EE 419 - Project 5
The Hunt for Red October

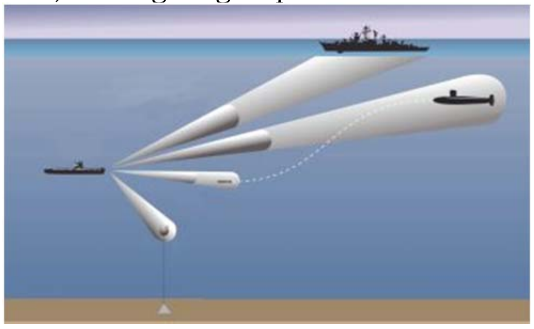
Learning Objectives:
Use Matlab to process discrete-time signals, including:

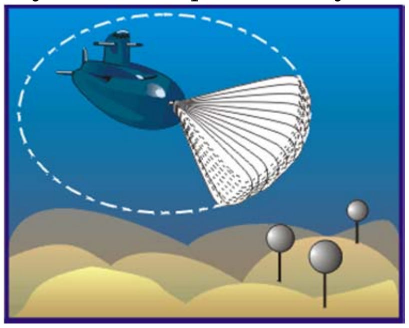
- **Correlation Signal Detection in the Presence of Noise**
- **Template Pattern Matching by Correlation Peak Amplitude**
- **Waveform Envelop from Hilbert Transform**
- **Time-of-Flight Determination from Correlation Peak Position**

REMINDERS:
1) When you are finished, be sure to always copy your m-files to a place that you can retrieve them again in the future (copy them to a memory stick, email them to yourself, etc.).
2) Please delete your files from the Lab Computer (so that everyone has to work as hard as you have!)

You are assigned as the Chief Sonar Technician aboard the newly commissioned US Navy SLO-class ultra-stealth, fuel-cell/fusion-reactor hybrid submarine: *USS CalPoly*, on her maiden voyage, with orders to locate and track the Russian navy's new submarine flagship, the Typhoon-class ballistic missile sub: *TK-21 Red October*.

Your particular job is to maintain and monitor the *CalPoly's* state-of-the-art underwater sonar tracking system. Using an array of sonar transducers located around the perimeter of the ship, you can "see" what is around the vessel at all times; and provide vital information for safe navigation and, in times of battle, for targeting torpedoes when threatened by hostile ships or enemy submarines.





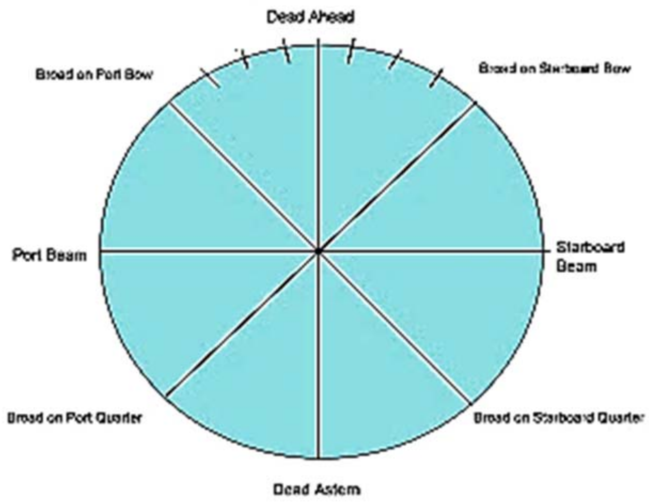
As you are searching the Laurentian Abyss off the eastern coast of Canada, **the sonar system picks up a number of different strong echoes from multiple directions** indicating numerous objects very close to the *CalPoly*. Just as you make the needed adjustments to engage the automated computer analysis that should identify the probable sources of the echoes, so that you can tell if they are friends, foes, whales, noise, or an imminent collision with a rock ledge.....the sonar computer reboots!!! *(Another #$%&@# automatic Windows security upgrade!)* The restart will take at least 3 hours (since the computers were confiscated from the EE circuits labs at CalPoly State University); and so in the mean time YOU need to analyze the sonar data files to see if you are in danger or not!

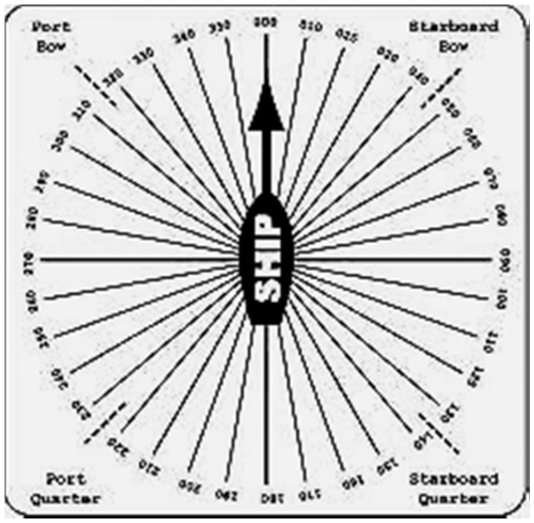
So, you grab your trusty laptop computer (the one with the free version of Matlab still loaded on it from your days in EE 459), hack in to the sonar data buffer in the ship's main computer (after breaking through the encryption codes and security clearances using the Captain's log-in), and set to work....

Revised: 2/7/17

What you have available to you to work with are sonar signal samples from the sonar data buffer, including: (NOTE: These sampled data records are available on PolyLearn in the *SubSonar.mat* file.)

- **Sonar echos** (2 seconds of data sampled at 50 KHz) collected from each of 8 directions: Dead Ahead (relative bearing: 0 degrees), Starboard Bow (relative bearing: 45 degrees), Starboard Beam (relative bearing: 90 degrees), Starboard Quarter (135 degrees), Dead Astern (180 degrees), Port Quarter (225 degrees), Port Beam (270 degrees), and Port Bow (315 degrees)
    - These sampled echo recordings started just after the sonar pulse was transmitted, and continued for 2.0 seconds.





Shipboard Relative Bearings and Direction Names

- Example data files (50 milliseconds long, sampled at 50 KHz) of:
    - The **sonar transmit pulse** (the waveform sent out by the sonar transducer) – the sonar "ping" sound that passes through the sea water and bounces back if there is something other than water to reflect it back. (A ≈10 millisecond long, Gaussian envelop, chirp signal linearly ramping its frequency between 3 KHz – 6 KHz, called *TxPulse*)
    - **Example sonar echo signals** (the waveforms that would be received back if the sonar pulse bounced off different objects) for the following underwater targets:
        - Russian Typhoon-Class Ballistic Missile Submarine (*TyphoonSubEcho*)
        - Russian Akula-Class Attack Submarine (*AkulaSubEcho*)
        - US Los Angeles Class Attack Submarine (*LosAngelesSubEcho*)

      [Note: These echo signals would not look exactly like the transmit pulse, as there may be several different surfaces on the object for the pulse to reflect off of (creating multiple echoes with different delays that are added together), and different reflection coefficients vs. frequency (which will alter the spectral energy distribution of the echo signal compared to the transmitted signal). Therefore, each kind of reflecting object will have a unique echo "signature", that comes from a convolution of the transmit pulse shape with an impulse response that results from the reflecting object's shape, hardness, size, and material composition.]

      Other smaller objects in the water (fish, dolphins, small rocks) that are much smaller than another submarine will act like point reflectors (with an impulse response that is like a single, weak impulse at their location). Therefore, reflections from them will generally have the same shape as the transmitted pulse, but be much lower in amplitude than the echoes from a large, hard object.

      Echoes received from solid rock formations (like nearby rocks or walls) will be very strong reflections that also closely resemble the TX Pulse shape.

1) Evaluate The Available Data

- **Using Matlab, observe the waveforms and frequency spectra** of the sonar transmit pulse, and the three example target echo signals**.**
  - **a.** Create a single figure that shows the time-domain waveforms for all four example signals (one per subplot). Be sure to properly label all axes and title each plot. Use an appropriate time axis for the x-axis of each (0 – 50 milliseconds).
  - **b.** Create another single figure that shows the DFT frequency spectrum (magnitude only) for all four signals (one per subplot). Be sure to properly label all axes and title each plot. Use an appropriate **analog frequency axis** for the x-axis of each.
  - c. Listen to each of the examples using Matlab's `soundsc()` function. (Check the Matlab help file [`> help soundsc`] for how to use this function if you do not know.) Hint: Since the sonar pulses use a fairly high range of frequencies, you can downshift their frequency into a more audible range by using the Matlab default sampling rate of 8192 Hz for playing back and listening to the sonar sounds using soundsc( ), instead of the actual 50 KHz sampling rate in the data. This will shift the 3-6 KHz sonar sounds to a more audible frequency range of 490 – 980 Hz.

  Turn in:
  - Copies of your 2 figures.
  - Matlab code that created the figures.
  - Describe the differences that you see between the waveforms and spectra of the different targets.
  - Which submarine has the "most stealthy" sonar signature (most difficult to distinguish from a rock)? Justify your choice.

- **Using Matlab, observe (plot and listen to) the Sonar Echo waveforms from the 8 different directions**, without any additional signal processing.

  Turn in:
  - Copies of at least 2 examples of echo waveform plots from two different directions.
  - Matlab code that created the examples.
  - Describe the waveforms (both plotted data shapes and sounds), and whether or not you are able to visually spot or audibly hear any of the expected target echo waveforms in any of these longer sonar echo waveforms. If so, identify which target signal you are able to find (if you can), and which sonar direction it is found in.

2) Create a Correlation Detection Function

In order to find possible target echoes among the noise of the received sonar echoes from each direction, you will need to use Correlation Detection – that is cross-correlating the received echo signals (including noise) with the waveform you would like to detect. Because the cross correlation of two high amplitude signals will tend to give big number values, one cannot determine the similarity of two signals just by observing the absolute amplitude of the cross correlation peak.

In order to make the correlation values more meaningful, it is typical to normalize them by a measure of the energy in each signal. The normalized cross-correlation function for two discrete-time signals x[k] and y[k] is usually defined as :

$$C_{xy}[k] = \frac{r_{xy}[k]}{\sqrt{E_x E_y}} = \frac{r_{xy}[k]}{\sqrt{r_{xx}[0]\, r_{yy}[0]}}$$

Where: $r_{xy}[k]$ = unnormalized Cross correlation between signals x, y
E = energy in each signal = sum of the squared signal values
$r_{xx}[0]$ = Auto-correlation of signal x at the 0 lag position = sum of the squared signal values

The normalized cross-correlation function should have values bounded between -1 to +1, with +1 indicating that the two signals have identical shapes and are properly aligned. Note that for noisy discrete-time signals, the normalized correlation function will generally not reach +/- 1.0.

Create a Matlab function m-file to compute the normalized Cross-Correlation between two discrete-time signals.

```
function Cxy = NormCrossCorrelate(xn, yn)
```

Functional Requirements:

- Your program should properly handle any length signals xn and yn.
- To run as fast as possible, you function should compute the cross-correlation using FFT's (and zero padding to ensure linear cross-correlation (instead of circular cross-correlation) and to make the FFTs as fast as possible - similarly to how you carried out fast convolution in a previous lab**). Do NOT use the Matlab xcorr( ) function!**
- You do not need to break up long sequences into shorter blocks using overlap-add or overlap-save block processing. Simply cross-correlate by processing both full signals at once.
- Be sure to normalize the cross-correlation results as shown in the equation above.
- Be sure to remove any extra length in your Cxy output from zero padding.
- Your function should plot the two input signals (xn, yn) and the Cxy results vs. index (3 subplots) in a single figure.

Testing:
To understand how your function positions the cross-correlation result, try performing an autocorrelation (cross correlate a signal like the TxPulse with itself); and observer where the zero lag autocorrelation peak occurs. This should give you insight into how to assign a lag time to each sample of the cross-correlation result.

For your report, verify your function using the XCorrTest and TxPulse arrays from the PolyLearn SubSonar.mat file. Be sure that you correlate TxPulse against XCorrTest, not the other way around:

NormCrossCorrelate(XCorrTest, TxPulse)

Turn in:

1. **Explain your selection for the FFT size** used in your FFT signal processing: Why did you use the particular value chosen?
2. **Matlab function code listing** (with appropriate comments)
3. **Test case results**: provide the plot created by your function for the specified XCorrTest/TxPulse test case, and interpret the results: What was the peak cross-correlation value? Where did it occur? Was this what you expected?

3) Who's Out There???

Using your cross-correlation function, evaluate each of the Sonar Echo data files from the 8 scanning directions.

Determine the following:

1. **Is there a submarine present in that scanning direction?**
2. **If there is a submarine present,**
    a. **How far away is it?**
    b. **What kind of submarine is it?**
    c. **Can you hear it if you listen to the cross-correlation result?**

Some hints:

1. When searching for cross-correlation peaks of two signals that are sinusoidal (so that the cross-correlation function is also somewhat sinusoidal) it is very helpful to find the **analytic envelope** of the signal. This envelope is similar to the shape you would get if you full-wave rectified and low-pass filtered the signal. Finding the peak of the envelop is usually a more precise measurement of the cross-correlation function peak position than simply finding the peak of the absolute value of the cross-correlation function.

A very simple way to compute the envelope of a sinusoidal signal is using the Hilbert transform. (Don't panic – it's just another Matlab function). If you have a bipolar (positive and negative valued) signal xn, you can find its positive valued envelope using:

```
xn_envelope = abs(hilbert(xn));
```

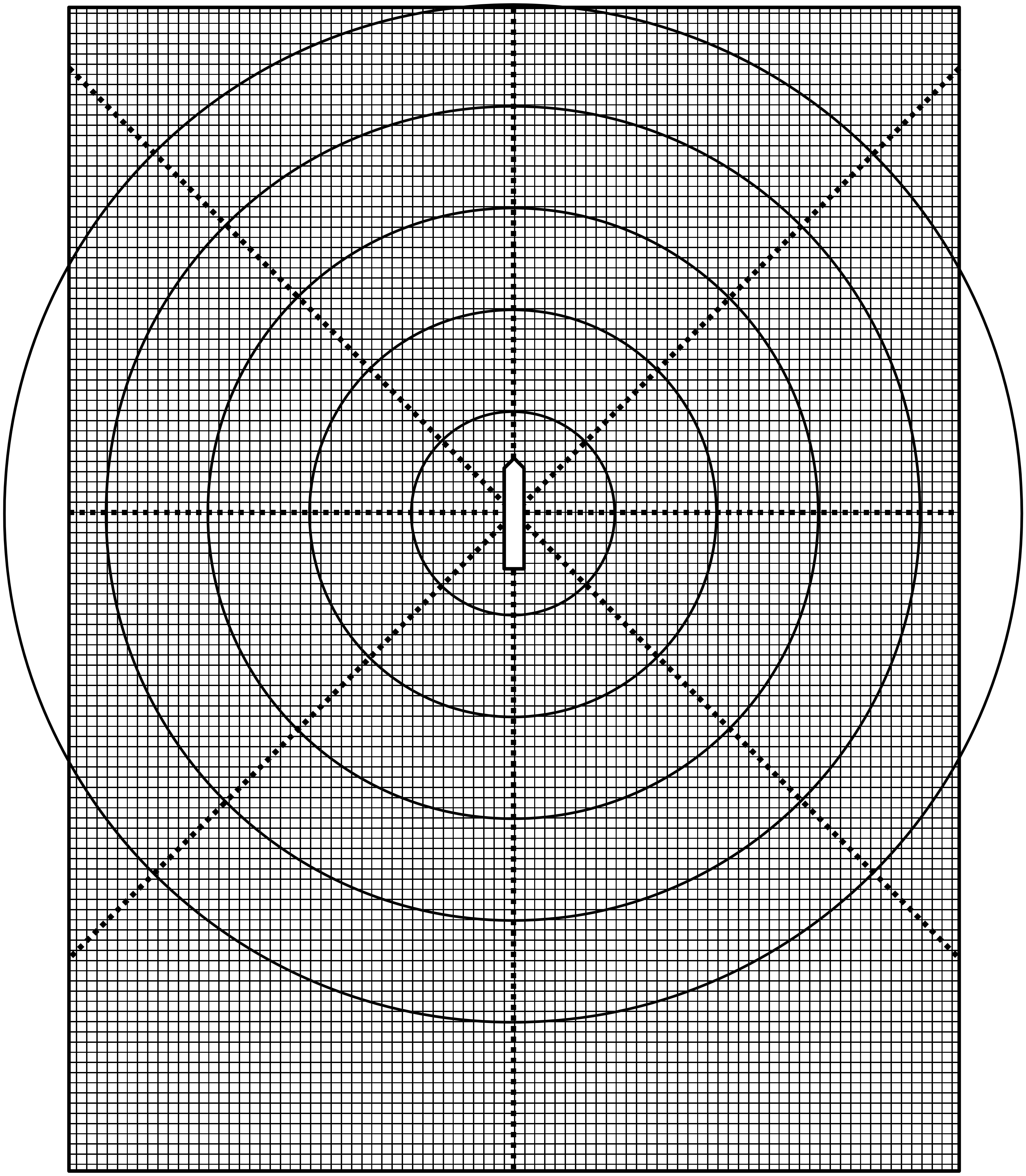
2. You will need to know the speed of sound in sea water to predict the target distance(s). You can Google it, of course. Don't forget that you have to wait for the transmitted pulse to travel from your sub to the target, and then the echo needs to travel back to your sub again before it shows up in the received echo waveform.

Turn in:

1. **Matlab code listing** (with appropriate comments)
2. **Test case results**:
   a. Complete the Table below with your findings from analyzing the eight sonar echoes. (If the particular target is not found in that sonar echo data set, just leave that box blank in the table.)
   b. Provide plots that support your conclusions for the distances and the type of target present. (Show only those with a submarine or big rock/wall present).
   c. Describe what you hear when you listen to the cross-correlation results when there is a target present (both what the target sounds like, and what you hear during the rest of the signal).

Direction	Relative Bearing (degrees)	Distance to the Target Found (meters)			
		Russian Typhoon-Class Ballistic Missile Submarine	Russian Akula-Class Attack Submarine	US Los Angeles Class Attack Submarine	Big Rock or Wall
Forward	**0**				
Starboard Bow	**45**				
Starboard Beam	**90**				
Starboard Quarter	**135**				
Aft	**180**				
Port Quarter	**225**				
Port Beam	**270**				
Port Bow	**315**				

3. **Draw a diagram (to scale)** showing the target positions relative to the SS CalPoly**. Indicate the position** of any valid target **on the following grid**. **Identify the target** (what type of sub, or rock/wall), and write in the numeric value of the **distance to the object**.



Revised: 2/7/17