Matlab exercises

Tutors: Ellya Kawecki & Hamza Alawiye

October 23, 2018

For each example, you should write a code script that we generate the desired outcome, with the given inputs. I.e., if the question has a vector and and a number listed as the inputs, your script should be prepared to accept those kinds of inputs. Feel free to jump between the questions.

Plotting:

1. Using the plot command, plot the graph of $y(x) = e^{\sin(x^2)}$ on the interval 0 and 2π. Name the script MyPlot1.

   Inputs: 0, 2π, and a functionhandle (i.e., f = @(x) ...).

2. Adapt the script from question 1 as follows: Label the axes, and add a title that includes your name and the College name.

   Inputs: should be the same as in question 1.

3. Adapt this script further: it should now accept another value $x \in [0, 2\pi]$ and output that value in a sensible way (e.g. use the "disp" command to tell us something about the value. Optional: can you use an "if" statement to demand that the extra value x actually does belong to $[0, 2\pi]$?

   Inputs: The same as questions 1 and 2, with the extra input x.

4. Write a new script, based off of question 1 and 2 (i.e., it doesn't have the extra input given by question 3), that takes an arbitrary interval $[a, b]$, and plots the function $f(x) = e^{\sin(x^2)}$ on this interval. The interval should consist of N equally spaced values. Name the script MyPlot2

   Optional: can you use an "if" statement that allows your code to run the same way, regardless of which order the values a and b are entered? I.e., MyPlot2(a,b,N) and MyPlot2(b,a,N) should do the same thing.

   Inputs: a, b, f, N.

5. Directly from the command window, use the "hold on" command, in conjunction with your MyPlot2 script, to plot the three following functions:

   $$f(x) = e^{\sin x^2}, \quad g(x) = e^{\cos(x^2)}, \quad h(x) = \ln(x),$$

   on the interval $[\pi, 3\pi]$. The interval should consist of 50 equally spaced points.

Iterative schemes: Consider the following numerical methods:

- Fixed Point method: $x_{n+1} = g(x_n), \quad n = 1, 2, \ldots$
- Newton's method: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 1, 2, \ldots$
- Bisection method: For an interval $[a, b]$, and a function $f : [a, b] \to \mathbb{R}$ satisfying

  $$\text{sign}(f(a)) = -\,\text{sign}(f(b)).$$

  Define $a_0 = a$, $b_0 = b$ and $c_k = (a_k + b_k), k = 1, 2, \ldots.$

  If $\text{sign}(f(c_k)) = \text{sign}(f(a_b))$, set $a_{k+1} = c_k$ and $b_{k+1} = b_k$. Otherwise set $a_{k+1} = c_k$ and $b_{k+1} = c_k$.

Newton's method and the Bisection method are both used to solve problems of the form: find $x \in [a, b]$ such that

$$f(x) = 0,$$

where $f : [a, b] \to \mathbb{R}$.

The fixed point method is used to solve problems of the form: find $x \in [a, b]$ such that

$$g(x) = x,$$

where $g : [a, b] \to [a, b]$ (i.e., x is a fixed point of g, since g maps x to x).

1. Write a script that implements the fixed point method, with a limit on the amount of iterations, and ensures (upon converging) that either the error $|x_k - x|$ or the residual $|g(x_k) - x|$ is smaller than a given tolerance ε.

   Inputs: N (maximum number of iterations), x_0 (initial guess), ε (tolerance), and a function handle (the function g).

2. Write a script that implements Newton's method, with a limit on the amount of iterations, and ensures (upon converging) that either the error $|x_k - x|$ or the residual $|f(x_k)|$ is smaller than a given tolerance ε.

   Inputs: N (maximum number of iterations), x_0 (initial guess), ε (tolerance), and two function handles (the function f, and it's derivative f').

3. Write a script that implements the Bisection method, with a limit on the amount of iterations, and ensures (upon converging) that either the error $|x_k - x|$ or the residual $|f(x_k)|$ is smaller than a given tolerance ε.

   Inputs: N (maximum number of iterations), x_0 (initial guess), ε (tolerance), a function handle (the function f), and the interval end points a and b.

4. Test the script for the Fixed Point method that you have written with the following examples:
   - $x = g_1(x) = \frac{1}{13}(14 - x^3), \quad x_0 = 0,$
   - $x = g_2(x) = e^{-x}, \quad x_0 = 1,$
   - $x = g_3(x) = x^3 + 3x^2 - 3, \quad x_0 = 0.5,$ or $x_0 = -0.5$.

5. Test the scripts for the Bisection method and Newton's method that you have written with the following examples (start with $x_0 = 1/2$):
   - $f_1(x) = \cos(x), \quad [a, b] = [-0.5, 3],$
   - $f_2(x) = x\cos(x), \quad [a, b] = [-1, 1],$
   - $f_3(x) = \sin(x)\sinh(x), \quad [a, b] = [-1, 1],$
   - $f_4(x) = x - \sin(x)$.

Fibonacci: The Golden Ratio is a number that occurs in nature and art. It is defined as

$$\varphi = \frac{1 + \sqrt{5}}{2}.$$

It is related to the sequence of Fibonacci numbers F_n (defined recursively by $F_1 = 0, F_2 = 1$ and $F_{n+1} = F_n + F_{n-1}, n \geq 2$) by taking the limit of ratios of successive Fibonacci numbers,

$$\varphi = \lim_{n \to \infty} \frac{F_n}{F_{n-1}}.$$

1. Write a script that generates *the* n^{th} Fibonacci number.

   Inputs: n.

2. Write a function that generates the *first* n Fibonacci numbers.

   Inputs: n.

3. Using either the script from question 1 or question 2, write a script that outputs the n^{th} approximation to the Golden Ratio (i.e., F_n/F_{n-1}).

   Inputs: n

4. Adapt the script from question 3, so that gives an approximation to the golden ratio, with accuracy 10^{-7} (Note 1e-7 = 10^{-7} in Matlab). Inputs: n, 10^{-7}.

5. Write a script that calculates the error from the approximation of the Golden ration, i.e., $e_n := |\varphi - F_n/F_{n-1}|$ (note that $\text{abs}(x) = |x|$ in Matlab.

   Let n run from $n = 1$ to $n = N$, where N corresponds to the F_n/F_{n-1} that produces error less than 10^{-7} in the code for question 4. Plot these errors against n.

   Inputs: n, 10^{-7}.