Report

In this homework, a Python routine called `pyRun_rootFinder.py` was implemented. This routine compiles, runs and plots the results of the Newtons root finder algorithm we studied in the class.

The test function is the following (ftn_type = 2):

\[ f(x) = (x - 1) \log(x) \]

The only root of this function is at \( x = 1 \). We tried to find this root using two different initial guesses, one that is relatively close to the solution \( x_0 = 0.1 \) and other that is farther away \( x_0 = 10 \). A plot of the test function and the initial guesses can be observed in figure 1.

To run the code, you just need to type in the command line:

\[
\texttt{>> python pyRun_rootFinder.py}
\]

The routine `pyRun_rootFinder.py` does not need any input by itself. However, the following variables could be modified in the code of the `pyRun_rootFinder.py` routine to change the parameters included in the runtime parameter file called `rootFinder.init`:

- `newton_path` : [char] Path where the newton_rootFinder directory is located.
- `run_name_input` : [char] Output file name, e.g., `rootFinder_[run_name_input].dat`.
- `method_type_input` : [char] Search method (`'newton'` or `'modified newton'`).

![Figure 1: Test function in blue, initial guesses in red and root in green](image-url)
• x_beg_input : [real] Lower threshold for the search domain.
• x_end_input : [real] Upper threshold for the search domain.
• max_iter_input : [int] Maximum number of iterations.
• ftn_type_input : [int] Type of function (1 or 2).
• init_guess_input : [real] Initial guess for root search.
• multiplicity_input : [int] The multiplicity of the root when using the 'modified newton' method.
• threshold_input : [real] Threshold value for solution convergence.

The output of the pyRun_rootFinder.py routine is the plot of the results for each threshold value to both screen and to a file, and the .dat files containing the plotting data. This files will be stored in the directory indicated in the variable newton_path. The name of the plot files follows the next convention:

• result_[ftn_type_input]_[threshold_input]_[init_guess_input]_solution.png or
• result_[ftn_type_input]_[threshold_input]_[init_guess_input]_error.png.

Results with initial guess $x_0 = 0.1$

In figures 2 to 7 we can observe the curves for the error and solution for three different values of threshold ($10^{-4}$, $10^{-6}$ and $10^{-8}$) using $x_0 = 0.1$ as initial guess.

For this initial guess, the solution appears to converge as a logarithmic function, getting close to 1 (the real value of the root) after 20 iterations. Furthermore, to reach each desired threshold value, the algorithm needed 26, 39, 52 iterations respectively, which looks like an arithmetic sequence.

Also, for this initial guess, the error appears to converge to 0 after 20 iterations. However, unlike the solution curve, the error curve has a peak at the third iteration, falling drastically from then to the value of zero.

Results with initial guess $x_0 = 10$

In figures 8 to 13 we can observe the curves for the error and solution for three different values of threshold ($10^{-4}$, $10^{-6}$ and $10^{-8}$) using $x_0 = 10$ as initial guess.

For this initial guess, the solution appears to converge as an exponential function, getting close to 1 (the real value of the root) after 10 iterations. Furthermore, to reach each desired threshold value, the algorithm needed 26, 39, 52 iterations respectively, which (again) looks like an arithmetic sequence and is the exact same number of iterations needed using the other value for the initial guess ($x_0 = 0.1$).

In this case (initial guess $x_0 = 10$), the error curve behaves very similar to the solution curve.
Surprisingly, the convergence of the solution and the error using an initial guess of $x_0 = 10$ is as fast as using an initial guess of $x_0 = 0.1$ (the same number of iterations are needed in both cases). Furthermore, using an initial guess of $x_0 = 10$, the error curve doesn’t show any peak, unlike the error curve using an initial guess of $x_0 = 0.1$. Also, the solution using an initial guess of $x_0 = 10$ appears to reach exponentially the real value of the root, while the solution using an initial guess of $x_0 = 0.1$ appears to reach logarithmically the real value of the root.

To explain this, we can look the derivative of the test function: $f'(x) = \log(x) + 1 - \frac{1}{x}$. This function has a large slope for values of $x$ between zero and one, and a very small slope for values of $x$ larger than one. Due this, the curves for the error and solution are more smooth using an initial guess of $x_0 = 10$ than using an initial guess of $x_0 = 0.1$. 

**Conclusion**

Fig. 2: Error convergence with initial guess $x_0 = 0.1$ and threshold value $t = 10^{-4}$
Fig. 3: Solution convergence with initial guess $x_0 = 0.1$ and threshold value $t = 10^{-4}$

Fig. 4: Error convergence with initial guess $x_0 = 0.1$ and threshold value $t = 10^{-6}$
Fig. 5: Solution convergence with initial guess $x_0 = 0.1$ and threshold value $t = 10^{-6}$

Fig. 6: Error convergence with initial guess $x_0 = 0.1$ and threshold value $t = 10^{-8}$
Fig. 7: Solution convergence with initial guess $x_0 = 0.1$ and threshold value $t = 10^{-8}$

Fig. 8: Error convergence with initial guess $x_0 = 10$ and threshold value $t = 10^{-4}$
Fig. 9: Solution convergence with initial guess $x_0 = 10$ and threshold value $t = 10^{-4}$

Fig. 10: Error convergence with initial guess $x_0 = 10$ and threshold value $t = 10^{-6}$
Fig. 11: Solution convergence with initial guess $x_0 = 10$ and threshold value $t = 10^{-6}$

Fig. 12: Error convergence with initial guess $x_0 = 10$ and threshold value $t = 10^{-8}$
Fig. 13: Solution convergence with initial guess $x_0 = 10$ and threshold value $t = 10^{-8}$