Posting Date: October 19, 2015.

Problem 1: Determine via numerical integration the percentage of power that lies in (i) the main lobe and (ii) the main lobe and the first side lobes of the power spectral density $S_X(f)$ of a random binary sequence with a chip interval T_c. Hint: without loss of generality and to make things easy, we can set $T_c = 1$.

Problem 2: Because of the rounded shape of the $\text{sinc}^2(f)$ function, the spectrum within the null-to-null bandwidth of the main lobe of the power spectrum $S_X(f)$ (i.e., B_1) of the GPS L_1 C/A code is not filled uniformly with power. In fact, there is very little power at the edges of the B_1 interval where, as it turns out, power matters most for accurate code phase (i.e., pseudorange) measurements. Let's consider a hypothetical radionavigation system with a spreading waveform that makes more efficient use of the spectrum within B_1. Let the (equivalent baseband) power spectrum of the hypothetical system's spreading waveform be given by

$$S_X(f) = \frac{1}{W}\Pi(f/W).$$

(a) Find the autocorrelation function $R_X(\tau)$ of the spreading waveform.

(b) How wide is the main peak in the autocorrelation function $R_X(\tau)$ (from first left to first right zero-crossing)?

(c) Compare this width to the width of the peak of the autocorrelation function $\bar{R}_X(\tau)$ for a random binary sequence with an equivalent null-to-null bandwidth $B_1 = W$.

(d) Given the results of your comparison, which spreading waveform (the new one introduced here that makes more efficient use of the B_1 spectrum interval or the traditional random binary sequence) would tend to produce more robust code phase measurements at high values of the noise floor N_0?

(e) Why do you suppose the random binary sequence (actually a pseudorandom approximation of it) was historically used for GPS?

Problem 3: In this problem, you will create the power spectrum of the GPS L_1 signal taken by the Stanford 46-meter diameter dish. Download the script `bigDishPSD.m` from iLearn and the data file

`prn31_22apr03_01hrs40min00sec_gmt_fl1_46_08mhz_250msec.mat`

from the server at

`http://overmars.engr.ucr.edu/GNSS_Course`

Complete the script `bigDishPSD.m` as necessary to estimate the power spectrum numerically using MATLAB's `pwelch` function. Discuss your results.

Problem 4: Write a MATLAB function that computes an LFSR-generated pseudorandom sequence, given the following inputs: LFSR size, connections, and initial state. Your function should adhere to the following interface specification:

```
%[lfsrSeq] = generateLfsrSequence(n,ciVec,a0Vec)
%
% Generate a 1/0-valued linear feedback shift register (LFSR) sequence.
%
% INPUTS
%
% n ------ Number of stages in the linear feedback shift register.
%
% ciVec -- Nc-by-1 vector whose elements give the indices of the Nc nonzero
%          connection elements.  For example, if the characteristic polynomial
%          of an LFSR is f(D) = 1 + D^2 + D^3, then ciVec = [2,3]' or [3,2]'.
%
% a0Vec -- n-by-1 1/0-valued initial state of the LFSR, where a0Vec = [a(-1),
%          a(-2), ..., a(-n)]'.  In defining the initial LFSR state, a
%          Fibonacci LFSR implementation is assumed.
%
% OUTPUTS
%
% lfsrSeq -- m-by-1 vector whose elements are the 1/0-valued LFSR sequence
%            corresponding to n, ciVec, and a0Vec, where m = 2^n - 1.  If the
%            sequence is a maximal-length sequence, then there is no
%            repetition in the m sequence elements.
%
%+------------------------------------------------------------------------------+
% References:
%
%
% Author:
%+==============================================================================+
```

Problem 5: In this problem, you will experiment with the properties of pseudorandom binary sequences generated by various means. Download the MATLAB script `correlationExperiments.m` from iLearn, along with the supporting functions, `oversampleSpreadingCode.m`, `ccorr.m` and `pi2str.m`. Your own `generateLfsrSequence.m` function supplies the final required supporting function.

The top-level script `correlationExperiments.m` generates pseudorandom binary sequences by one of three methods: (1) via the MATLAB `rand` function, (2) from a mapping of digits of the transcendental number π, and (3) from m-sequences. You can use this script to experiment with the properties of sequences generated in these different ways.

Read through the script and run it a few times with `codeType = 'rand'` to get a feel for what it does. You may adapt the script or write an entirely new script to answer the questions below.

(a) In the lecture, it was claimed that the variance of the cross-correlation

$$R_{ab}(k) = \sum_{n=1}^{N} a'_n b'_{n+k}$$

between two random ± 1-valued sequences of length N at any particular value of k is equal to N. Verify this by evaluating $R_{ab}(k = 0)$ for 100 different random sequence pairs and calculating the sample variance of your collected values.

(b) The autocorrelation

$$R_a(k) = \sum_{n=1}^{N} a'_n a'_{n+k}$$

of ± 1-valued m-sequences of length N is $(N + 1)\delta(k) - 1$, where $\delta(k)$ is the Kronecker delta. It was also claimed in lecture that the maximum cross-correlation value is lower-bounded as follows, where M is the number of m-sequences considered:

$$\max_k R_{ab}(k) \geq N\sqrt{\frac{M-1}{MN-1}} \approx \sqrt{N}$$

Verify these two claims for 6 different m-sequences (different connection vectors) generated by an LFSR with $n = 10$ stages (sequence period $N = 2^n - 1 = 1,023$). On the last page of this problem set you will find the connection indices for maximal length sequences for $n = 10$ (reference: `www.newwaveinstruments.com`). You will find the MATLAB function `ccorr.m`, which performs circular correlation, convenient for calculating the auto- and cross-correlation functions of periodic sequences.

(c) To generate a continuous-time spreading signal $x(t)$ from a ± 1-valued pseudorandom sequence, we multiply each element a'_i in the sequence by the support function $p(t)$, appropriately shifted, and sum the result. Thus,

$$x(t) = \sum_{i=-\infty}^{\infty} a'_i p(t - iT_c)$$

where T_c is the chip interval. In computer code, we represent $x(t)$ by a series of discrete samples whose sampling interval is small compared to T_c. The operation of converting the sequence $\{a'_i\}$ (which can already be thought of as a roughly-sampled version of $x(t)$) to an unambiguous representation of $x(t)$ is called *oversampling*. Read through the MATLAB function `oversampleSpreadingCode.m`, available on iLearn, to understand what this operation entails. Examine the auto- and cross-correlation functions $R_X(\tau_i)$ and $R_{X_1,X_2}(\tau_i)$ corresponding to an independent pair of oversampled spreading codes derived from two pseudorandom binary sequences of period $N = 1,023$ generated by each of the following methods:

(a) Via the MATLAB `rand` function.

(b) A mapping of digits of the transcendental number π.

(c) A 10-stage maximal length LFSR.

You will find that the script `correlationExperiments.m` is already set up to do this. For each pseudorandom source, find the ratio of the maximum of $R_{X_1}(\tau_i)$ to the maximum of $R_{X_1,X_2}(\tau_i)$ over all values of τ_i within one period. Which of the three pseudorandom sources appears to produce codes with the best auto-correlation properties? Which produces codes with the best cross-correlation properties?

Problem 6: Verify through a MATLAB experiment that the power spectral density of a random binary sequence is given by $S_X(f) = T_c \, \text{sinc}^2(fT_c)$. Generate a long ± 1-valued sequence in MATLAB (say, $N = 2^{14}$ elements or so). Then re-sample this sequence with a sampling interval that is a small fraction of your assumed chip interval T_c. This process is called *oversampling*. To ensure good auto- and cross-correlation properties, the sampling rate should not be an integer multiple of the chipping rate (can you explain why this is so?). Then use the `pwelch` function to estimate the power spectrum of the oversampled sequence. Plot the result of the `pwelch` function on a dB scale. Zoom in to see just the main lobe and the first few sidelobes. What is the expected height of the main lobe in dBW/Hz? Does this match your plot? What is the effect of using the option `[...] = PWELCH(...,'twosided')`? Experiment with different values for (1) the original binary sequence length, (2) the oversampling interval, (3) the `pwelch NFFT` parameter. Describe the trends you see along each of these experimental axes. You may use the MATLAB function defined in `oversampleSpreadingCode.m` (found on iLearn) for oversampling.

Now, consider a *periodic* binary sequence with period T_p. The power spectrum does not follow a continuous $T_c \, \text{sinc}^2(fT_c)$ curve. Instead, power in the frequency domain is concentrated at intervals of $1/T_p$ Hz. Explain why this is so. Examine the power spectra of two different oversampled m-sequences of period $N = 1,023$.

Connection indices for maximal length sequences for $n = 10$ (reference: `www.newwaveinstruments.com`):

```
10 stages, 2 taps:  (1 set)

[10, 7]


10 stages, 4 taps:  (10 sets)

[10, 9, 8, 5]
[10, 9, 7, 6]
[10, 9, 7, 3]
[10, 9, 6, 1]
[10, 9, 5, 2]
[10, 9, 4, 2]
[10, 8, 7, 5]
[10, 8, 7, 2]
[10, 8, 5, 4]
[10, 8, 4, 3]


10 stages, 6 taps:  (14 sets)

[10, 9, 8, 7, 5, 4]
[10, 9, 8, 7, 4, 1]
[10, 9, 8, 7, 3, 2]
[10, 9, 8, 6, 5, 1]
[10, 9, 8, 6, 4, 3]
[10, 9, 8, 6, 4, 2]
[10, 9, 8, 6, 3, 2]
[10, 9, 8, 6, 2, 1]
[10, 9, 8, 5, 4, 3]
[10, 9, 8, 4, 3, 2]
[10, 9, 7, 6, 4, 1]
[10, 9, 7, 5, 4, 2]
[10, 9, 6, 5, 4, 3]
[10, 8, 7, 6, 5, 2]


10 stages, 8 taps:  (5 sets)

[10, 9, 8, 7, 6, 5, 4, 3]
[10, 9, 8, 7, 6, 5, 4, 1]
[10, 9, 8, 7, 6, 4, 3, 1]
[10, 9, 8, 6, 5, 4, 3, 2]
[10, 9, 7, 6, 5, 4, 3, 2]
```