ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE
School of Computer and Communication Sciences

Software-Defined Radio: A Hands-On Course | Assignment date: Sept 21, 2016
Due date: Sept 28, 2016

A MATLAB Refresher and AM Signals

In this first assignment, you have to write and upload on moodle several MATLAB files. Take advantage of the reference card that you find on the course web page. Also, take into account the style guidelines discussed in class.

Unless otherwise instructed, in this assignment you are allowed to use only standard MATLAB functions (as opposed to toolboxes).

1. MATLAB Refresher

Exercise 1. This short exercise will be useful for the next exercise. Let $s(t)$ be a baseband signal with support in the interval between 0 and `d` [seconds]. Let `s` consist of its samples taken every `1/fs` seconds, with the fist sample taken at $t = 0$.

1. Generate the vector `t` such that `plot(t, s)` produces a plot with the correct time scale on the x-axis.
2. Let `s_f` be the DFT of `s`. Generate the vector `f` such that `plot(f, fftshift(abs(s_f)))` produces a plot with the correct frequency scale on the x-axis.

Exercise 2. Write the function `tfplot` characterized by the following header and save it as `tfplot.m`. Keep this file somewhere safe, as it will be used again in future assignments.

```
function tfplot(s, fs, name, plottitle)
% TFPLOT Time and frequency plot
%    TFPLOT(S, FS, NAME, PLOTTITLE) displays a figure window with two subplots.
%    Above, the signal S is plotted in time domain, with the x-axis
%    labeled in seconds. The sampling frequency is FS.
%    Below, the absolute value of the signal in the frequency domain, with the
%    x-axis labeled in Hz. NAME is the "name" of the signal, e.g., if NAME is
%    's', then the labels on the y-axes will be 's(t)' and '|s_F(f)|',
%    respectively.  TITLE is the title that will appear above the two plots.
```

Fig. 1 shows the output of the function when called like this:

```
tfplot(m, fs, 'm', 'A sinc signal')
```

(where `m` is the sinc signal shown in the figure).

For the future usage of the function `tfplot` you can use continuous line instead of the stem plot shown in Fig. 1. However, one should keep in mind the fact that we actually work with sampled versions of continuous time signals.

For helping you, we provide on the course webpage[1] a function `sincplot` which produces Figure 1. `sincplot` uses a compiled version of the function `tfplot`. In order to test your implementation, you can just replace `sol_tfplot` with your own version of `tfplot`.

2. Amplitude Modulation (AM)

Exercise 3.

1. Write a function `s = my_ammod(m, K, A, fc, fs)` that modulates the signal `m` using AM, where `fc` is the carrier frequency f_c, `K` and `A` are constants as defined in class, and `fs` is the sampling frequency. Save the function as `my_ammod.m`.
2. In a new function `test_am()`, create the message signal

$$m(t) = \frac{1}{2}\cos(2\pi f_{\text{info}} t),$$

   where $f_{\text{info}} = 10\,\text{Hz}$, $f_c = 300\,\text{Hz}$, $A = K = 1$, and $f_s = 4000\,\text{Hz}$. Let the duration of the signal be $d = 1\,\text{s}$. Use your function `tfplot` to display the message signal as well as the modulated signal. Do the plots correspond to your expectations?
3. Write a function `m = my_amdemod(s, fc, fs)` that demodulates the AM signal `s` and returns the message signal `m`, normalized to have values between -1 and 1. As before, `fc` is the carrier frequency and `fs` is the sampling frequency.

   Hint: To filter the signal, use a 2$^{\text{nd}}$ order Butterworth filter[2].
4. Go back to `test_am` and add code that uses `my_amdemod` to demodulate the signal returned by `my_ammod`. Plot the result using `tfplot` and verify that the message signal has been correctly reconstructed.
5. Now modify the parameters of the message signal $m(t)$ to produce a more audible signal, and use `sound` to play both the original and the reconstructed message signal to verify that they are the same. Note that we do not necessarily use the default sample rate of the command `sound` (you should check the help page).
6. Once you feel confident that your code works and that you understand what is going on, download the AM data file from the course webpage. This file contains an AM signal of carrier frequency $f_c = 20\,\text{KHz}$, sampled at $f_s = 100\,\text{KHz}$.

   Write a function `play_am()` that loads the signal from the file, demodulates it using `my_amdemod`, downsamples[3] the result by a factor 10 and plays it using `sound` (again, for this command you should pay attention to the sampling rate). Can you identify the music?

   As for the previous exercise, in order to help you with the implementation, we provide on the course website the compiled versions of the functions you have to write.

What to hand in:

An archive containing the files `tfplot`, `my_ammod`, `my_amdemod`, `test_am`, and `play_am`.

---

[1] on moodle.epfl.ch

[2] see the MATLAB help on the functions `butter` and `filter` to learn how to filter the signal.

[3] cf. the MATLAB function `downsample`

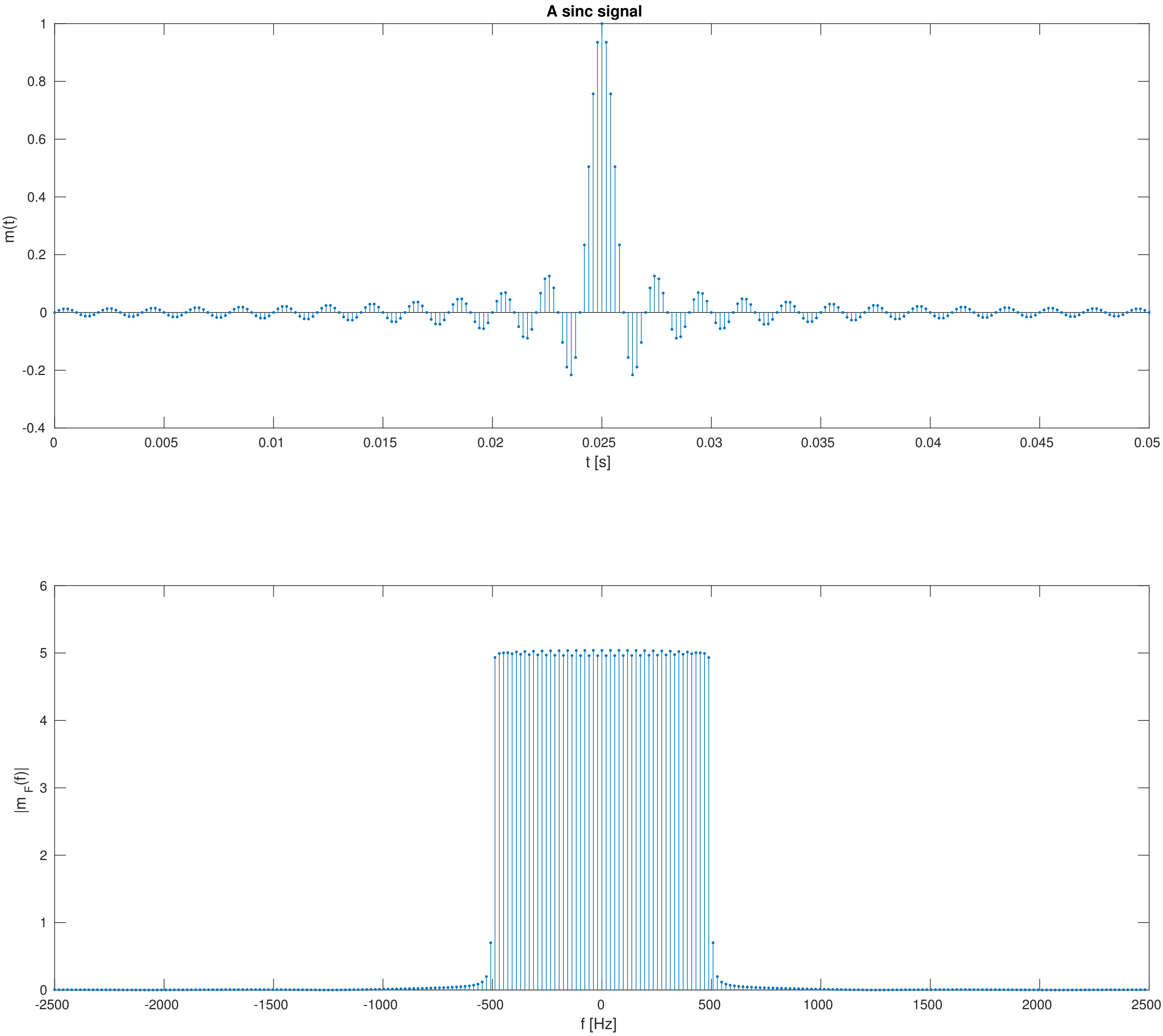

Figure 1: Example output of the function `tfplot`