

Homework #1

Deadline: February 23 (Wednesday) 4:00am EST

Scorelator will begin accepting submissions from February 15 (Tuesday)

Problem 1

Geometrically, the act of computing a square root of a positive number A is the act of constructing a square with area equal to A . Note that if a square has the same area A as a rectangle of length L and width W , then the side-length $\sqrt{A}$ of the square must lie between L and W . Also, observe that we can make any rectangle “more square” (while maintaining the same area) by replacing its length L by the averaged value $(L + W)/2 =: L_{\text{new}}$ and replacing its width W by $A/L_{\text{new}} =: W_{\text{new}}$. This averaging process can be iterated to generate a sequence of rectangles that become more and more square. The value of $\sqrt{A}$ will always be sandwiched between the length and the width of each rectangle; so if we think of the sequence of rectangle lengths as converging to $\sqrt{A}$, then the absolute error of the approximation will be bounded by the difference of the length and width of each rectangle.

(a) Write a function `new_sqrt` that:

- takes as input a positive real number A followed by a tolerance level;
- starts with a rectangle of length A and width 1;
- generates sequences of lengths L and widths W of rectangles that are getting “more and more square”, using the averaging process described above;
- continues until the absolute error is no more than the given tolerance;
- returns the sequence of L -values as an approximation to $\sqrt{A}$.

(b) Write a script that calls `new_sqrt` to compute sequences that tend to

$$\sqrt{10}, \quad \sqrt{3141}, \quad \sqrt{55555}$$

with absolute error not exceeding 10^{-9} .

Save the sequences as column vectors (i.e. vertical lists) in three ASCII files: `A11.dat` for $\sqrt{10}$, `A12.dat` for $\sqrt{3141}$ and `A13.dat` for $\sqrt{55555}$.

Problem 2

The value of $\sqrt[3]{21}$ can be approximated by fixed-point iterations using the following maps:

$$(i) \phi_1(x) = \frac{1}{21} \left(20x + \frac{21}{x^2} \right)$$

$$(ii) \phi_2(x) = x - \frac{x^3 - 21}{3x^2}$$

$$(iii) \phi_3(x) = \sqrt{\frac{21}{x}}$$

Write Matlab code (scripts or a script that calls a function) to compute the sequence $\{x_n\}$ generated by each of these iterated maps, starting with $x_1 = 1$. For the stopping criterion, use $\left| 1 - \frac{x_n}{x_{n-1}} \right| \leq 10^{-7}$.

Save the sequences as column vectors (i.e. vertical lists) in three ASCII files: `A21.dat` for ϕ_1 , `A22.dat` for ϕ_2 and `A23.dat` for ϕ_3 .

(PLEASE TURN OVER)

Problem 3

The **secant method** for finding a root of $f(x) = 0$ is a modification of Newton's method where the derivative $f'(x_k)$ is replaced by the difference quotient $\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$. Thus, the recursive formula for the secant method is

$$x_{k+1} = x_k - \frac{(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})} f(x_k)$$

Note that, like for the Fibonacci sequence, to compute the next term in the sequence (x_{k+1}) , we need the term just computed (x_k) and the preceding term (x_{k-1}) . Therefore, *two* initial guesses are needed to get the process started.

Write Matlab code that computes the sequence $\{x_k\}$ generated by the secant method to approximate:

- (i) $\sqrt{10}$, starting with $x_1 = 3$ and $x_2 = 4$
- (ii) the solution of $x^3 + 4x^2 - 10 = 0$, starting with $x_1 = 1$ and $x_2 = 2$
- (iii) the solution of $e^x = \tan x$, starting with $x_1 = -4$ and $x_2 = -2$

For the stopping criterion, use $|f(x_k)| \leq 10^{-14}$.

Save your sequences as column vectors (i.e. vertical lists) in three ASCII files: **A31.dat** for part (i), **A32.dat** for part (ii) and **A33.dat** for part (iii).

Note: If you use **fprintf** to save your output, make sure you format it with at least 8 digits after the decimal point. You do not need to worry about specifying the format if you use the **save** command.