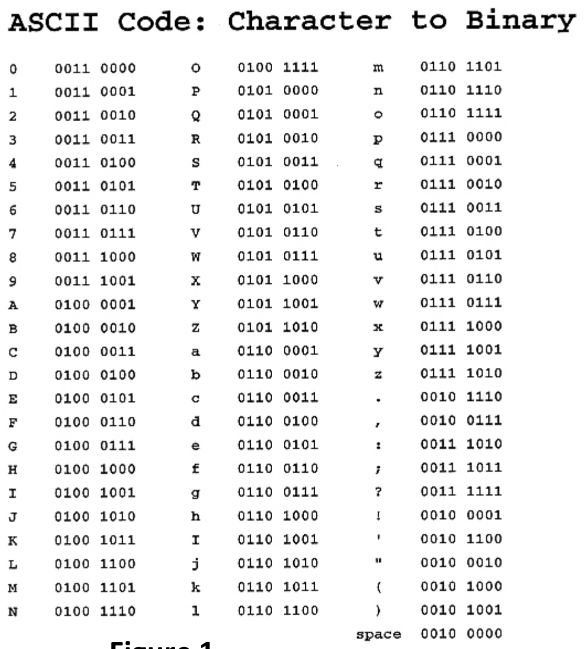
Spring 2022 / 520.214 – Signals & Systems
Project 1
Due: 3/30/2023, 11:59pm

Sonar digital communication

Background

In modern digital communication systems, information (e.g. speech, video and computer data) is encoded as sequences of binary ones and zeros prior to transmission. Any message can be converted into a sequence of 0’s and 1's (Figure 1).

Transmitter:
Once a binary code for the message is generated, these digital bits are converted to a continuous-time signal for transmission purposes. For example, a 1 may be represented as a positive rectangular pulse, and a 0 as a negative rectangular pulse, each of duration T seconds (Figure 2A). The transmitted signal is a concatenation of these 0's and 1's, as shown in Figure2B. Figure 2C shows other examples for pulse shapes. Once a shape is chosen, a message can be constructed by concatenating the 0’s and 1’s in the right order.



ASCII Code: Character to Binary

0	0011 0000	O	0100 1111	m	0110 1101
1	0011 0001	P	0101 0000	n	0110 1110
2	0011 0010	Q	0101 0001	o	0110 1111
3	0011 0011	R	0101 0010	p	0111 0000
4	0011 0100	S	0101 0011	q	0111 0001
5	0011 0101	T	0101 0100	r	0111 0010
6	0011 0110	U	0101 0101	s	0111 0011
7	0011 0111	V	0101 0110	t	0111 0100
8	0011 1000	W	0101 0111	u	0111 0101
9	0011 1001	X	0101 1000	v	0111 0110
A	0100 0001	Y	0101 1001	w	0111 0111
B	0100 0010	Z	0101 1010	x	0111 1000
C	0100 0011	a	0110 0001	y	0111 1001
D	0100 0100	b	0110 0010	z	0111 1010
E	0100 0101	c	0110 0011	.	0010 1110
F	0100 0110	d	0110 0100	,	0010 0111
G	0100 0111	e	0110 0101	:	0011 1010
H	0100 1000	f	0110 0110	;	0011 1011
I	0100 1001	g	0110 0111	?	0011 1111
J	0100 1010	h	0110 1000	!	0010 0001
K	0100 1011	I	0110 1001	'	0010 1100
L	0100 1100	j	0110 1010	"	0010 0010
M	0100 1101	k	0110 1011	(	0010 1000
N	0100 1110	l	0110 1100	)	0010 1001
				space	0010 0000

Figure 1

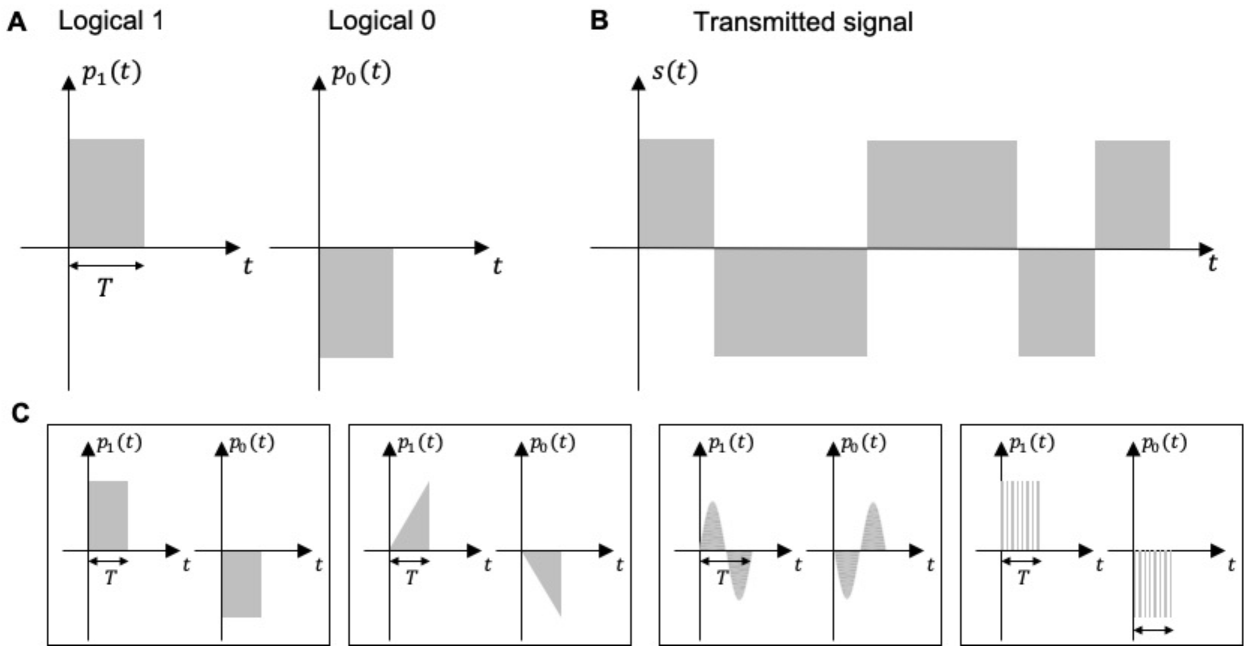


Figure 2

Receiver:
Often, noise corrupts the signal during transmission. Noise may arise from many sources (interference from other transmitters, inclement weather, obstacles, etc). The job of a receiver is to accurately discern what information was transmitted even with the presence of noise. The advantage of a binary communication system is that the receiver needs to only determine whether a logical 1 or 0 was sent, rather than to correctly identify all values of the original signal.

Matched filters:
The problem of recovering digital signals corrupted with noise has many solutions. One popular solution is using what is called a matched filter. This receiver is the best solution for recovering digital data encoded using a known waveform, in the presence of *additive* random noise. A matched filter operates as outlined in the system in Figure 3.

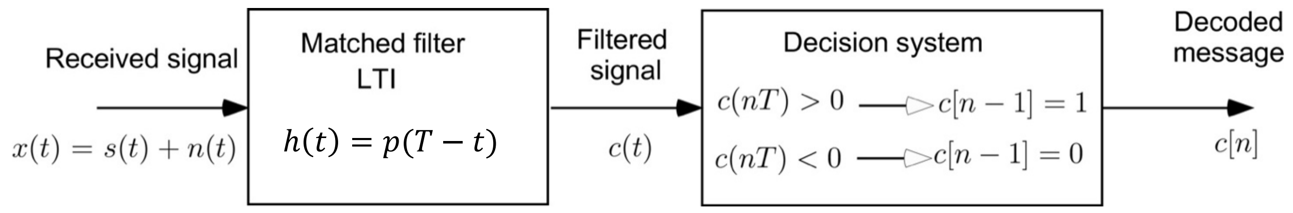


Figure 3

First, the communication system <u>agrees</u> on a choice of a canonical shape $p_i(t)$ for the logical bit 1/0 (one of the options in Fig. 2C). Next, the transmitter generates the signal using this pulse and transmits signal $s(t)$. The receiver detects a different signal $x(t) = s(t) + n(t)$ which is a noisy version of the transmitted signal $s(t)$, where $n(t)$ is an unknown background noise. Naturally, the receiver does not know the message $s(t)$ and needs to recover it from $x(t)$. That's where a matched filter system comes in.

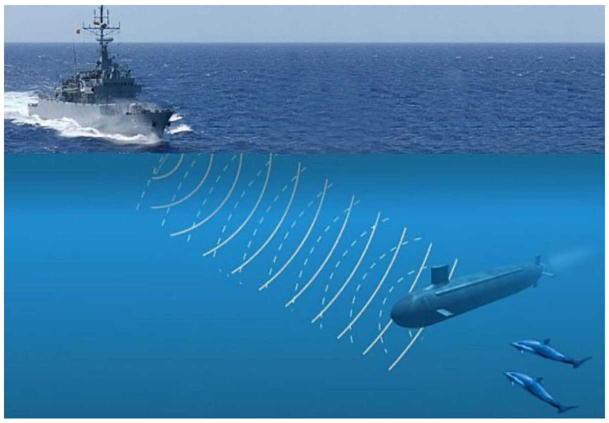
How does a matched filter receiver work? The receiver operates as a linear time-invariant system with impulse response $h(t)$ as a flipped version of the expected logical 1, shifted by the bit period T. This shift ensures that the system is causal (i.e. $h(t) = 0, t < 0$). At time T, the filtering operation achieves its maximum value if the input to the system is a 1, at time T. Looking at the output signal, we can decide whether the incoming input matches a 1 or not. This matched operation is repeated every T seconds to check for a match between the received signal and a logical 1. This LTI system is called a *matched filter*.

Project assignment

This project explores the concept of binary codes, sonar communication and matched filter systems. The project consists of 3 parts.

1. The project is an *individual* assignment. You can discuss ideas with classmates but make sure you write your own code and report.
2. Please submit all your code AND a report where you describe your approach, your design decisions, any theoretical formulations, as well as show any graphics (plots) and analyses of your code. Do NOT include code as part of your report.
3. Your submission should include your MATLAB function **decode.m** for part 2 and **transmit_noise.m** for part 3 (details below), along with your **report** (in .doc or .pdf format – max 10 pages with plots). The code submitted should be fully debugged and only use built-in MATLAB functions (no third-party software or special MATLAB toolboxes).
4. Submit your report and code on canvas.

Part 1 – Sonar systems



How can a matched filter be used in active sonar? Assume you are a sonar operator on a submarine and are told to find where a friendly ship is with active sonar. Active sonar operates by sending a 'ping' signal and listening for the return echoes from objects. Your ping signal is the transmitted signal $z(t)$. Your sonar system has a receiver that scans for return (echo) signals $e(t)$. If there is an object in your path, this return signal will consist of your ping signal embedded in noise and delayed by the time it took to travel to the object and echo back. Your job is to detect this returned ping using a matched filter system.

You sent a sinusoidal ping signal of duration 0.3sec (SonarPing); and you received a returned echo signal (SonarEcho). Both signals are given to you in the file ActiveSonar.mat, and both signals are sampled at 100 samples every second. Assuming sound travels 5000 feet per second in water, the object's distance can be calculated by measuring the time it took for the echo to return.

1.1 Load the file ActiveSonar.mat

1.2 Determine how far away the friendly ship is by using a matched filter receiver on the echo (see background section). Describe all your steps and include plots of the received signal and the filtered signal to explain your answer. Make sure all axes are labeled properly.

Part 2: Digital Message Reception

The friendly ship you detected in Part 1 now wants to send you a message and you both have an agreement to use the *triangular wave pulses* with positive slope for 1 and negative slope for 0 (Figure 2C, second example – finite ramp signal) with duration 0.5 sec. Again, there is a great deal of noise in the water making reception of the entire signal potentially difficult.

The received signal is stored in file ReceivedSignal*.mat and all signals are sampled at 100 samples/sec.

2.1 Using a matched filter, determine what binary message stream was sent.

2.2 Decode the message into readable characters using a lookup table (Figure 1). It is provided in file ascii.code, which you can read in MATLAB using the function *fread* or *textscan*. What message was sent?

NOTE: Write your code so it can be used for any new Received Signal, not just the one given in this example. Submit a function **decode.m** with the following arguments:

msg = decode(sig_received,fs,pulse)

where sig_received is the actual signal, fs is the sampling rate (samples/sec) and pulse is an optional input argument representing the logical 1/0 pulse signal (The default pulse is a square wave with duration 0.3sec). The function output is msg which is a string.

Part 3: Digital Message Transmission

Now, you want to send a message back to the friendly ship. Use the lookup table provided in file ascii.code to put together a message of your choice. It should have a minimum of 5 characters. Use the same triangular pulse waveform used in Part 2 to represent your message as a transmitted signal s(t). A sampling rate of 100 samples/second is again assumed. To appreciate how noise can corrupt this message, generate a noisy version of your signal by adding a noise signal of the same length. Use the function *randn* to add noise to your signal. The sum of your clean signal s(t) and the noise interference n(t) is now the new received signal x(t).

3.1. Show a plot of your original signal s(t) and its noisy version x(t). Make sure you label your axes correctly.

3.2. Use the receiver function from Part 2 to decode the message. Are you able to recover your entire message?

Both you and the friendly ship decide to 'test the water' by exploring the best choice of pulse signal for effective communication (i.e. which signal in Figure 2C) works best. You will generate a message and convert it into a signal s(t). Then you will add variable levels of noise to it: $x(t) = s(t) + a.n(t)$ where a is a scalar that increases the amount of noise in the received signal. As a gets bigger, the receiver will make mistakes about what the transmitted signal was until it is unable to reliable tell the signal anymore.

3.3. For each choice of pulse signal (Figure 2C), gradually increase the noise level (a) and test how well the receiver decodes the message. Submit the code (**transmit_noise.m**) you used to generate and test each of the pulse signals in Figure 2C. Which *pulse signal* is most robust to noise (i.e. can be decoded at higher levels of noise)?

Relevant MATLAB functions

help or *doc*	Find out more details about each function (e.g. type help conv)
conv	Convolve two signals. Caution is needed using this function: arguments need to be in a specific format, study the description in the help page.
Load	Load a variable into MATLAB from .mat file. You will need this for loading the given signal files into MATLAB
textscan	Read text file into MATLAB variable. See documentation for details. You can use this to read the ascii codebook. Alternatively, you can use *fread.*
Fliplr	Flip a horizontal vector. Caution is needed using this function: arguments need to be in a specific format, study the description in the help page.
Randn	Generate a noise signal (Gaussian noise) of any desired length
plot	Display a signal. Carefully adjust the parameters in order to properly label x and y axes

Expectations – Rubrics for grading

1. Please include any specific figures or evaluation metrics asked for in the submitted report.
2. Ensure the functions are named as indicated above and they support all the requirements of the given input format(defaults).
3. Meet all the specifications given for your function (it will be fully tested). If your code generates an error, you will be loose points.
4. Please explain how you did your analysis instead of just mentioning the issues in your report. Try to address each of the questions mentioned.
5. Major inconsistencies in implementations and their descriptions will be penalized.

The following sections will be carefully graded in your final report and code:

Part 1

- Report
  - Description of matched filter
  - How to find echo in received signal
  - Plot of received signal with correct axes
  - Plot of filtered signal with correct axes
  - Estimate of ship distance based on echo results
- Code
  - No need to submit your code for this section (your report should describe all steps)

Part 2

- Report:
  - Description of matched filter operation (convolution, bit-decoding)
  - Description of SCII-decoding
  - Correct decoded messages
- Code:
  - Runs without error
  - Proper function name and inputs
  - Bit-decoding works
  - ASCII-decoding works
  - Correct output on test

Part 3

- Report:
  - Description of ASCII-to-bits
  - Description of bits-to-signal
  - Description of testing at different SNR
  - Description of test results
  - Plots of original and noisy signals with correct axes
- Code:
  - Runs without error
  - Proper function name and inputs
  - Generates proper outputs