York University
Department of Electrical Engineering and Computer Science
EECS 4214

# Lab #1 Signals and Systems Refresher

Sebastian Magierowski

Sept. 5, 2015

## 1 Purpose

Digital communications requires a fairly sophisticated understanding of signals and the systems that process those signals. In this lab you will review basic concepts related to the analysis of signals and the systems. This should be review from your Signals and Systems course. You may want to refer to the material from that course if you need more brushing up. This lab should also serve as a chance to practice the manipulation of signals in MATLAB (You don't have to, but I highly, highly, very highly recommend you use MATLAB for your computing needs in the course).

## 2 Objectives

By the end of this lab, you will be able to:

1) Generate deterministic signals of your own choosing.
2) Compute a variety of signal transforms.
3) Filter incoming signals.

## 3 References

1) Bernard Sklar text (required text for the course): Section 1.2, Section 1.6, Appendix A, Appendix E.

## 4 Continuous-Time Signals

Communication is based on the exchange of signals. In this class we focus on the exchange of electromagnetic (EM) signals. For our purposes we don't have to worry about the physical details of this signal but rather discuss its mathematical representation as a **continuous** function of time, $x_c(t)$. For example,

$$x_c(t) = A\sin(\omega t + \theta) \tag{1}$$

could represent the characteristics of an electronic voltage or current waveform resulting from some EM excitation. Naturally much more sophisticated waveforms are possible. In the equation above $\omega$ is the **angular frequency**, typically in radians-per-second and $\theta$ is the phase of the signal. Recall the relationship between the angular frequency and the **natural frequency** (I'll just refer to this as the "frequency"), $f$

$$\omega = 2\pi f \tag{2}$$

where $f$ is in units of cycles-per-second or Hz for short.

**Problem 1** On your computer create a plot of a sine wave showing two periods (no more no less if you want full marks). The amplitude of the sine wave should be 2 and its frequency should be 5 MHz and its phase should be 0. Did I mention that I highly recommend you use MATLAB for questions such as this? Not only is it really easy to plot things in this software, but MATLAB also comes equipped with a bunch of powerful functions that you will be using again and again in this course. Let me give you some hints.

To create a vector (or array) of numbers I can just type

```
t = 0:1e-3:12.1;
```

which creates a row vector of 12101 elements consisting of numbers from 0 to 12.1 inclusive in steps of 0.001. Such a vector can then be used to create another vector, say

```
omega = 23;
y = 12*cos(omega*t);
```

where `omega` defines our angular frequency an `y` is another 12101 element vector now corresponding to a cosine function. To plot the result in a nice an pretty fashion you can use something like

```
plot(t,y,'k','LineWidth',2);grid;
set(gca,'Xscale','linear','FontSize',12,...
    'MinorGridLineStyle','none');
xlabel('$time$ [s]','Interpreter','latex',...
       'FontSize',14,'FontName','times','Units','Normalized',...
       'Position',[0.5,-0.07],'FontName','times-roman');
ylabel('Amplitude','Interpreter','latex',...
       'FontSize',14,'FontName','times','Units','Normalized',...
       'Position',[-0.08,0.5],'FontName','times-roman');
title('A Signal Plot')
```

where I'll let you figure out and decide which parts you want to use and which parts you do not want to use.

Fig. 1 is an example (but you should be able to notice that it is not quite correct) of what I'd like to see from you in this problem. Notice the judicious labelling (names and units where applicable). **For full marks** you should be doing the same in all your quantitive plots.

Besides being continuous the signal above is also **periodic** meaning that $x_c(t) = x_c(t+kT)$ where $k$ is any integer you want and $T = 1/f$ is the **period** of the signal; the time over which the signal repeats.

An example of a **non-periodic** signal that you will see ALOT in this course is

$$x_c(t) = \frac{\sin(\pi t f)}{\pi t f} = \mathrm{sinc}(tf) \tag{3}$$

which we often refer to as the "sinc" pulse or sinc function.

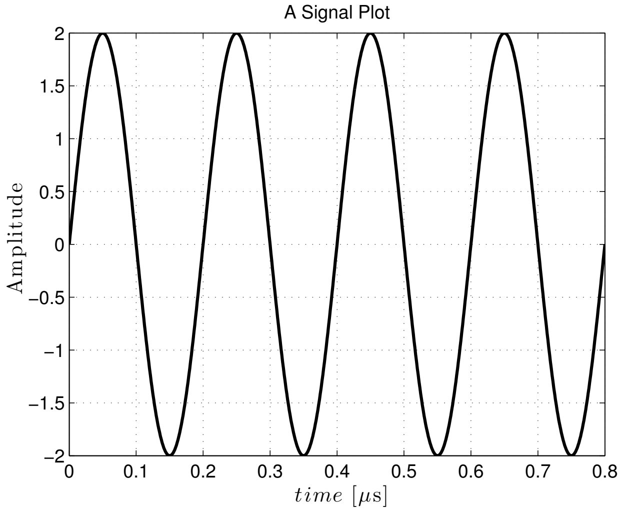

Figure 1: An example of a good (but not quite correct) reply to **Problem 1**.

**Problem 2** Sketch by hand (but by all means use MATLAB to verify your answer if you wish, it even comes with a convenient `sinc` function) a sinc pulse (of maximum height equal to unity) versus $t \cdot f$ for a frequency $f$ of 5 MHz. Your sketch should clearly demonstrate the symmetry of the function. Carefully annotate the $x$ and $y$ axes of your sketch (with labels and numbers) so that we are sure you understand key properties of this function.

# 5 Continuous-Time Signal Transforms

It is easy to imagine our communication signals as continuous functions of time as discussed above. Our own actions seem to clearly occur with respect to time that thinking of signals operating in the temporal dimension is straightforward.

As you have already learned though, there are other ways of expressing our signals. These alternate expressions are extremely popular because they highlight other signal properties (e.g. spectral properties) and because they greatly simplify certain signal calculations (e.g. the effect of filtering).

The **Fourier series** (FS) is a prominent example of alternate expressions applicable to periodic signals with period $T_o = 2\pi/\omega_o = 1/f_o$ and can be stated as

$$\mathbf{FS}: \; x_c(t) = \sum_{n=-\infty}^{\infty} a_n e^{jn\omega_o t} \quad \| \quad \mathbf{FS}^{-1}: \; a_n = \frac{1}{T_o}\int_t^{t+T_o} x_c(t) e^{-jn\omega_o t}\,\mathrm{d}t \tag{4}$$

where, with regards to the **FS** expression, we see that the continuous signal may be stated in terms of a sum of complex exponential functions or, more generally, **basis functions**.

**Problem 3** By hand, calculate all of the complex Fourier series coefficients (i.e. all the non-zero $a_n$) of the sine wave described in **Problem 1**.

My use of the label, $\mathbf{FS}^{-1}$, is meant to imply a concept akin to “inverse Fourier series”, but this is not at all standard in the literature (so don’t expect to find a consistent use if googling) and I use it as a personal tool to help me organize my thoughts with respect to other signal transforms. In essence my $\mathbf{FS}^{-1}$ expression above is just the equation you use to calculate your **Fourier coefficients**, $a_n$. In the literature my **FS** is sometimes referred to by the general label of "synthesis equation" and my $\mathbf{FS}^{-1}$ is sometimes referred to by the general label of "analysis equation" and the two form an "analysis-synthesis" pair.

Another popular and related expression applicable to signals of finite energy[1] is the **Fourier transform** (FT). If you scan the literature (or just google it if feeling lazy) you'll see a bunch of different expressions for this which are all basically equivalent after a little bit of scaling here and there. In engineering we typically use the forms

$$\mathbf{FT}^{-1}: \; x_c(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X_c(j\omega)e^{j\omega t}\mathrm{d}\omega \;\;\|\;\; \mathbf{FT}: \; X_c(j\omega) = \int_{-\infty}^{\infty} x_c(t)e^{-j\omega t}\mathrm{d}t \tag{5}$$

$$\mathbf{FT}^{-1}: \; x_c(t) = \int_{-\infty}^{\infty} X_c(f)e^{j2\pi ft}\mathrm{d}f \;\;\|\;\; \mathbf{FT}: \; X_c(f) = \int_{-\infty}^{\infty} x_c(t)e^{-j2\pi ft}\mathrm{d}t \tag{6}$$

You will probably recall that $X_c(j\omega)$ and, identically, $X_c(f)$ define our signal spectrum.

**Problem 4** By hand, calculate the Fourier transform of the sinc pulse described in **Problem 2**.

Actually you can find the FT of signals that are infinite in energy[2] as long as those signals are periodic. How? Just represent your periodic signal in FS form

$$x_c(t) = \sum_{n=-\infty}^{\infty} a_n e^{jn\omega_o t} \tag{7}$$

and take the FT of that

$$X_c(j\omega) = 2\pi \sum_{n=-\infty}^{\infty} a_n \delta(\omega - n\omega_o) \tag{8}$$

$$X_c(f) = \sum_{n=-\infty}^{\infty} a_n \delta(f - nf_o) \tag{9}$$

where $\delta$ is the Dirac delta function.

# 6 Discrete-Time Signals

The EM signals that we send through wires or through the air (or water or space or ... ) are indeed continuous in time (i.e. analog) as introduced above. This should make sense since the scales on which we excite and sense signals in the natural world effectively resemble a continuum in time and space (yes, if you start looking closely enough you will start seeing that at some scales this is an illusion).

However the vast majority of the equipment we build to actually extract the information contained within our signal is not analog. It is digital. Digital equipment is defined by the fact that it process signals that can only take on discrete quantities in time and discrete quantities in amplitude. That is signals that assume only certain values (discrete-amplitude) only at certain times (discrete-time). You may imagine that a machine that can only process digital signals is much simpler to make than a machine that has to process analog signals. If you did imagine this then you are right! In fact they are so easy to make that it is possible to think of them in a highly modular manner and hence construct extremely sophisticated versions of them (alas, so sophisticated

[1] "Energy signals" which are signals of finite energy are discussed in class and in Ch. 2 of the Sklar text.

[2] Infinite energy signals are so-called "power signals" and are discussed in class as well as Ch. 2 of the Sklar text.

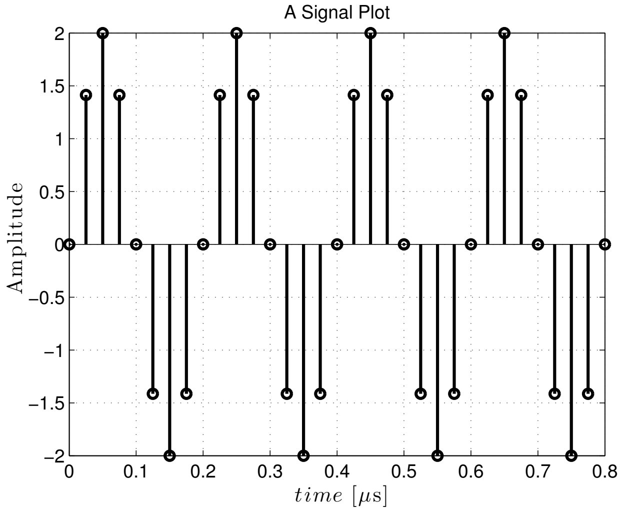

Figure 2: An example of the general (but not correct) form of the reply to **Problem 5**.

that they eventually become quite hard to engineer and require the collaborative effort of 100s of people) with unprecedented levels of ability to process discretqe signals in all sorts of complex manners.

Of course you might wonder what the point of processing discrete signals is anyway. Aren't these just useless because they only have certain values at certain times. Hopefully by now you appreciate enough about word length and Nyquist-Shannon sampling to suspect that discrete signals are in fact no that useless.

In the following discussion we will address the handling of signal discrete in time and possibly continuous in amplitude. In other words, not digital signals, but rather **discrete-time signals**.

A common reference to a discrete-time signal is in terms of a continuous analog signal sampled at discrete instances of time. That is, we refer to $x_c(nT_s)$ where $T_s$ is the **sampling period** and $n$ is an integer. The **sampling frequency** is denoted by $f_s = 1/T_s$. Since the samples of a signal are just a sequence of numbers it is also very common to drop any direct reference to time and employ the notation $x[n] = x_c(nT)$.

**Problem 5** Revisit **Problem 1** and this time "plot" the sine waveform using a more appropriate representation. In particular we need to acknowledge that all the signals we display on our computers, which are digital machines, are in fact digital representations of any ostensibly continuous-time signals. Continue to pretend that your computer can produce perfectly accurate values of your continuous-time sine wave, but assume that $f_s = 3 \times 5$ MHz (i.e. oversample your ideal continuous-time waveform by 3). Instead of using MATLAB's `plot` function use its `stem` function to display the signal in a way that more clearly conveys its discrete-time nature. Fig. 2 is an example (again, not correct) of the form that your answer should take.

If our signal repeats (i.e. is periodic) every $N_o = 2\pi/\Omega_o$ samples $x[n] = x[n + lN_o]$ (where $l$ is an integer) then we can employ a discrete-time Fourier series (DTFS) expression

$$\textbf{DTFS}:\ x[n] = \sum_{k=-\infty}^{\infty} c_k e^{jk\Omega_o n} \ \ \| \ \ \textbf{DTFS}^{-1}:\ c_k = \frac{1}{N_o}\sum_{n=m}^{m+N_o-1} x[n]e^{-jk\Omega_o n} \tag{10}$$

Note the similarities and differences between these expressions and those associated with continuous-time signals above. Again my use of the term $\textbf{DTFS}^{-1}$ is not a standard way of labelling, in general we use this function to calculate the Fourier series coefficients of the discrete-time signal. The angular frequency variable, $\Omega_o$, is effectively in units of radians per sample (not radians per second).

For signals of finite duration we also have the discrete-time Fourier transform (DTFT) through which to form alternate signal expressions

$$\textbf{DTFT}^{-1}: \; x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\Omega})e^{j\Omega n}\mathrm{d}\Omega \quad || \quad \textbf{DTFT}: \; X(e^{j\Omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n} \tag{11}$$

As we have normalized time from $t$ to $n$ in the DTFT we also normalize [3] our concept of frequency to $\Omega = \omega T_s$ (also in radians/sample). Note that $\Omega$ is *continuous* so the DTFT of $x[n]$ **is continuous**. Recall also that the function $e^{jx}$ is periodic over $x$ with a period of $2\pi$ so the DTFT $X(e^{j\Omega})$ is periodic with $\Omega$ over $2\pi$. Typically we present $X(e^{j\Omega})$ over $\Omega$-domain in the interval $[-\pi, \pi)$ (that is $-\pi \leq \Omega < \pi$, but no one will fault you for just using $-\pi \leq \Omega \leq \pi$). From our normalization relation you should appreciate that $\Omega = \pi$ is the normalized version of $f = f_s/2$ (i.e. the Nyquist sampling frequency) in the $f$-domain.

As with the (continous-time) FT, the DTFT can also be used describe a periodic signal, $x[n] = x[n + lN_o]$, first expressed as a DTFS. In particular we can obtain the expression

$$X(e^{j\Omega}) = \frac{2\pi}{N_o}\sum_{k=-\infty}^{\infty} c_k \delta(\Omega - k\Omega_o) \tag{12}$$

Finally we are ready to state the final transform of the discussion. The discrete Fourier transform (DFT). This is arguably the most practical of transforms for our needs as it expresses signals consisting of a finite number of samples, $N$, (a very practical case indeed) and results in a transformation also consisting of a **discrete spectrum** (unlike the DTFT) with $N$ components. This latter point is sometimes referred to as **frequency sampling**. Without further ado the DFT expressions are

$$\textbf{DFT}^{-1}: \; x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{jk\frac{2\pi}{N}n} \quad || \quad \textbf{DFT}: \; X[k] = \sum_{n=0}^{N-1} x[n]e^{-jk\frac{2\pi}{N}n} \tag{13}$$

MATLAB can conveniently calculate the DFT for you with its function `fft`. MATLAB can also calculate the inverse DFT for you with the function `ifft`.

**Problem 6** Revisit the signal in **Problem 1** and this time compute the absolute values of DFT (i.e. $|X[k]|$) of that signal using an oversampling rate of 8× and 34×. In both cases gather a total of 512 samples. Show your DFT results (i.e. $|X[k]|$ vs $k$) in a plot (there are lots of points so its ok to use `plot`, but its also ok to use `stem`). Make sure that you show exactly the DFT results as defined in (13) above (you'll get a chance to manipulate you DFT results in the following problems). Which one of these DFT calculations do you think is a better estimate of the spectral content of the signal? Explain why you said one is better than the other. Explain why one oversampling rate gives a better estimate of the spectral content of the other.

**Problem 7** In **Problem 6** you plotted the DFT of a signal. But we can use the DFT to show approximations of our other transforms too. With that in mind use the DFT for the 8× oversampled signal that you calculated for that sine wave to plot the (approximation to) the imaginary component of DTFT (i.e. $\text{imag}\{X(e^{j\Omega})\}$) of the same signal over the interval $[-\pi, \pi)$. Make sure your plot's axes are clearly labelled. You might find the MATLAB function `fftshift` useful for this problem.

[3] The normalization, $\Omega = \omega T_s$, is clear if you look more deeply at the derivation of the DTFT, I just state the result in this lab write-up.

**Problem 8** In **Problem 6** you plotted the DFT of a signal. Now use this result to plot the Fourier series coefficients (i.e. the $c_k$ in $\mathbf{DTFS}^{-1}$) of this discrete-time signal (both real and imaginary components). Plot your results on the same $x$ axis used in **Problem 7**. Again you might find MATLAB's `fftshift` useful here. A little hint: Do you think you'll have to scale your DFT result in order to get the right value? Compare the expressions for **DFT** and $\mathbf{DTFS}^{-1}$.

**Problem 9** In **Problem 6** you plotted the DFT of a signal. But we can use this to also approximate the FT of a continuous version of that signal. Do so for the sine signal you've been analyzing throughout this lab. Plot your result over the interval $-f_s/2$ to $f_s/2$ (I'm not picky about $<$ and $\leq$) and just show the imaginary spectrum. Again subjecting your DFT to `fftshift` should be helpful and keep scaling in mind too!

**Problem 10** Show the expression for the FT (I don't need to see a derivation, but it would probably be helpful for you to do it as I might ask this on a test) of a **rectangular pulse** with a height of 1 and a duration of $T$ seconds.

**Problem 11** Use the DFT (i.e. `fft` in MATLAB) to plot the absolute value of the FT of a rectangular pulse of height 1 and duration of 2-$\mu$s (microseconds). On the same plot draw (using the computer, not by hand) the theoretically expected result (one useful MATLAB command for getting multiple curves on one plot is `hold`). Hint: Your DFT-based calculation and the theoretical result better be really close. You'll probably have to scale your raw DFT result and shift it. It's up to you to figure out by how much.

# 7 Filtering

We have only very basic things to say about filtering. Recall that the output of a continuous-time filter can be expressed as the convolution of that filter's input signal, $x_c(t)$, and its impulse response $h(t)$,

$$y(t) = h(t) * x_c(t) = \int_{-\infty}^{\infty} h(\tau)x_c(t-\tau)\mathrm{d}\tau = \int_{-\infty}^{\infty} h(t-\tau)x_c(\tau)\mathrm{d}\tau \tag{14}$$

The FT of the impulse response defines the frequency response of the filter. That is

$$H(j\omega) = \int_{-\infty}^{\infty} h(t)e^{-j\omega t}\mathrm{d}t \tag{15}$$

The FT's of the input and output signals can further be related via

$$Y(j\omega) = H(j\omega)X_c(j\omega) \tag{16}$$

The same ideas apply to discrete-time systems whose impulse responses are now considered in terms of the DTFT.

$$y[n] = h[n] * x[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] \tag{17}$$

$$H(e^{j\Omega}) = \sum_{n=-\infty}^{\infty} h[n]e^{-j\Omega n} \tag{18}$$

$$Y(e^{j\Omega}) = H(e^{j\Omega})X(e^{j\Omega}) \tag{19}$$

**Problem 12** Calculate by hand the impulse response, $h[n]$, of a discrete-time filter whose frequency response is 1 for $|\Omega| \leq 0.3\pi$ and 0 for $0.3\pi < |\Omega| \leq \pi$.

In MATLAB once you have the impulse response, $h[n]$, of your desired filter you can employ the `filter` function to see the effect. In particular if you can get your filter's response to any input signal if you use the line

```
y = filter(b,1,x)
```

where `b` is your whole impulse response $h[n]$ and `x` is your input sequence $x[n]$ and `y` is your output sequence $y[n]$.

Of course if $h[n]$ is infinite you will have to truncate it somewhere as MATLAB obviously can't handle infinitely long vectors.

**Problem 13** Compute (e.g. with MATLAB) and plot (on the same plot) the magnitude of the DT (i.e. discrete-time) frequency response from 0 to $\pi$ (i.e. just show the single-sided frequency response) for the filter described in **Problem 12** when only $L = 11, 23, 51$ samples of the impulse response $h[n]$ are retained. Comment on the results. (Note, there's no need to use the `filter` function for this problem.)

**Problem 14** Confirm that the filter for $L = 23$ from **Problem 13** works. Excite it with

$$x[n] = 2 \cdot \cos\left(\frac{\pi}{10}n\right) + 5 \cdot \sin\left(\frac{2\pi}{3}n\right) \tag{20}$$

and show that at least qualitatively the response is expected by examining the spectrum of the output, $y[n]$. Show this by plotting the absolute value of the DTFT of the filter's input and output signals on the same graph. As always clearly label which curve is which (the MATLAB function `legend` may be helpful in this regard). In this problem you should also be using the `filter` function.