AMS 595/DCS 525 – MATLAB Project

Due: Thursday, September 21th, 2023, 11:59 PM

NOTE:. *You must write your code on your own. Do not copy code from any source. To get full credit your codes must be well documented with comments. Submit your codes to GitHub* ***and*** *Brightspace along with brief documentation and any relevant plots. Your codes should have the correct file extensions and should run* ***without*** *any needed modifications.*

Q1. Base Converter (20 Points)

Write a script called `BaseConverter.m` that asks the user to give the following inputs:

- An integer n, where $2 \leq n \leq 10$.
- A number in base-n.
- An integer m, where $2 \leq m \leq 10$.

Your script should convert the user's number in base-n to base-m. Return this value by printing it to the console as shown in the example below. Your code should check that the user's inputs are valid and should display the correct errors if they are not.

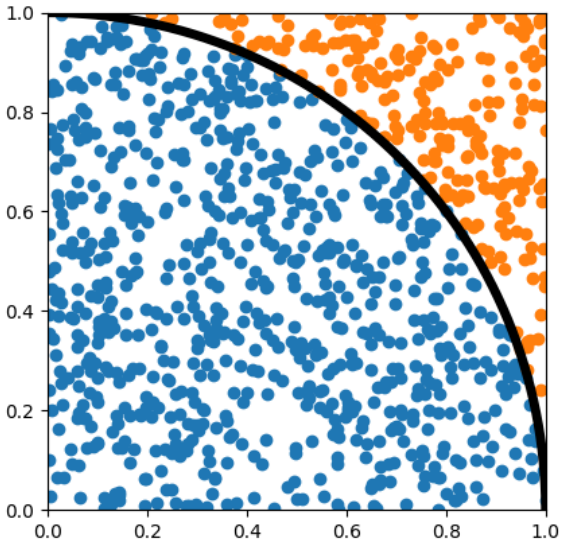
Example: $123_5 \Rightarrow 1102_3$

- Input 1: `n = 5`
- Input 2: `123`
- Input 3: `m = 3`
- Output: `Your number in base 3 is:  1102`

Q2. Gaussian Elimination (35 Points)

Write a function `GaussElim(A, b)` that takes as inputs a matrix $A \in \mathbb{R}^{n \times n}$ and a vector $b \in \mathbb{R}^n$. Using Gaussian elimination, have this function return a vector $x \in \mathbb{R}^n$ such that $Ax = b$. You may assume that the matrix A is always nonsingular. Your function should raise any relevant errors.

Q3. Approximating π (35 points)



Background: The value of π can be determined using a Monte Carlo algorithm. More specifically, randomly choosing $(x, y) \in [0, 1] \times [0, 1]$ means that all such pairs must lie within a square of unit area. The same points may or may not lie within one quarter of a circle centered at the origin with radius 1, see the figure above. The area of such a circle is, of course, $\pi/4$ – this is the probability that a given random point will lie inside the circle. We can use this to compute π by generating many random points and counting both how many lie inside the circle and how many lie inside the square (i.e. how many were generated total); the ratio of these two numbers will be an approximation to $\pi/4$. As the number of random points used in the comparison increases, so will the precision of our approximation.

Task You may find the MATLAB function `rand` helpful for the following tasks, its documentation can be found here.

(a) Write a function `MontePi(n)` that takes in an integer n, where n represents the total number of points to be used in the Monte-Carlo approximation. This function should approximate π using a `for` loop and the scheme outlined above. Your function should return three values:

1. The approximated value for π
2. The absolute error: $|\pi - \pi_{\text{approx}}|$
3. The relative error: $\frac{|\pi - \pi_{\text{approx}}|}{\pi}$.

(b) Write a script called `MontePiPlots.m` that constructs the following three plots:

1. Time the execution of your code for various values of n, and plot the resulting execution times against n.
2. Compute approximations of *pi* for various values of n, and plot the resulting approximations against n.
3. For a fixed value of n, a graphical display should plot the random points **as they are generated**, with points inside the circle plotted in a different color than those outside the circle. The final value for π should be printed on the plot.

You may use your function from Part (a) for Part (b).

Grading Scheme

– 90 points for correctness of code.

– 10 points for documentation, proper coding practices (commenting, ease of reading, etc.), and correct use of Git.