FIT3139: Lab questions for week 4

Question 1

Let

$$A = \begin{bmatrix} 1 & 4 \\ 1 & 1 \end{bmatrix}$$

Work out the following on a paper. (A calculator is allowed if you need one.):

- What is the characteristic polynomial of A?
- Find its eigenvalues?
- What are the corresponding eigenvectors?

Question 2

A system is defined using the following relationship:

$$-0.5S_k = S_{k-1} - S_{k+1}$$

where initial values are $S_0 = 0$ and $S_1 = 1$.

- At what rate is the system growing?
- Compute and plot the values of $S_0, S_1, \cdots S_{100}$

Question 3

Implement the interval bisection method [1] to find at least one root of the following equations:

- $x^3 - 2x - 5 = 0$,
- $e^{-x} = x$,
- $x \sin(x) = 1$,
- $x^3 - 3x^2 + 3x - 1 = 0$

You should terminate the iterations when the bracket size falls below the tolerance of `tol` $= 10^{-5}$. If needed you can pass functions as arguments of other functions. This is standard and works out of the box in Python. It can be done in Matlab too using `@`[2].

Question 4

Implement Newton and secant methods on the same set of equations as above to find at least one root. Terminate the iteration when the absolute difference between successive values, $|x_{k+1} - x_k|$, falls below a tolerance of 10^{-9}.

[1] the lower and upper values of a bracket to the root which is binary searched in that interval can be passed to your program as arguments. The onus is on you to choose the bracket sensibly

[2] `https://au.mathworks.com/help/matlab/matlab_prog/pass-a-function-to-another-function.html`

Your `find_root_secant` should receive a function argument, and your `find_root_newton` should receive a function argument and another function which is the derivative.

Compare the convergence from the two algorithms. Is it aligned with what we learned in theory?

Question 5

Compute and list the first several iterations of Newton's method for solving each of the following equations, starting with the given initial guess.

(a) $x^2 - 1 = 0, x_0 = 10^6$

(b) $(x-1)^4 = 0, x_0 = 10$

Can you estimate the rate of convergence from the output?

Question 6

Build a program to compute at least one root for an arbitrary polynomial using the Newton method. Represent the target polynomial as a list of coefficients in descending order. For example, the function $f(x) = 2x^3 + x + 1$ should be represented as the list `[2, 0, 1, 1]`. Use as your initial guess x_0, the root of the linear approximation of the original function. For the example function $f(x)$, this would be $x_0 = -1$.

Optional for fun

Modify your Newton method algorithm to work symbolic expressions. There are symbolic packages to do this in both, Python[3] and Matlab[4].

Optional for fun

In neutron transport theory, the critical length of a fuel rod is determined by the roots of the equation $\cot(x) = (x^2 - 1)/(2x)$. Can you find the smallest positive root of this equation using one of your implementations of Bisection, Newton or Secant method?

[3] http://docs.sympy.org/latest/tutorial/calculus.html

[4] `http://au.mathworks.com/help/symbolic/diff.html;jsessionid=49c6fd7f665bba501e52e107d8bb`