APM348 **Due: 11:59pm January 26**

HOMEWORK 1

In this homework you will implement one-dimensional Newton's method in Python and use it to solve the pig selling problem from lecture. You are responsible to submit a PDF homework report on the due date answering the 1 question in section 1, the 3 questions in section 2, and the 5 questions in section 3. You are encouraged to attempt the questions in section 4, particularly if you have prior Python experience, but they will not be graded.

Clone the file `homework1-exercises.ipynb` into your Jupyter Notebook and solve the exercises directly there. Export the Jupyter Notebook as a PDF and submit it to gradescope.

1 Function Handles in Python

Sometimes you will want to pass a (mathematical) function as an argument to a computer subroutine (function) as we will do shortly when implementing Newton's method. This is needed since the Python function we write for applying Newton's method needs to know the function for which it is trying to find roots. We could implement that function separately. However, for short functions it will be easier to use anonymous functions.

1. In a Python script, use an anonymous function for $g(x) = \cos(x^2)$ as `g = lambda x :  cos(x^2)`. Write a loop that iteratively applies g, starting from a value of 1. Your script should output $g(1)$, $g(g(1))$, $g(g(g(1)))$, ... You may wish to plot the sequence of values. What value is the sequence approaching? Which function is this value a root of? This is a fixed point iteration, which is expected to converge slowly. Newton's method can find this root much quicker, so let us implement it.

2 Newton's Method

Newton's method produces a sequence of approximations to a solution of the equation $f(x) = 0$. You must provide an initial guess and a procedure for evaluating $f(x)$ and $f'(x)$. We use the root of the linear approximation to f at x_n to obtain a (hopefully) better approximation to the root, x_{n+1}. This gives the iteration

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

2. Implement Newton's method. Use the stopping criteria that the function value be less (in absolute value) than a tolerance (`tol`) and a minimum of `miniter=3` iterations be made. Use the command `print` to output formatted results from each iteration.

3. Try your function out on finding the roots of $f(x) = \exp(x) - 4x$. Note the function handle for $f'(x)$ is obtained by differentiation of $f(x)$ by hand. The upper plot shows $f(x)$ and its two roots. Initial guesses in the range [0,3] converge to one of the two roots as shown in the lower plot. What property of $f(x)$ would you guess defines the boundary between each 'basin of attraction'?

4. Find the roots of $\cos(x^2) - x$. How many iterations does it take to obtain 12 correct digits? Use Python's `root_scalar` function to compute the same value.

3 Pig Selling Problem

Recall the optimization model of maximizing the profit $P(t) = p(t)w(t) - C(t)$ from the sale of a pig with weight $w(t)$ in kg, price $p(t)$ in dollars per kg, and cost of keeping the pig $C(t)$ up to time t in dollars.

5. Write a Python script that uses anonymous functions and the `root_scalar` command (or your Newton's method function) to solve the pig selling problem. Specify $p(t)$, $w(t)$, and $C(t)$ along with $p'(t)$, $w'(t)$, and $C'(t)$ at the beginning of the file as function handles and then find t so that $p'(t)w(t) + p(t)w'(t) - C'(t) = 0$.

   Hint: you will need an anonymous function that combines your other functions like
   `dPdt = lambda t : pp(t)*w(t) + p(t)*wp(t) - Cp(t)`

6. When should the farmer sell the pig if $p(t) = 0.65 - 0.01t$, $w(t) = 90 + 3t$, and $C(t) = 0.45t$?

7. When should the farmer sell the pig if $p(t) = 0.65 - 0.01t$, $w(t) = 90\exp(\gamma t)$, and $C(t) = 0.45t$? You will need to pick a value of γ. Use a nominal value of $\gamma = \frac{1}{30}$. Make a plot of the optimal time to sell versus γ for γ in the range 0.030 and 0.035. The graph monotonically increases. Explain the meaning of this result.

8. Compute the sensitivity of the optimal time to sell with respect to γ (at the nominal value of γ). You will need to use a numerical approximation to the derivative.

9. Make **different** but reasonable choices for $p(t)$, $w(t)$, and $C(t)$. Is the solution method (root finding) successful for your choices? Briefly discuss the model's robustness.

4 Extra: Automatic Differentiation

It is annoying to have to differentiate functions by hand (or symbolically) for use with Newton's method. One alternative is to numerically differentiate the function using previous root estimates and their function values. This is called the secant method, but the approximation slows the convergence of the method.

Automatic differentiation is different from symbolic and numeric differentiation. It is able to produce a value for $f'(a)$ while computing $f(a)$ by applying the rules of differentiation to the elementary steps involved with the expression of f. Richard Neidinger has provide a MATLAB class for automatic differentiation. I have "translated" this class to Python, and we will use it.

10. Load the `ValDer` library (stands for value and derivative). This file allows you to generate an object that will return both the function value and the derivative. Try it out with `a = ValDer(2,1); y = 1+a*a;` which means that `a` is our independent variable, and we will evaluate it when `a=2`, and `y` is the dependent variable. You should expect `y.val = f(2) = 1+2^2 = 5` and `y.der = f'(2) = 2*2 = 4`.

11. Implement Newton's method with automatic differentiation.

12. Find the roots of $\exp(-\sqrt{x})\sin(x\log(1+x^2))$ using your function.