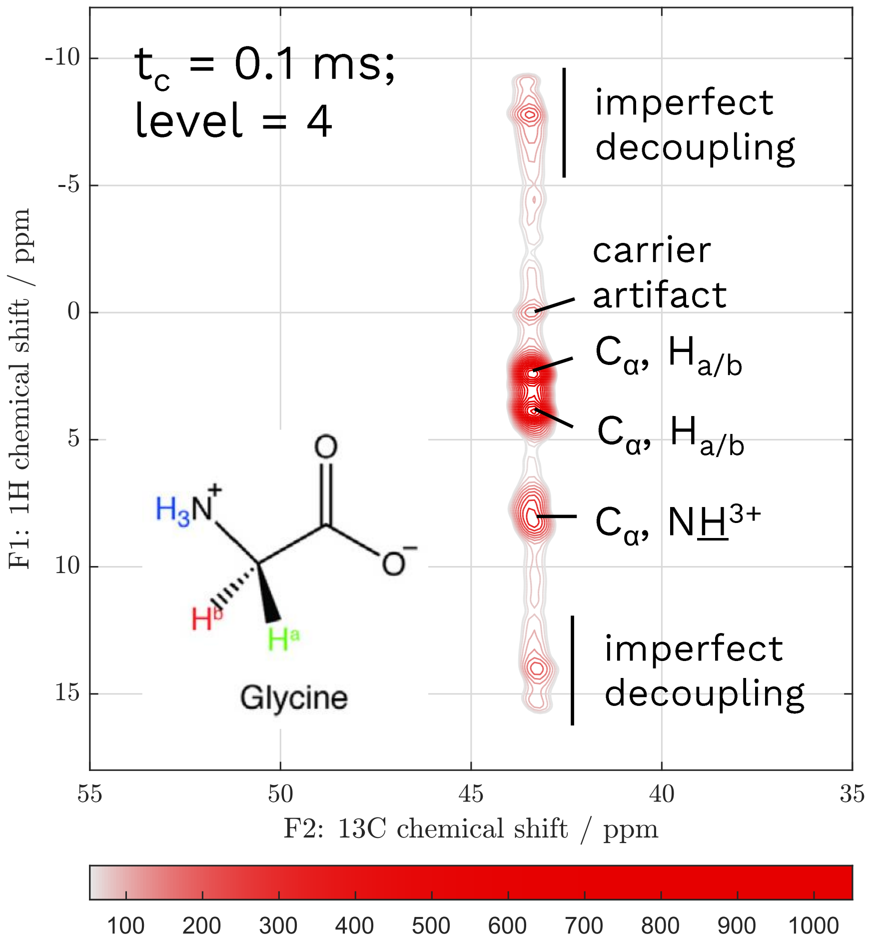


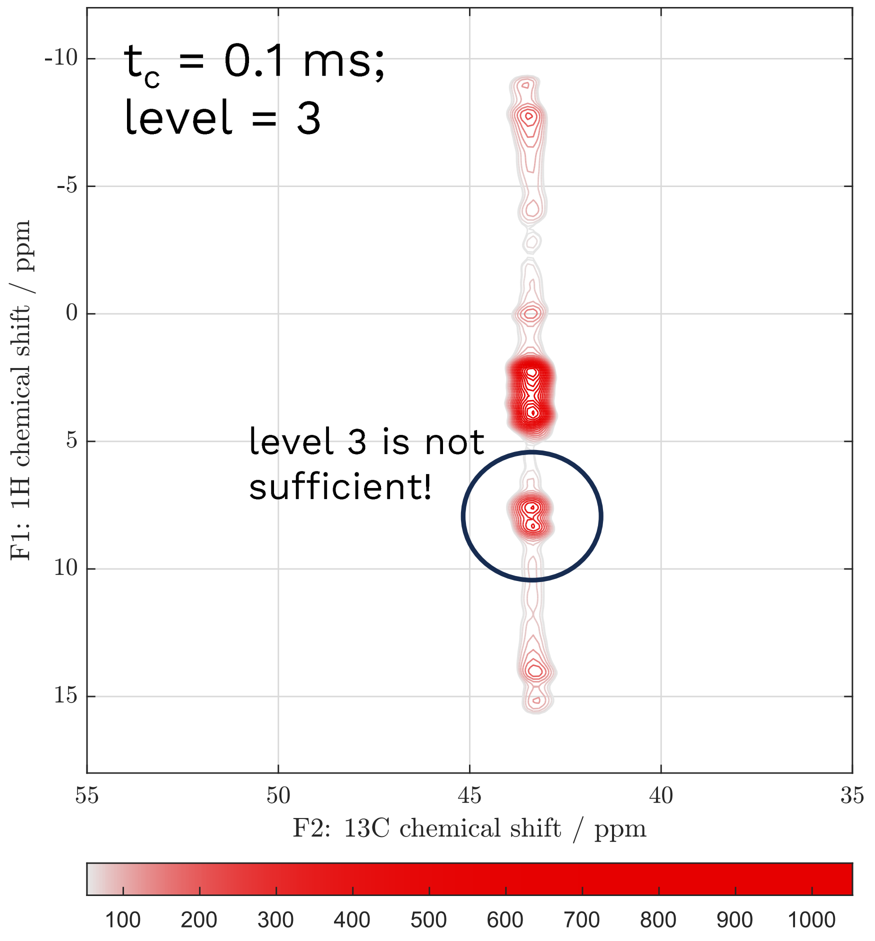


t_c = 0.1 ms;
level = 4

^{1}H-^{13}C cross correlation

include frequency-switched Lee-Goldburg ^{1}H-^{1}H decoupling: fslghetcor.m

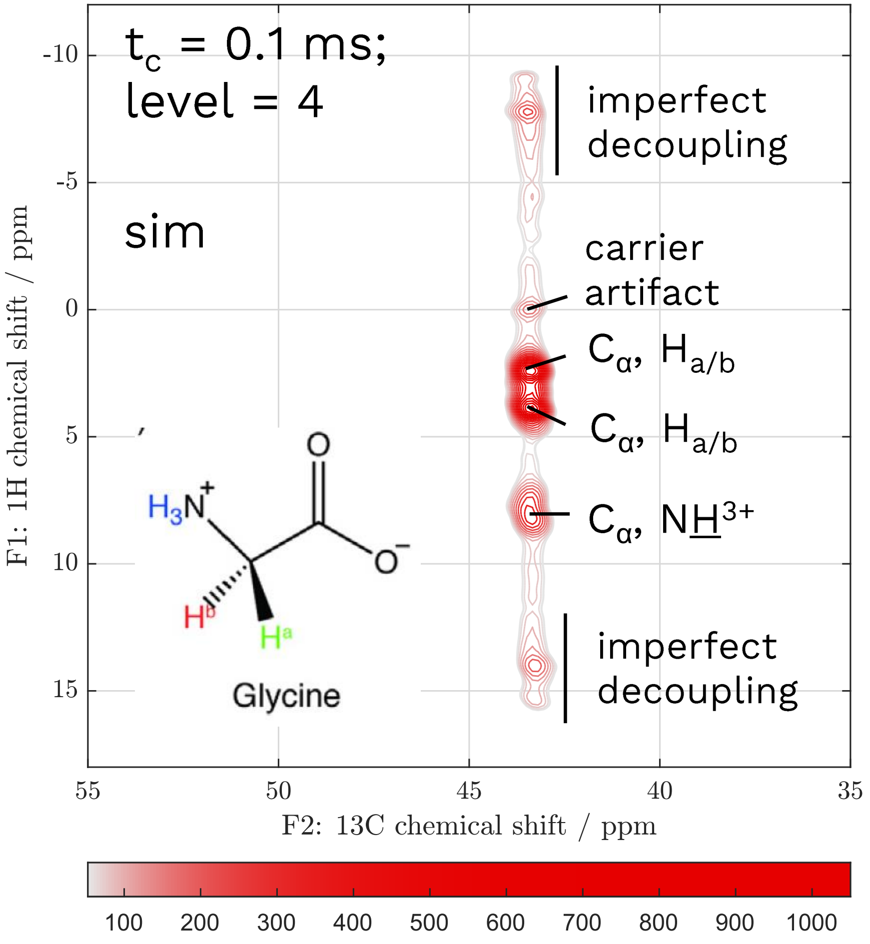


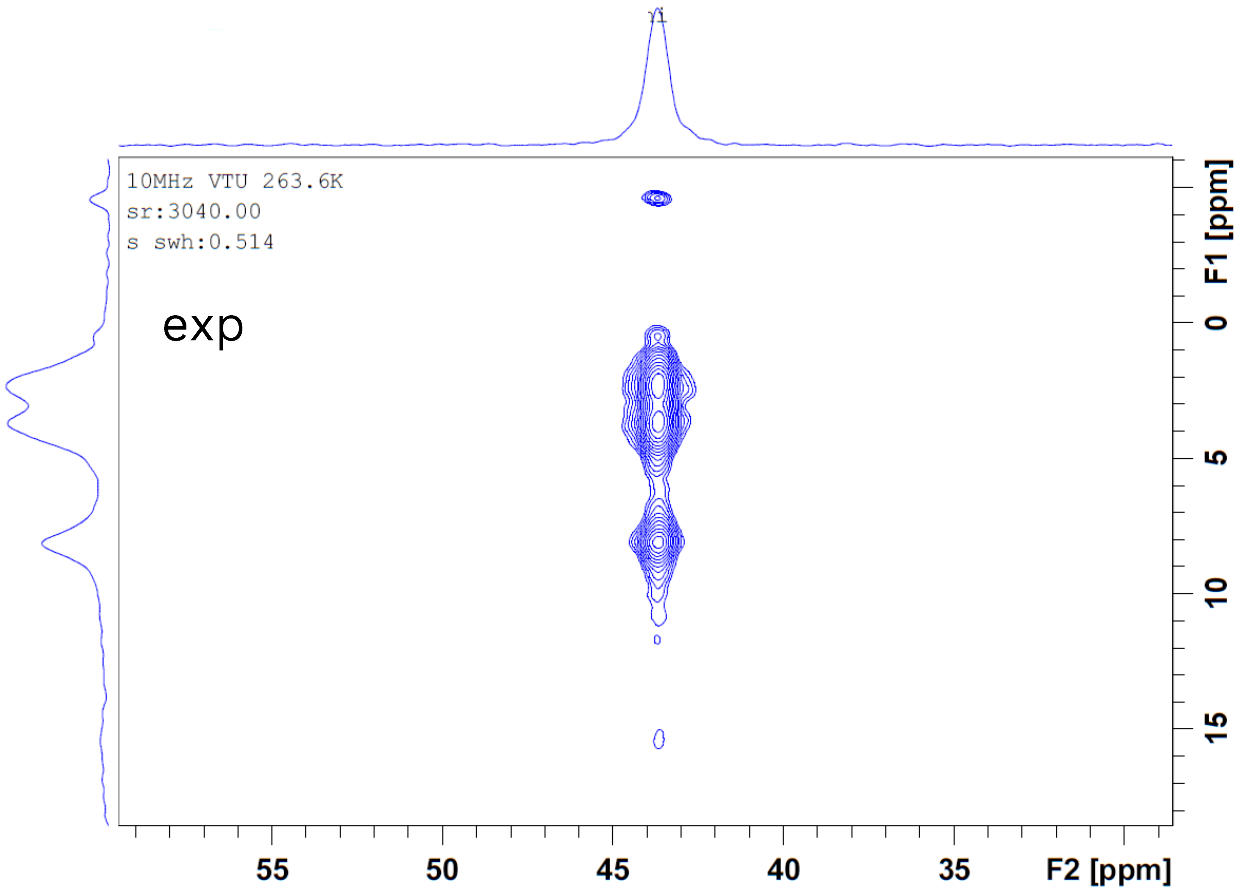


Vinod-Kumar et al. *In preparation*

^{1}H-^{13}C cross correlation

FSLG HETCOR of alpha-Gly: experiment





To summarize

Simulating MAS NMR with Spinach:

1. Specify spin system and interactions

2. Specify experimental parameters

in the example

3. Define operators for radio wave pulses of arbitrary phase (typically using operator.m)

4. Write the pulse sequence (carry out the propagations step-by-step, typically using evolution.m or step.m)

in the experiment

→ Add your favorite pulse sequence to the library!

Acknowledgement

Spinach

Ilya Kuprov
(University of Southampton)

Emmy Noether-Programm
DFG Deutsche Forschungsgemeinschaft

SFB 1527
HyPERiON

Universität Konstanz

Sanjay Vinod Kumar

R. Venkata SubbaRao
(now at NYU Abu Dhabi)