




Math 3607: Homework 10

Wednesday, April 13, 2022

TOTAL: 30 points

- Problems marked with  are to be done by hand; those marked with  are to be solved using a computer.
- **Important note.** Do not use *Symbolic Math Toolbox*. Any work done using `sym` or `syms` will receive NO credit.
- **Another important note.** When asked write a MATLAB function, write one at the end of your live script.

1. (Secant method; exercise from lecture)  Assume that iterates x_1, x_2, \dots generated by the secant method converge to a root r and $f'(r) \neq 0$. Let $\epsilon_k = x_k - r$. Show that

- (a) The error ϵ_k satisfies the approximate equation

$$|\epsilon_{k+1}| \approx \left| \frac{f''(r)}{2f'(r)} \right| |\epsilon_k| |\epsilon_{k-1}|.$$




- (b) If in addition $\lim_{k \rightarrow \infty} |\epsilon_{k+1}| / |\epsilon_k|^\alpha$ exists and is nonzero for some $\alpha > 0$, then


$$|\epsilon_{k+1}| \approx \left| \frac{f''(r)}{2f'(r)} \right|^{\alpha-1} |\epsilon_k|^\alpha, \quad \text{where } \alpha = \frac{1 + \sqrt{5}}{2}.$$

2. (Multidimensional Newton's method; **FNC** 4.5.5) Suppose one wants to find the points on the ellipsoid $x^2/25 + y^2/16 + z^2/9 = 1$ that are closest to and farthest from the point $(5, 4, 3)$. The method of Lagrange multipliers implies that any such point satisfies


$$\begin{cases} x - 5 = \frac{\lambda x}{25}, \\ y - 4 = \frac{\lambda y}{16}, \\ z - 3 = \frac{\lambda z}{9}, \\ 1 = \frac{1}{25}x^2 + \frac{1}{16}y^2 + \frac{1}{9}z^2 \end{cases}$$

for an unknown value of λ .

- (a)  Write out this system in the form $\mathbf{f}(\mathbf{u}) = \mathbf{0}$.
- (b)  Write out the Jacobian matrix of this system.
- (c)  Use `newtonsys` from class with different initial guesses to find the two roots of this system. Which is the closest point to $(5, 4, 3)$ and which is the farthest?

3. (Polynomial vs. piecewise polynomial interpolation; **FNC 5.1.2**)  The following table gives the life expectancy in the U.S. by year of birth:

year	1980	1985	1990	1995	2000	2005	2010
expectancy	73.7	74.7	75.4	75.8	77.0	77.8	78.7

- (a) Defining “year since 1980” as the independent variable, use `polyfit` to construct and plot the polynomial interpolant of the data.
- (b) Use `interp1` to construct and plot a piecewise cubic interpolant (use ‘`spline`’ option) of the data.
- (c) Use both methods to estimate the life expectancy for a person born in 2007. Which value is more believable?
4. (Piecewise cubic interpolation; **FNC 5.1.3**)  The following two point sets define the top and bottom of a flying saucer shape:


Top:

x	0	0.51	0.96	1.06	1.29	1.55	1.73	2.13	2.61
y	0	0.16	0.16	0.43	0.62	0.48	0.19	0.18	0

Bottom:

x	0	0.58	1.04	1.25	1.56	1.76	2.19	2.61
y	0	-0.16	-0.15	-0.30	-0.29	-0.12	-0.12	0

Use piecewise cubic interpolation to make a picture of the flying saucer.

5. (Optional: Fast inverse square root)  The almost universally used algorithm to compute \sqrt{a} , where $a > 0$, is the recursion

$$x_{k+1} = \frac{1}{2} \left(x_k + \frac{a}{x_k} \right),$$

easily obtained by means of Newton’s method for the function $f(x) = x^2 - a$; a derivation was shown in class recently. One potential problem with this method is that it requires a floating-point division, which not all computer processors support, or which may be too expensive for a particular application. For these reasons, it is advantageous to devise a method for computing the square root that does not require any floating-point division, (except division by 2, which can be easily done by shifting the binary representation one bit to the right), but only addition, subtraction, and multiplication. The trick for doing this is to use Newton’s method to compute $\frac{1}{\sqrt{a}}$, and then obtain \sqrt{a} by multiplying by a .


Your goal is to write a MATLAB function for computing the square root using the method describe above. Division by 2 is allowed, but no other floating-point division is. The function must work for any input value $a > 0$.

- Set the tolerance for the *relative error* in your solution to 10^{-12} where

$$\text{relative error} = \frac{|x_{k+1} - x_k|}{x_k}.$$

- Use your function to compute $\sqrt{35}$, $\sqrt{2.3 \times 10^{-6}}$, and $\sqrt{17^{15}}$.

Notes.

- This sort of software assisted acceleration is used in gaming. Google *Origin of Quake3's Fast InvSqrt()*.
 - I recently stumbled upon this YouTube video with an excellent explanation on this algorithm. Check it out [here](#).
6. (Optional: Basin of attraction)  Do **LM** 13.1–33 and 34. This is a long problem and it is not very likely that it will appear on your exam. However, if you want extra challenge or are interested in generating cool figures using MATLAB skills acquired and your understanding of rootfinding, give it a go. The final outcome of the lengthy process is the colorful representation of so-called the *basin of attraction* as shown below.

