Experiment 3: NMR Analysis of Keto-Enol Tautomerization
CHE 357
Alec Beaton

Introduction
In this experiment, you will use NMR spectroscopy to investigate solvent, temperature, and concentration effects on the tautomerization equilibrium constant (K_e) of 2,4-pentanedione (also called acetylacetone or acac). You will calculate the equilibrium constant under several different experimental conditions, using the relevant peak integrations of NMR spectra. You will additionally generate van't Hoff plots to determine thermodynamic quantities.

Experiments
For these experiments, we prepare NMR samples of 2,4-pentanedione in different deuterated solvents. For each solvent, we prepare a sample that is 0.1 mole fraction and 0.2 mole fraction in 2,4-pentanedione. Given the cost of deuterated solvents, many of these samples will be prepared for you when you arrive to lab. **You will be required to prepare 0.1 and 0.2 mole fraction samples of 2,4-pentanedione in acetone-d6. Please come to lab on Thursday with the math worked out, so you know the appropriate amounts of acetone-d6 and 2,4-pentanedione to measure out. Your math will need to be approved by me before you use any chemicals.**
The different solvents we will use are the following: acetonitrile-d3, chloroform-d, benzene-d6, acetone-d6, and benzene-d6 + mystery solvent.

We will operate the NMR spectrometers together.

Report
Be sure to include an abstract, introduction, experimental, results/discussion, conclusion, and references.
I will be looking for the following:
1. Introduction discussing basic NMR theory (see powerpoint slides and/or disucss with me)
2. **Table of K_e in the different solvents**, along with their dielecric constants (ε) and compare to literature values. **Based on these results, report identity of the mystery solvent with justification.**
3. **Table of K_e in each solvent at different concentrations**, and compare to literature values.
4. **Table of K_e at each temperature** in the two different solvents.
5. Using the values of K_e you have found at different tempertatures, **generate two van't Hoff plots** to determine the enthalpy and the entropy of the tautomerization reaction.
6. **Table of the enthalpies and entropies** found from your van't Hoff plots, and compare to literature values.
7. **Discussion** of your results – report any observed **trends** (e.g., the effect of solvent polarity on the equilibrium constant, the effect of temperature on the equilibrium constant, etc.) **and possible explanations** for those trends.
8. **Discuss reasons for deviations** in the values you report compared to the literature values.

Note: The van't Hoff plots will require use of the following equation: $\ln(K_e) = \frac{\Delta H}{RT} + \frac{\Delta S}{R}$

For the literature values of K_e, please see Zielinski and Grushow (2002) https://doi.org/10.1021/ed079p707.

Experiment 4: NMR Lineshape Analysis of Pyruvic Acid Hydrolysis
CHE 357
Alec Beaton

Introduction

In this experiment, we will use NMR spectroscopy to investigate the effect on H^+ concentration on the hydrolysis of pyruvic acid. We will use the linewidths of the proton resonances corresponding to pyruvic acid and to 2,2-dihydroxypropanoic acid to calculate the forward rate constant and the reverse rate constant, respectively. We will then combine this information to determine the overall rate constant for hydrolysis.

Experiments

For these experiments, nine samples will be prepared of pyruvic acid in D_2O with varying amounts of H^+ concentration. Given the use of concentrated HCl in the sample preparation, these samples will be prepared for you.

Sample No.	Vol. Pyruvic Acid (μL)	Vol. D2O (μL)	Vol. HCl (μL)
1	10	600	0
2	10	575	25
3	10	550	50
4	10	525	75
5	10	500	100
6	10	475	125
7	10	450	150
8	10	425	175
9	10	400	200

We will acquire ^1H NMR spectra on both the 300 MHz and the 400 MHz. We will operate the NMR spectrometers together.

Report

Be sure to include an abstract, introduction, experimental, results/discussion, conclusion, and references.

I will be looking for the following:

1. Introduction discussing basic NMR theory related to this experiment - why D_2O instead of H_2O? What is T_1 relaxation, and how is it different from T_2 relaxation? Which relaxation time is related to linewidth and why? What happens during the course of an NMR experiment – what does ATM, lock, shimming, and acquisition mean, in this context? (see powerpoint slides and/or discuss with me)
2. Introduction discussing kinetics used in this experiment (see handout)
3. **Plot of linewidth vs. $[H^+]$** for pyruvic acid proton resonance at both 300 and 400 MHz, with **linear fits.**

4. **Plot of linewidth vs. [H^+]** for 2,2-dihydroxypropanoic acid proton resonance at both 300 and 400 MHz, with **linear fits.**
5. Table listing the **forward rate constant** (k_f), **reverse rate constant** (k_r), and **equilibrium constant** determined at both fields. Compare the equilibrium constant to a literature value.
6. **Discussion of your results** – was the equilibrium constant found to be different at one field vs. another? Is this expected and why/why not?
8. **Discuss reasons for deviations** in the values you report compared to the literature values.

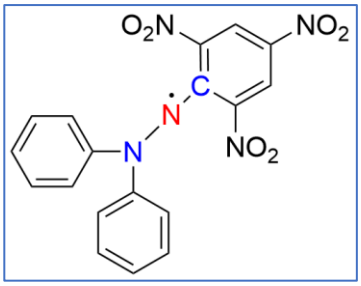
For the literature value of the equilibrium constant for pyruvic acid hydrolysis, please see the reference posted on blackboard by Griffiths and Socrates, 1966.

For a detailed discussion of the kinetics, see the **theory** and **methods** sections of the handout **NMR Study of a Reversible Hydrolysis Reaction.**

Lab 8: Introduction to ESR Spectroscopy
Alec Beaton
CHE 357
March 2020

Please respond to the following questions in a separate word document and email this to me – this serves as your lab report. Answers should be short and sweet (3 sentences max). Most of the answers will be provided either on the slides of the Power Point posted to Blackboard or in the accompanying video. If you refer to scientific literature, please provide a citation. The report is due **Thursday, April 9**.

1. Provide one example of a sample that is considered ESR active, and one example of a sample that is not ESR active ('ESR silent'). Why do you detect ESR signal in one sample but not the other?
2. Describe the difference between CW and pulsed/FT spectroscopy.
3. What is the origin of the upfield/downfield terminology used in NMR?
4. Why are the magnetic field strengths used in ESR spectroscopy so much weaker than those used in NMR spectroscopy?
5. Compare and contrast NMR and ESR spectroscopy (provide 3 ways in which these techniques are similar, and 3 ways in which they are different).
6. What is the hyperfine interaction in ESR spectroscopy? Does the hyperfine interaction affect NMR spectra – and if so, how?
7. What is nuclear spin state of the most naturally abundant isotope of Mn?
8. Based on your answer to question 7, sketch the ESR spectrum for the Mn^{2+} ion (please only consider the hyperfine interaction).
9. Consider the ESR spectrum of DPPH. In the figure to the right, which of the colored nuclei contribute to the hyperfine interaction, and which of the colored nuclei do not contribute to the hyperfine interaction? Why?



10. On slide 11, the ESR spectrum for 4-hydroxy-TEMPO is shown. Does the oxygen contribute to the hyperfine structure of this spectrum? Why or why not? Does the nitrogen contribute to the hyperfine structure? Why or why not?
11. What is the gyromagnetic ratio for the electron? What is the gyromagnetic ratio for the proton? Please express both in units of MHz/T.

Lab 7: Simulating NMR Spectra
Alec Beaton
CHE 357
March 2020

This is a worksheet meant to guide you through simulating NMR data using Python. **For this worksheet, you only need to respond to the questions throughout the worksheet that are bold and in blue (as shown here).** You do not need to write a formal lab report. Send your responses as a separate word document via email to be graded. As you work through this worksheet, please do not hesitate to contact me with any questions.

NOTE: ***You are strongly encouraged to type out all code excerpts below rather than copying and pasting.***

I. INTRODUCTION

The motives for this programming exercise are the following:

- Introduction to Python programming language (some basic structures, syntax)
- Review of elementary NMR theory
- Introduction to digital signal acquisition and processing
- Understanding the origin of the FID and Lorentzian lines through simulation

As discussed in Labs 2 and 3, during an NMR experiment, we acquire a signal in the time domain. This signal is referred to as the Free Induction Decay or FID. The general form of the FID is as follows:

$$s(t) = e^{i\Omega t - \frac{t}{T_2}}$$

where Ω represents the frequency of precession of the nucleus in question (typically called the "***offset***" – more on this later) and T_2 represents the ***transverse relaxation*** of that nucleus. As you learned in the previous labs, we typically analyze our NMR data in the frequency domain. This gives rise to the NMR spectrum. To obtain our spectrum from the time domain data, we perform a ***Fourier transformation***, as follows:

$$S(f) = \int_{-\infty}^{\infty} e^{-i2\pi ft}\, s(t)\, dt$$

In this programming exercise, we will use Python to simulate several FIDs and then carry out the Fourier transformations to analyze our simulated FIDs in the frequency domain. You will hopefully see the relationship between time domain data and frequency domain data, and how changes in the offset and transverse relaxation affect the properties of the signal.

II. SETTING UP PYTHON

Python is a popular programming language widely used in data science. It is similar to Matlab or Mathematica but is free to use. Python requires a package manager. I recommend downloading Anaconda as your package manager. Using this link, be sure to download the Python 3.7 version. Once

installed, you should be able to open a jupyter notebook by opening a terminal and typing **jupyter notebook**.

To open a terminal:

(A) On Mac, go to Launchpad >> Other >> Terminal, or use Finder to find Terminal. This should open a square window, called the Command Prompt, into which you should type **jupyter notebook**.

(B) On Windows, type Command Prompt in the start menu search bar (go to the bottom left corner and type in the search bar 'Command Prompt'). Type **jupyter notebook**.

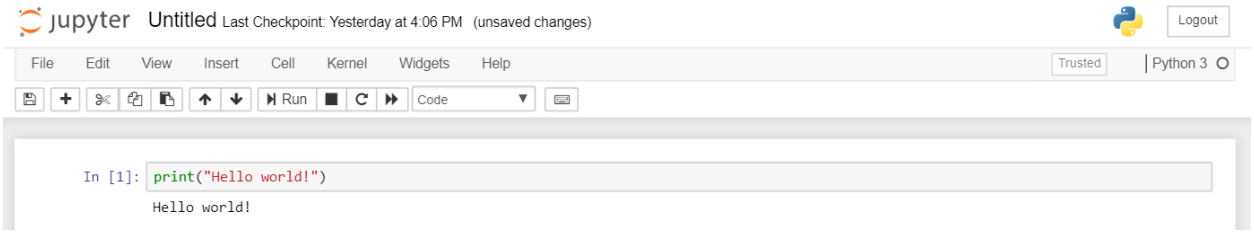
This should in turn open a webpage using your default browser. On this webpage, create a new notebook using New >> Python 3 notebook.

The basics of a jupyter notebook are that you enter code inside a cell. You then press **Shift + Enter** and this 'runs' the cell – i.e., runs the code inside the cell. It will generate an output of the code (if there is any) directly beneath the cell. The output of a code might be a "print" statement (which is basically exactly as it sounds – the code prints out a string of characters) or a plot of some sort.

In the first cell, try typing:

```
Print("Hello world!")
```

Then run the cell by clicking on it and hitting **Shift+Enter**. Beneath the cell, you should see the words Hello world! appear. It should look like the following:



If you have any issues with this, do not hesitate to email me and we can figure out a solution!

III. SIMULATING EFFECTS OF T_2

Open a jupyter notebook. In the first cell, enter the following:

```
from pylab import *
%matplotlib inline
```

what we are doing here is importing the pylab package, which gives us access to helpful tools like **fft** which allows us to perform a Fast Fourier Transformation. The second line imports the useful graphing package in Python, matplotlib, which we will use to visualize our data.

First, we set up our time axis. Our time axis will take the form of an **array** – one of the most important and basic data structures in any programming language. An array is merely a list of elements. You can

think of it as a column in Excel where you have entered sequential data points in each cell. We will create our time axis using the following:

```
t_axis = r_[0:2:6000j]
```

what we are doing here is generating an array of data points starting from 0 and ending at 2, with 6,000 evenly spaced points in between – you can think of this as 6000 cells in an Excel column. Note that we have given our array a unique name **t_axis**. At any point in the code, we can refer back to this array by typing **t_axis**.

For the sake of the simulation, we have created a time axis that starts at t = 0 seconds and ends at t = 2 seconds, with 6,000 points in between. This is the same as acquiring our FID for 2 seconds with 6,000 time points. The concept of time points introduces the notion of a discrete function vs. a continuous function. Note that equations 1 and 2 above are continuous functions. Continuous functions are very familiar from math courses and simple to understand, but in reality we deal with discrete functions when acquiring our data – the difference being that, instead of dealing with data composed of an infinite number of points, we have a dataset which is composed of a finite number of points spaced evenly apart.

Thus we can only ever acquire our data discretely, not continuously. The more times we sample the data in a fixed amount of time, the closer we get to acquiring a continuous FID. The limits on our ability to sample are often determined by instrumental features which we need not worry about.

This introduces an important concept in signal acquisition, called the **dwell time.** This is the amount of time that elapses between each consecutive time point. Typically this number is very small, but it is very important for our consideration of the data in the frequency domain due to the **Nyquist condition**

$$\Delta t = \frac{1}{SW}$$

which determines the **spectral width** or SW of our frequency domain data.

Stated alternatively, when we take the Fourier transform of our discretely sampled time domain data, we will obtain a frequency domain spectrum that spans our spectral width centered about 0. This means the maximum frequency we can detect is $\frac{1}{2\Delta t}$ otherwise known as the **Nyquist frequency**. The take away message from all this is that the time that elapses between the discretely sampled signal determines the frequencies we are able to detect.

We can set up our frequency axis as follows:

```
dwell_time = t_axis[1] - t_axis[0]
print("Dwell time is",dwell_time,"seconds.")
SW = 1./dwell_time
print("SW is",SW,"Hz, and Nyquist frequency",SW/2.,"Hz.")
f_axis = r_[-SW/2:SW/2:6000j]
```

Here we are finding our dwell time by subtracting the first element of our time axis (t_axis[0]) by the second element of our time axis (t_axis[1]). We then determine the spectral width from the dwell time, and set up our frequency axis to be centered about 0 with the proper spectral width. **This also prints the dwell time, spectral width, and Nyquist frequency, which you should report.**

Next we will generate the signal. We first define an array of zeros with the following:

We will next simulate the data with no offset. This is equivalent to using the following form for the time domain data:

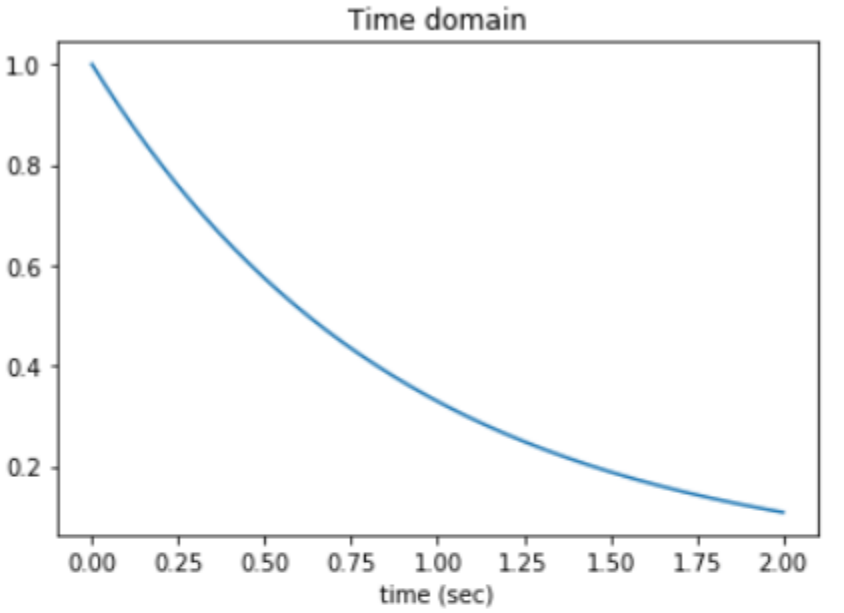
$$s(t) = e^{-t/T_2}$$

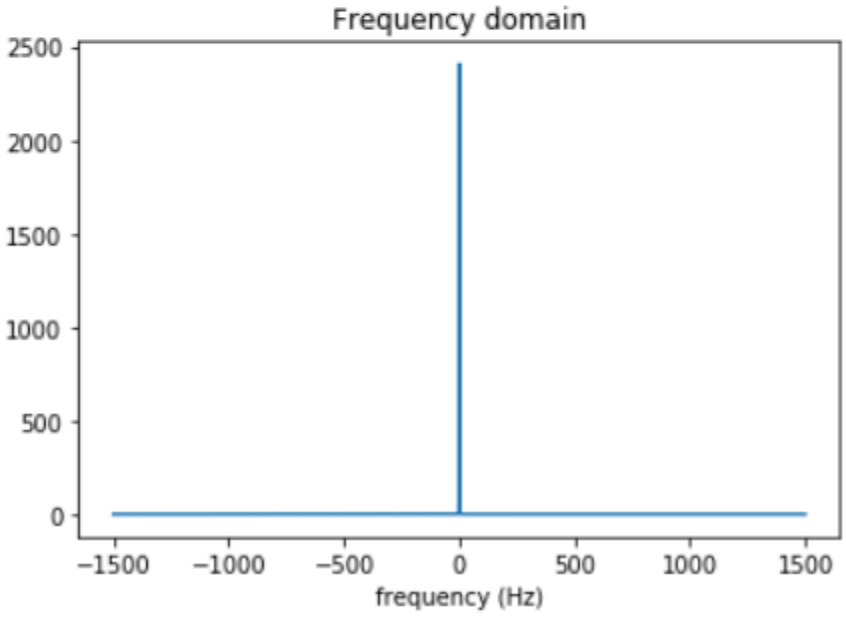
Enter the following code into a single cell:

```
T2 = 0.9
signal = zeros_like(t_axis,dtype=complex128)
signal += exp(-t_axis/T2)
figure('time domain')
title('Time domain')
plot(t_axis,signal)
xlabel('time (sec)')
figure('frequency domain')
title('Frequency domain')
plot(f_axis,fftshift(fft(signal)))
xlabel('frequency (Hz)')
```

First we choose our T_2 to be 0.9 seconds. We then declare an array named **signal** that has the same size as our time axis, but is composed of just zeros. We have importantly indicated that this array must be able to accommodate *complex* data points, which matters when it comes to quadrature detection in NMR. We then load the data into this array, by using the **+=** symbol. This is equivalent to writing signal = signal + exp(-t_axis/T2) but is just shorthand. We then prepare two figures, one for time domain signal and the other for frequency domain signal. In the first, we plot the FID by first specifying the x-axis (t_axis) and then the y-axis (signal). In the second, we plot the frequency domain spectrum in a similar manner. For the y-axis, we are not interested in plotting signal, but rather the Fourier transform of signal. You can consider fftshift(fft( )) to perform the equivalent of the Fourier transformation.

This should generate the following two plots:





This is precisely what we expect: we see a simple decaying exponential for our time domain data, and a sharp peak for our frequency domain data. **Change the value of T2 to 0.1 and attach screenshots of the plots. What is different about the time domain signal (i.e., is there a sharper decay or a slower decay)? What is different about the frequency domain signal (i.e., is the lineshape wider or more narrow)?**

To facilitate analysis, you can change the limits of the frequency domain plot by including the following:

```
xlim(-500,500)
```

This essentially "zooms" in on the frequency domain plot, from -500 to 500 Hz.

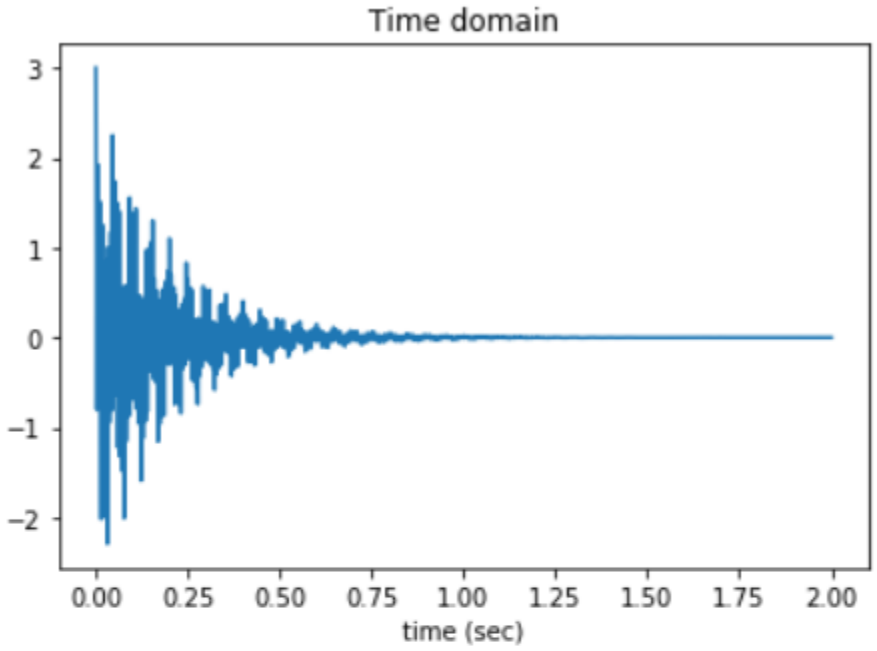
IV. SIMULATING EFFECTS OF OFFSET

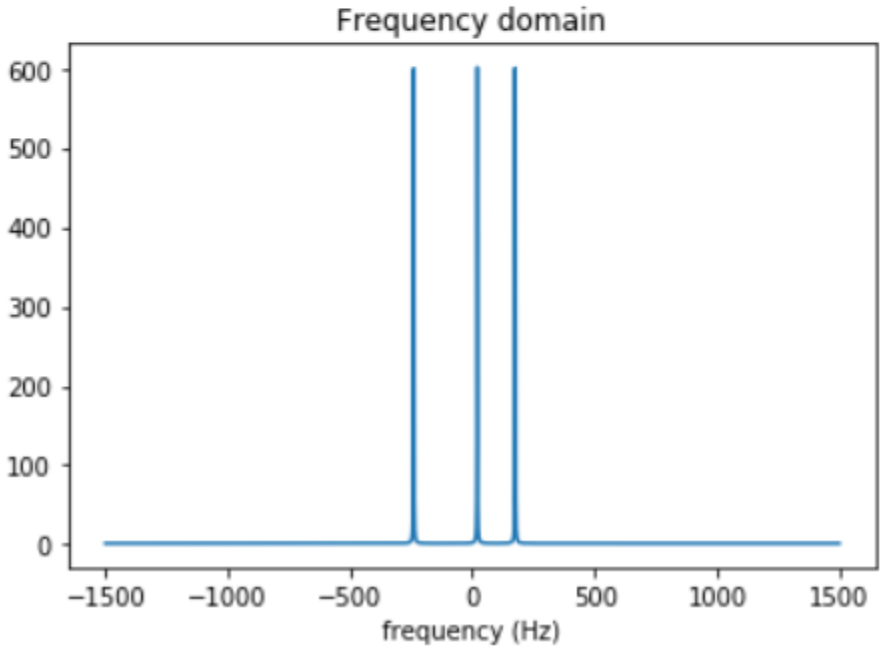
In the previous analysis, we had only considered our system with no offset – i.e., a single spin with a resonance frequency of exactly the carrier frequency of our spectrometer (e.g., 400 MHz). Often times, we detect signals that are ***offset*** from the carrier frequency of our spectrometer. This is what gives rise to chemical shift. We will next consider the case of *uncoupled* spin systems composed of multiple spins, which resonate at frequencies which are slightly off from the carrier frequency.

```
offset_list = [-240,20,175]
T2 = 0.2
signal = zeros_like(t_axis,dtype=complex128)
for f in offset_list:
    signal += exp(1j*2*pi*f*t_axis - t_axis/T2)
figure('time domain')
title('Time domain')
plot(t_axis,signal)
xlabel('time (sec)')
figure('frequency domain')
title('Frequency domain')
plot(f_axis,fftshift(fft(signal)))
xlabel('frequency (Hz)')
```

The code should look very familiar. The major differences are our introduction of the **offset_list** which is an array of values. These represent the resonance frequencies of three different spins. The first spin resonates 240 Hz *below* the carrier frequency, while the other two resonate at 20 Hz and 175 Hz *above* this frequency. We have next used the form of equation 1.1 for our time domain signal, where we have expressed the offset in Hz (i.e., $\Omega = 2\pi f$). We then use a ***for loop*** which enables us to iterate over all the elements of the **offset_list** one at a time. This enables us to calculate the FID for each of the three spins in our system. Importantly, the FIDs are additive, which means the overall FID for our three spin system is simply the sum of each individual FID (which we accomplish through the **+=** shorthand).

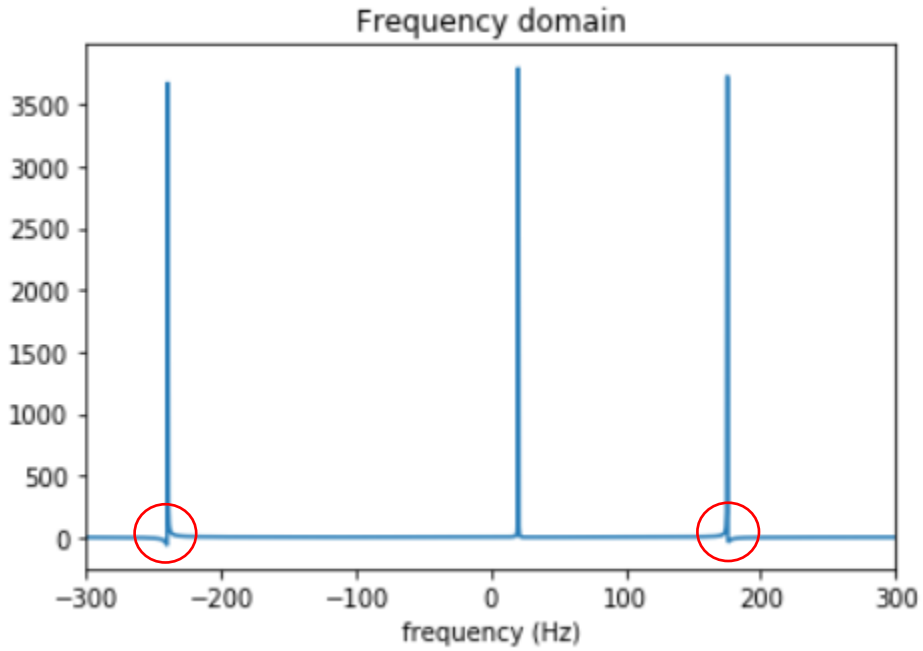
It should generate the plots below:





Here we see an *oscillatory* decay of our FID due to the presence of the imaginary number i (expressed as 1j in most programming languages). **Next change the value of T2 to 2.0 and attach screenshots of the plots.**

We have in fact increased the lifetime of our signal by increasing the T_2 – our FID in this case shows us that 2 seconds is not enough time to acquire the full signal. By only acquiring our signal for 2 seconds, and are only observing a portion of the signal. This in turn leads to ***phase errors*** in our Fourier transformed data, which are indicated in the red circles below.



These phase errors lead to imperfect lineshapes and are a nuisance for quantitative NMR, which is important in most aspects of routine NMR such as in organic chemistry and metabolomics.

Next, scroll to the top of the page and click on 'Kernel' – in the dropdown list that emerges, choose Restart & Clear Output. This should refresh the page and erase all data (but not *code*) in your notebook. In the second cell, change t_axis to end at 10 seconds instead of 2. Report the dwell time, spectral width, and Nyquist frequency for this new time axis. Did our spectral width increase or decrease relative to our first simulation? Run all the subsequent cells. Attach plots for the last cell (which should be the signal composed of three spins, with three different offsets, with a T2 of 2.0 seconds). In your frequency domain data, is the phasing error still present?