E2-243
Programming Exercise - 7

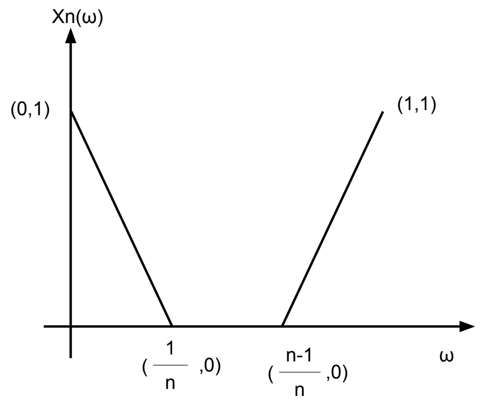
TA: Zitha Sasindran, Prerna Arote

Instructions:

- Do not submit your code, output files, etc.
- There will one lab exam towards the end of the semester that will test your understanding of the concepts taught in class. The questions in the lab exam will be somewhat similar to these questions in both content and implementation complexity. If you do not program these exercises, handling the lab exam will not be easy! In a way, programming these assignments yourself will be your preparation for the lab exam.
- You may use any discrete plot like the 'stem' function in MATLAB.
- Whenever $x \in$ interval, you may have to take appropriate discrete points in the interval to realize the functions in matlab. Please use appropriate commands for continuous plots when x is continuous.

---

1. Let X_n be the sequence of real valued random variables defined on the sample space Ω = [0,1], (The event space is the Borel sets and the probability of an interval is its length), defined as shown in the figure below:



   (a) In the program, create an array of values of $w \in [0, 1]$ with appropriate resolution. Write a function **Xn(w,n)** which will plot X_n Vs w for different values of n. Call the function **Xn()** from your main program and plot X_n Vs. w for n=2, 3, 10 and 100. All the plots for different values of n should be visible in the first subplot window after the execution. (You could use *hold on*).

(b) Clearly $x = X_n(w) \in [0, 1]$. In the main program create an array of $x \in [0, 1]$ with appropriate resolution. Write a function **Fx(x,n)** which will take the array x and n as inputs and plot the cumulative distribution function $F_{Xn}(x)$. Call the function **Fx(x,n)** from your main program and plot the cumulative distribution function $F_{Xn}(x)$ for n=2, 3, 10, 100. All the plots for different values of n should be visible in the second subplot window after the execution. **Check:** Note that as n increases, the probability mass at 0 increases. The CDF should change accordingly.

(c) Set the axes limits, labels and legends appropriately for all the plots. Use *subplot()* command to display 2 plots (Xn and Fx) simultaneously.

2. X and Y are jointly distributed discrete random variables. Both X as well as Y take 5 distinct integer values.

   (a) ✓ In the program initialize a 5X5 matrix **P** to define the joint distribution of X and Y. Initialize the matrix **X** (5X1) which defines the 5 distinct values of X. Initialize the matrix **Y** (5X1) which defines the 5 distinct values of Y.

   (b) ✓ Write a function **CheckDist(P)** which takes **P** as input and outputs '1' if **P** is a valid distribution and '0' if **P** is not a valid joint probability distribution. Call **CheckDist()** from your main function to validate the input **P**. The remaining execution should happen only for a valid input. Print a statement in command window stating whether **P** is a valid distribution or not. The function **CheckDist(P)** should be generic enough and should work for any dimension of P.

   (c) ✓ Write a function **Marginals(P)**. The function **Marginals()** should take **P** as input and output 2 arrays **PX** and **PY** corresponding to the marginal distribution of X and Y. Call the function **Marginals()** from the main program and obtain the marginal distributions for X and Y. Print **PX** and **PY** in the command window. The function **Marginals(P)** should be generic enough and should work for any dimension of P.

   (d) Write a function **CheckIndep()**. The function should take **P,X,Y** as input. It should call the function **Marginals()** to compute the marginal distribution. From marginal distribution and joint distribution, the function should output '1' if X and Y are independent and '0' if X and Y are not independent. Call the function **CheckIndep()** from your main program and conclude if X and Y are independent. Print a comment in the command window stating whether X and Y are independent. The function **CheckIndep (P)** should be generic enough and should work for any valid dimension of P, X and Y

   (e) CHECK: Hence after the execution of Q2b, the command window should display (a) If P is a valid distribution. (b) In case P is valid, then PX and PY. (c) In case P is valid, then comment regarding independence of X and Y.

3. **Expectation, Variance and Standard Deviation:**
Let the following 30 integers represent the score of 30 students in an exam in which the minimum possible marks was 0 and the maximum possible marks was 10.

$$6, 8, 2, 7, 8, 1, 8, 5, 4, 9, 2, 10, 6, 7, 8, 4, 6, 2, 10, 8, 9, 8, 10, 8, 2, 7, 7, 8, 4, 10.$$

   (a) ✓ Store these numbers in the array `Scores`.

   (b) ✓ Write MATLAB functions to compute the average, variance and standard deviation of these numbers.

   (c) ✓ Write a MATLAB function `Freq(array,x)` that returns the count of data elements in `array` having value `x`. Call this function with parameters `Scores`, `x` (for `x` $\in \{0, 1, \ldots, 10\}$). That is, `Freq(Scores,x)` returns the frequency of the score `x` in the array `Scores`.

   (d) ✓ Write a MATLAB function `NormFreq(array,x)` that returns the normalized frequency of the data element `x` in the array `array` (where normalization is done as follows: $\sum_{x=0}^{10}$ `NormFreq(array,x)` $= 1$). You can think of `NormFreq(Scores,x)` (for `x` $\in \{0, 1, \ldots, 10\}$) as the PMF of the integer random variable X that represents the scores of the students. In other words, $p_X(\mathrm{x}) =$ `NormFreq(Scores,x)`. Plot this normalized frequency.

   (e) ✓ Calculate the expected value, variance and standard deviation of the random variable X (defined in (d) above) using the standard formulae and compare with the average, variance and standard deviation you computed in (b) above. Also, verify your result with the output from the MATLAB built-in functions `mean()`, `var()`, `std()`.