Act 9/10

Optimization 101: Squaring a Number

1./

2.

3. $pow(x_12)$'s: 4.45189 s x*x's: 0.395591s \longrightarrow most efficient

4. Squareit is much more efficient than pow (x,2) and slightly ks efficient compared to x^*x .

Overhead: v.2s

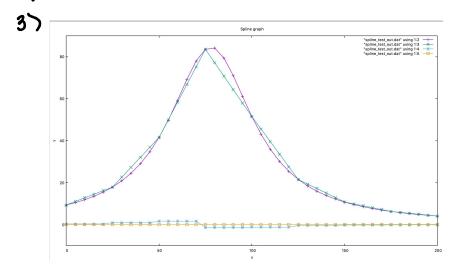
This would by worthwhile if you had to square a substantially large function and not just 2 numbers.

- 5. the macro was almost as efficient as x*x the 1st run, and more efficient on the 2nd run. I'd say they are "Heat" for most efficient.
- b. Using the macro would be most efficient. It is just as efficient as x*x but it would be better at squaring large functions and not just numbers. It is also easier to read for a user.
- 7. All of the times are now in the 10-5 or 10-6 range. This to clearly the best way to run code efficiently.

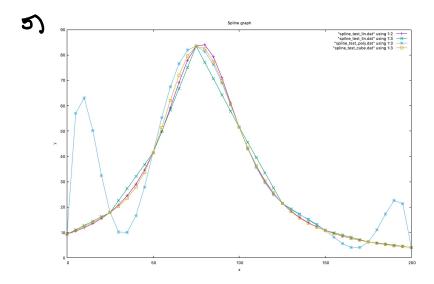
GSL Interpolation Routines

1) 🗸

2) 🗸



4) I got "Illegal Spline Type!" -> Pixed that lake.



Cubic Interpolation is closest to the "exact" value for y throughout the whole x-ronge. Linear does slightly worke than the cubic interp. Poly aloes an ot job mar the place, but a par job hear the cha points.

Python Scripts for CH Programs

- 1. 🗸
- 2.1 have tried
- 3. no
- 4. 🗸

Cubic Splining

- 1.
- 2. inline /
- 3. Used the same remax from an earlier session w/ the same spacing.

Command Line Myskry Jenemy Bowers