NMR School Windischleuba 26.3-1.4.2023

Problem Set 2: Hilbert-Space Simulations

In this Problem Set, we will look at very simple numerical simulations of magnetic resonance experiments. We will focus on understanding, and not too much on computational efficiency. For the latter, there are many software libraries and packages available, e.g. *Simpson*[1] or *Spinach*[2]. The problem set consists of a three Hilbert-space simulations that can be implemented in any programming language that handles matrix operations. We provide templates for MATLAB and Python to solve this exercise.

Problem 1: A simple isotropic spin-$^1/_2$ system

The MATLAB script `ex_1_one_spin_pulse_acquire.m` gives you the basic structure for this simulation. But some of the code was removed. Your task is to fill in the gaps. Note that sections in the script are indicated with "`%%`".

In the first three sections, we clear the workspace, build the spin operator matrices and then the Hamiltonian. MATLAB can natively deal with complex numbers. Note that the whole simulation will be performed in the rotating frame. That is why there is no Zeeman frequency in the Hamiltonian, and we simply use a chemical shift offset.

In section 4), you have to insert the following:

1. The starting density operator `rho0`. Note that this simulation assumes that we start from thermal equilibrium, and you can use the reduced density matrix in the high-temperature approximation.

2. The pulse operator `pulseOp`.

3. The detection operator `detOp`. We would like to use quadrature detection, which will allow us to distinguish positive from negative frequencies. Think about where your magnetization will end up depending on your choice of the pulse operator.

4. The time step `dt`, which has two main functions. First, it will be your propagation step. Since we cannot use continuous variables in a computer, we will propagate the density matrix step-by-step, and the step size will be given by `dt`. Second, it is also the "dwell time" of your acquisition, *i.e.* it also corresponds to the sampling step during the detection in an actual experiment. Which criterion has this time-step to fulfill to provide sufficient sampling?

5. The number of propagation and sampling steps `nPoints`.

Section 5) will finally contain the actual simulation.

[1] `https://inano.au.dk/about/research-centers/nmr/software/simpson/`

[2] `http://spindynamics.org/group/?page_id=12`

6. First, we will apply a $\pi/2$-pulse along the axis that you chose with `pulseOp`. What is the corresponding propagator? Write it to the variable `Upulse`[3]. This propagator is applied to the density matrix in the following line. Assume an ideal on-resonance pulse.

7. In the following, the density matrix will evolve under the system Hamiltonian. We will propagate the system stepwise from time t_i to $t_{i+1} = t_i + \Delta t$, where i runs from 1 to `nPoints` (MATLAB counting starts with 1, not 0), and the time step Δt is given by `dt` in the script. Write the propagator for free evolution for a time step `dt` into `U0`. Remember that the Hamiltonian in the script is given in linear frequencies.

8. Inside the `for`-loop, we always "detect" the signal before propagation. This corresponds to evaluating the expectation value of your detection operator. Write the corresponding expression into `sig(it)`

The rest of the scripts performs plotting and signal processing steps to get to the spectrum. You do not have to change anything but the parameter `twin`, which determines the decay time of an exponential window function that is multiplied onto the signal before Fourier transform. What is the effect of this parameter in time and in frequency domain?

Problem 2: An isotropic two-spin-$^1/_2$ system

The MATLAB script `ex_2_two_spin_pulse_acquire.m` gives you the basic structure for this simulation. It is essentially the same as in the first problem, so you will have to insert everything that you had to insert in Problem 1 again. The difference is that now we would like to simulate the additional effect of a $J-$coupling.

Section 2) starts with defining the matrices for the operators of a spin-1/2, but for this simulation we will need the operators in the uncoupled basis of two spin-$^1/_2$, *i.e.* $|I_1 m_1 I_2 m_2\rangle$. Construct them using the `kron()` function for the direct product.

Then use the solution of Problem 1 and adjust sections 4) and 5) to the two-spin problem.

Problem 3 (optional): Powder spectrum of a spin-$^1/_2$ system with chemical shift anisotropy (CSA).

The MATLAB script `ex_3_one_spin_CSA_pulse_acquire.m` gives you the basic structure for this simulation. It is again essentially the same as in the first problem, so you will have to insert everything that you had to insert in Problem 1 again. But now we calculate the spectrum of different orientations in a powder and add them up to get a powder spectrum.

1. Fill in section 3) again as in Problem 1. But this time we will start directly along x, and we will not use a pulse (we already know the solution, we do not need to calculate it numerically).

[3] NB: matrix exponentials in MATLAB can be written by `expm()`

2. In section 4) we define the parameters of the CSA tensor. Additionally we load a file that contains the orientations of different crystallites (in the variables `alphas`, `betas` and `gammas`). Additionally, every orientation comes with a corresponding probability (in the variable `weights`). Remember that not all angles have the same probability when integrating over a sphere (Which angles have the highest probability? Which have the lowest probability?). You can comment in different files. `leb_2ang_rank_11` contains a small number of orientations, which leads to a quick simulation that is practical for debugging. The other file contain more orientations. You will explore the effect of this later.

3. Section 5) is similar to the previous problems, but now we have to loop over all possible orientations. For each orientation, the Hamiltonian will be different. In order to calculate the latter you have to insert the following:

   - Use the provided function `erot()` to construct the rotation matrix `R` and use it to convert the CSA tensor in the principle axis system (`CSA_PAS`) to the CSA tensor in the lab frame (`CSA_LAB`).
   - Choose the appropriate element form the 3×3-matrix `CSA_LAB` that corresponds to the effective isotropic chemical shift `offset` of the given orientation.

4. Again build the propagator `U0` from the Hamiltonian.

5. Since we are adding up the signals from different crystallites, we always add the result of each crystallite to the signal of the crystallites that were already calculated. Remember that you have to weigh each signal with its corresponding probability. Insert you solution to the line `sig(it)=sig(it)+...`

6. Play around with the orientation files and the acquisition parameters to get a nice CSA spectrum.

7. What would we need to adjust to simulate a spectrum under MAS? How would the resulting spectrum look like for this system (without doing a simulation)?