EC 504 – Fall 2023 – Homework 7

Due interactively in class anytime. Submit figures as you get them to `/projectnb/ec504rb/students/yourname/HW7` on your SCC account. If you use gnuplot nice to provide script to reproduce figures with input data files provided as well.

GOAL: This is a short exercise to learn how to measure and fit performance to curves. Also to present the result in graphical form. On the EC504 GitHUB at Plotting_and_Fitting there is information on how to use gnuplot and even in **LittleGnuplot.pdf** instructions on how to install gnuplot in your laptop.

While you can use other graphic tools and share them in class I RECOMMEND as programmable plotting tool. See http://www.gnuplot.info. It is well work learning a few simple examples. All can be done easily using gnuplot on the SCC through if you log in with `ssh -Y username@scc1.bu.edu`. The IDE allow you save to pdf easily and view it with `xdg-open` . Below is an example using the sorting code. but others will be considered in class or as a product of your projects. Some more code to produce out data are in GitHBUT at `CodeBucket` Others maybe considered in class or as a product of your projects.

1 Sorting Code

The code for this starts with your result in HW2 of sorting. The solution of the basic software will be on GitHUB. See HW2 for descriptions. This problem does not require turning in additional code but it should be extended to get better statistical estimates and error bars to get chi square of fits. This intentional open ended as warm up for the analysis that many will want to use for the Team project. There should be short text file that describes the fitting methods and the ensemble averages in you fits. Indeed you may work with your team on this project but list all the collaborator in this write up. In class we will compare methods and ideas.

LEAST FITTING GUIDES: There are of course many tutorials on curve fitting and error estimate for the fitting parameters. Gunplot gives the output automatically but there is no documentation. One a bit mathematical but pretty concise is in Mathematica – of course. LeastSquareFitting Please google for more tutorials and share with class on Slack

2 Part 2: Better Statistics and Fit with Error bars

Extend the data files for HW2 to un statistics to estimate error bars and to a least square fit to estimate the goodness of fit If you have time you could add error bars to the table above and fit with error bars. The average of random cases defined as mean square deviation (called σ or standard error) a second Add error bars to the average (called σ) defined as mean square deviation

a second column next to the tabulate averages `xxxx`. These are define for each algorithm and size N by

$$\sigma^2 = \frac{1}{N_{trials}} \sum_{i=1}^{N_{trials}} (Swaps[i] - Mean)^2 = \frac{1}{N_{trials}} \sum_{i=1}^{N_{trials}} Swaps[i]^2 - Mean^2$$

where above we suggested fixing $N_{trials} = 10 - 100$ range. The average numbers of swaps in the 100 trials for each algorithm and size N in the table are:

$$Mean = \frac{1}{N_{trials}} \sum_{i=1}^{N_{trials}} Swaps[i]$$

You will want to have your code compute the standard error and put into another column in your output file. By the way all these analysis skill will likely come in handy for the team project.

Not you want a enlarge output table something like:

```
# N  insertion Mean Error | merge   Mean  Error  | ...
16   xxxx    xxxx       xxxx      xxxx   ....
32   xxxx    xxxx       xxxx      xxxx   ....
64   xxxx    xxxx       xxxx      xxxx   ....
128  xxxx    xxxx       xxxx      xxxx   ....
....
```

I used a print statements C style, `fprintf`, which is easier to format. The syntax:

```
fprintf(cFile,"#      N          Insertion     Mean      Error   |
                       Merge        Mean       Error     |
                       Quick        Mean       Error     |
                       Shell         Mean      Error     | \n ");

// Compute loops and averaging.

fprintf(cFile," %10d  %10d  %10.5e  %10.5e   %10d  %10.5e  %10.5e  %10d  %10.5e
         %10.5e  %10d  %10.5e  %10.5e \n", N, TotInsert,MeanInsert, rmsInsert,
         TotMerge, MeanMerge,  rmsMerge,  TotQuick,  MeanQuick, rmsQuick,
         TotShell ,  MeanShell,  rmsShell );
```

The syntax for opening and closing a file are:

```
FILE* cFile;
cFile = fopen ("Plotfile.txt","w+");
```

```
// Print lots of stuff accumulated from the loops to file in tabular form

fclose(cFile);
```

For general background information the `sorting.h` files has a few more sorting algorithms to play with. We could add others like bucket and improve pivots for quicksort etc.