# MATLAB code for spin dynamics simulations

v0.1: Soumyajit Mandal, 11/12/2016

The folder contains a set of MATLAB functions for simulating the spin dynamics of NMR experiments. Various subfolders contain specialized versions of the functions for simulating specific scenarios, as described later. The key ideas used in the simulation are summarized below:

1. The spin dynamics is assumed to be that of an ensemble of uncoupled spin-1/2 nuclei, such as protons. This allows the dynamics to be represented by the motion of a “magnetization vector” in 3D space, but it also prevents J-coupling, multinuclear experiments, etc. from being modeled.
2. The simulation is based on discretizing the sample volume of interest into a large number of “isochromats”, which are defined as small regions within which the static and RF magnetic fields (*B*0 and *B*1, respectively), are assumed to be constant. The user should select the (*B*0, *B*1) distribution based on the system being studied. For example, a uniform *B*0 gradient and relatively constant *B*1 is a common assumption for inside-out sensor geometries. In this case *f*(*B*0, *B*1) reduces to a uniform distribution over *B*0.
3. Within each isochromat, spin dynamics are calculated by discretizing the pulse sequence into a series of time intervals of adjustable length. Each interval corresponds to a particular value of RF amplitude (*A*) and relative phase *φ* (i.e., phase in the rotating frame). This allows complex RF pulses to be modeled as a collection of (*A*, *φ*) pairs. In addition, *A* can be set to zero to model “free precession” intervals, i.e., gaps between pulses.
4. Each time interval results in a generalized rotation of the magnetization vector, i.e., rotation by angle θ about an axis. In general, both θ and  will differ between isochromats.
5. The default simulation functions do not model relaxation or diffusion. However, some functions do include these effects. Relaxation is modeled by decreasing the magnitude of the magnetization vector during each time interval. The user is advised to use the functions that include relaxation or diffusion only if necessary, since they are significantly slower than the default ones.
6. Output signals, typically spin echoes, are measured in the gaps between pulses. The user has to specify the length of the acquisition window, i.e., the duration over which the signal is measured.
7. By default, the functions assume normalized values of *B*1 such that. As a result, a nominal 180o pulse has a duration of π, and so on.
8. The “bare” spin dynamics simulator functions are named “sim\_spin\_dynamics\_X”, where X varies depending on the assumptions. These functions expect the user to input vectors of interval durations, (*A*, *φ*) pairs, and acquisition flags. The latter controls whether output signals are measured during each interval. The user is advised to read the documentation for each function (type “help [function\_name]”) and in general use the latest versions by checking when they were last modified.
9. A variety of helper functions are also available to construct common pulse sequences, such as the CPMG, in the format expected by the spin dynamics simulation functions. For example, the following commented code will simulate a basic CPMG experiment in a uniform *B*0 gradient:

% Define simulation parameters

numpts=2001; maxoffs=20; % Number of isochromats; range of *B*0 values (in units of *B*1)

del\_w=linspace(-maxoffs,maxoffs,numpts); % Create vector of isochromats

tacq=5\*pi; window = sinc(del\_w\*tacq/(2\*pi)); % Define window function for acquisition

window=window./sum(window);

% Create pulse sequence parameters

% 1) Excitation pulses – durations, phases, amplitudes

texc0=[pi/2 -1]; pexc0=[pi/2 0]; aexc0=[1 0];

% 2) Refocusing cycle (delay, pulse, delay) – durations, phases, amplitudes

tref0=pi\*[3 1 3]; pref0=[0 0 0]; aref0=[0 1 0];

% Run simulation to find the asymptotic signal, i.e., the signal that satisfies the CPMG condition and % is spin-locked with the effective rotation axis of the refocusing cycle

[neff0]=calc\_rot\_axis\_arba2(tref0,pref0,aref0,del\_w,0); % Find rotation axis of refocusing cycle

% Assume a two-part phase cycle; the phase of the excitation pulse varies by π between the two parts

[masy0\_1]=cpmg\_van\_spin\_dynamics\_asymp\_mag3(texc0,pexc0,aexc0,neff0,del\_w,tacq);

[masy0\_2]=cpmg\_van\_spin\_dynamics\_asymp\_mag3(texc0,pexc0+pi,aexc0,neff0,del\_w,tacq);

masy0=(masy0\_1-masy0\_2)/2; % Calculate magnetization vector after phase-cycling

snr0=trapz(del\_w,abs(masy0).^2); % Calculate RMS signal amplitude

[echo0,tvect]=calc\_time\_domain\_echo(masy0,del\_w); % Calculate time-domain echo signal