**CS 4350-5350**

**High Performance Computing with Embedded Systems**

**Process Timing Analysis**

**(300 pts.)**

Date: 9-29-2020

Due: 10-6-2020 (5 pm)

**Given:** Three independent processes and timing functions

**1. Find: (300 pts.)**

Utilizing your knowledge of timing methods in Python, analyze the computational time of three separate processes (ie. a triple nested **for loop**, calculating the **euclidean distance** of an n-dimensional vector...). Each process should have a minimum amount of complexity to make it useful for analysis. Your method should include functions for each process. You should be able to call a process from within a main function in your Python code. The ability to specify the number of times to run the process internally to calculate a mean value should be an argument you pass through. You also need to be able to specify how many times you will run this process to capture mean values for a final vector of mean value times for that particular experiment. Example ( Process\_1([input] no\_of\_trials\_for\_process, [input]no\_mean\_values\_to\_ **x\_mean\_i** = [x\_mean\_0, x\_mean\_1, ... ,x\_mean\_99]\_i'  and this will create a matrix, X\_mean = [**x\_mean\_0**, x\_mean\_1, ..., **x\_mean\_99**]  for a size X\_mean = 100 x 100. You will do this for three separate processes: Process\_1(), Process\_2, and Process\_3() to generate three data arrays: X\_mean\_P1, X\_mean\_P2, and X\_mean\_P3 . These should all be size=100x10.

Now:

**a)** Create scatter plots for your 100 experiments with 100 mean values for processes 1 through 3. You should use the subplot function to create a row of three plots for the three processes. Label these appropriately (x-axis[ trial no] and y-axis [time, in seconds]). Remember, you will have 100 data points per experiment; so, you will have 100 curves with 100 points per curve on your scatter plot for each process.

**b)**Utilizing your three process arrays, create three heat maps of the data arrays (ie. a heat map is creating an image from the data in the array. You can easily normalize each array [0 to 1] and scale by 255. Convert those values into uint8() and save the image as a grayscale image. Do this for each data array (X\_mean\_P1, X\_mean\_P2 and X\_mean\_P3). Using your subplot command plot these in a 1x3 format.

**c)** Combine your three layers in a single RGB image and display and save this image.

**d)** Create histograms for each of your grayscale images with range [0 to 255], 256 bins: plot all your histograms on a single plot with proper labels.

**e)** Calculate standard deviation and mean for each column in your original data X\_mean\_mean\_i and std\_i for i=0 to 99. Do this for all three processes. Plot your mean values (X\_mean\_mean\_i) for all three processes together and do the same for the standard deviations (std\_i). Use subplot to display these in a 1x2 array with proper labeling.

Turn in all source code, plots and data with a Word document displaying results (a) through (e).