HW Goals:

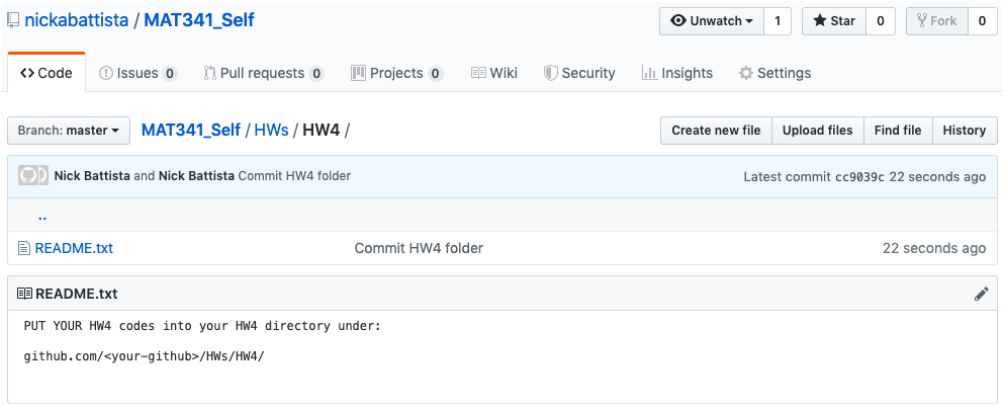
The purpose of this homework is to continue familiarizes ourselves with two different algorithms for 1D optimization - the ***Golden Search Algorithm*** *and* ***Successive Parabolic Interpolation****.*

HW DUE:

Tuesday, October 29 by 11:59pm

WHERE:

Put all of the codes for this homework into your personal GitHub: `MAT341_Self`→`HWs`→`HW4`. That is, you will have to `add, commit` and `push` these scripts to your GitHub for me to pull, see below for example:



1. Write a script called `golden_Search.m` that takes in one input, *tol* and returns the *number of iterations*, N, necessary to achieve a certain the error tolerance, *tol*, e.g.,

$$\text{function } \mathbf{N} = \text{golden\_Search(tol)}$$

Within this script, implement the *golden search algorithm* with the search step, $\tau = (\sqrt{5}-1)/2$, as discussed in class.

Use your algorithm to find the minimum of $f(x) = 0.5 - xe^{-x^2}$ within the interval $[0, 2]$. Have it return (output) the number of iterations, N, it takes to achieve the specific error tolerance, *tol*, that is inputted.

Answer the following and write your answers as comments at the bottom of the script:

(a) How many iterations does it take to achieve $1e-8$ accuracy?

2. Write a script called `successive_Parabolic_Interpolation.m` that takes in one input, *tol* and returns the *number of iterations*, N, necessary to achieve a certain the error tolerance, *tol*, e.g.,

function **N** = successive_Parabolic_Interpolation(tol)

Within this script, implement the *successive parabolic interpolation algorithm* with initial points $x_1 = 0$, $x_2 = 0.6$, and $x_3 = 0.9$.

Use your algorithm to find the minimum of $f(x) = 0.5 - xe^{-x^2}$ within the interval $[0, 2]$. Have it return (output) the number of iterations, N, it takes to achieve the specific error tolerance, *tol*, that is inputted.

Answer the following and write your answers as comments at the bottom of the script:

(a) How many iterations does it take to achieve $1e{-}8$ accuracy with the initial guesses as described above?

(b) Change the initial x_3 value to $x_3 = 2$. How does the number of iterations change? Does MATLAB issue any warnings? What does the warning say? What do you think that means intuitively about the linear system for finding the parabola coefficients?

Change the initial x_3 to $x_3 = 1.2$ for Problem 3.

3. Write another script called `vary_Error_Tolerances_To_Compare.m` that takes no inputs (nor returns anything), e.g.,

function vary_Error_Tolerances_To_Compare()

that calls the previous two algorithms you've made `golden_Search` and `successive_Parabolic_Interpolation` and calls them to run for a variety of error tolerances. Recall that each of those scripts outputs the number of iterations, N, it takes to achieve a particular error tolerance. Run both of those algorithms for each error tolerance in the following vector:

$$errTolVec = [1e{-}1 \;\; 1e{-}2 \;\; 1e{-}3 \;\; 1e{-}4 \;\; 1e{-}5 \;\; 1e{-}6 \;\; 1e{-}7 \;\; 1e{-}8 \;\; 1e{-}9 \;\; 1e{-}10 \;\; 1e{-}11 \;\; 1e{-}12],$$

e.g., you will loop over every component of the vector $errTolVec$. Save each algorithm's number of iterations into a storage vector. Use a different vector for each algorithm.

Make **two** plots that illustrate each algorithms number of iterations, N vs. specific error tolerances, *tol*. For one plot, use logarithmic axis in the horizontal direction only (e.g., `semilogx`) while in the other use logarithmic axis for both (e.g., `loglog`). On each figure, plot both sets of data. That is, you want to see a comparison of each algorithm's number of iterations vs. error tolerances.

Make sure to use the *# of Iterations, N* you calculate as the dependent variable (vertical axis) and the *error tolerance, tol* as the independent variable (horizontal axis).

Make sure to:

(a) Label the axes

(b) Make the Golden Search's line color blue and the Successive Parabolic Interpolation's line color red

(c) Change the thickness of the line (e.g., line width) to 5.

(d) Make a figure legend, e.g., `legend('Golden Search','Succ. Para. Interp.')`. (if you listed the Golden Search first in the figure)

(e) Recall to plot multiple sets of data on the same plot, use the `hold on` command after the plotting statement.

(f) You can explicitly make multiple figures, by using the command `figure(1)` and `figure(2)` before anything related to its plot.

Answer the following and write your answers as comments at the bottom of the script:

(a) Which algorithm appears converges faster to the minimum for less accurate tolerances?

(b) What happens when you increase the accuracy threshold? Does that algorithm always converge quicker? Why do you think one of the algorithms changes its convergence speed?