UNIVERSITY OF UTAH
ELECTRICAL AND COMPUTER ENGINEERING DEPARTMENT

ECE1020 **COMPUTING ASSIGNMENT 7**
N. E. COTTER **MATLAB® CONTROL FLOW: FILTERING**

READING
Matlab® Student Version: learning Matlab 6, Ch 6-2 to 6-6
Mastering Matlab® 6, Ch 10

TOPICS
Control flow: For, while, if, switch

OVERVIEW
In previous assignments, we have considered all of the parts from which we could construct a communication link between a base transmitter and a receiver:
- translating commands from a control language, (expressed in strings such as 'right' '1'), into a sequence of binary numbers,
- creating a codebook for translating binary information into longer binary strings (allowing us to perform error correction later on),
- transmitting sinusoids of two different frequencies representing 0's and 1's in the codewords,
- using the radiation pattern for an antenna to determine the power of the signal at the receiver (as a function of distance and direction),
- using correlation and summation (i.e., a matched filter) to determine whether each received bit signal (plus noise) represents a 0 or 1,
- calculating the distance between the received binary bits and each codeword to find the best match,
- translating the index of the codeword into its corresponding bit pattern (i.e., the binary information we had at the very beginning of this process), and
- interpreting the bit pattern as a command to be carried out (such "right 2").

We are now ready to simulate an entire system (with some simplification) in a fashion that allows us to process an arbitrarily long sequence of input commands.

PROCEDURE
In this assignment, you will use for, while, if, and switch statements to write a complete communication system simulator.

+5 pts Script File for calculations
Using a text editor program on your PC, create a script file called **control_flow.m** containing matlab commands to perform the calculations in this assignment.

+5 pts Do **not** use semicolons at the ends of commands in your script files.

+5 pts Create a string variable (having eight rows) called command_str containing the commands 'right', '0', 'right', '1', 'left', '1', 'right', and '0'.

+5 pts Use a for loop to interpret the command strings:
- processing the strings (i.e., 'right', 'left', '0', or '1') in order, and (continued in next item)

+10 pts
- translating the strings (using a switch statement) into numbers (as opposed to binary codes used earlier):
  - 0 = 0
  - 1 = 1
  - left = 2
  - right = 3

+5 pts
- Concatenate the eight command numbers to create one horizontal array (of 8 numbers) called command_num. Then end the loop.

+10 pts Create a 4x6 array called codewords that consists of four 6-bit codewords. You may create the codewords in any way you desire, including code you wrote for earlier assignments.

+10 pts Use a for loop to translate the numbers from command_num into the corresponding codewords. In other words, if the number is 0, use the first codeword; if the number is 1, use the second codeword, and etc. (Remember that Matlab indexing is one-based.) Concatenate the codewords to create a binary array (with only one row) called command_code (containing 48 bits for the 8 codewords for the 8 command numbers for the original 8 commands). These are the encoded bits to be transmitted to the rover.

+5 pts Rather than translating the bits in command_code into cosine waveforms and using the antenna radiation calculations from earlier assignments, we will simply create an array indicating which bits are received incorrectly. Using the randn function and a logical operator, create an array called "noise" consisting of 48 zero and one values. The value should be zero when the entry in the array returned by randn is less than 1.5. The 1's in "noise" will correspond the erroneous bits in the received code words.

+5 pts To flip the erroneous bits to the wrong value, compute the exclusive-or of the command_code and noise arrays and place the result in an array called command_recv. (On average the noise will change only a few bits.)

+5 pts In conjunction with the following steps, use an outer "for" loop to process the command_recv array one codeword at a time until the end of the array is reached. The goal is to decode the incoming commands.

+5 pts The first operation you will perform in the loop is to extract the next received codeword from command_recv and place it in an array called codeword_recv. In other words, extract 6 bits from the front end of command_recv and place them in

codeword_recv. Don't forget to remove these six bits from command_recv for the next time through the for loop, but check to see if the array is already empty first so you don't generate an error.

+10 pts Using Hamming distance, (the number of bits that are not equal for two binary numbers), determine which codeword is closest to codeword_recv. Append the corresponding command number, (e.g., 2 if the codeword corresponds to the command 'left'), to an array called command_dec. When the loop is finished, command_dec will contain our sequence of decoded commands.

+10 pts Using a "switch" command, translate the command number in command_dec into the corresponding command string, (e.g., 'left'), and add that string as another row to a string array called command_str_recv. If we are lucky, command_str_recv will be the same as the command_str that we started. (With the level of noise we are using, however, you will get one error approximately half the time.) Where we have decoding errors, our rover will turn the wrong direction or go the wrong distance.

+5 pts Run Script File
Run your script file by typing the name of the file without the .m
>> control_flow
Use a diary file to capture the output of control_flow.

If you make any changes in your **control_flow.m** file, be sure to run the following Matlab command to insure that Matlab reads your file again the next time you run it:
>> clear all

E-mail your script file (control_flow.m) and your diary file to your TA, (as two separate e-mails). In the Subject line of your e-mail, be sure to put Your Name, "ECE1020 Comp7," and the file name, (e.g. control_flow.m). Also, print out the files and hand them in to the TA or to the ECE1020 locker.