University of Oslo Fall, 2023
Department of Mathematics Mandatory assingment: 1

Author: Jonas Semprini Næss

MAT4110: Introduction to Numerical Analysis

Problem 1.

a.) *Let (x_k) be a sequence obtained by Newton's method for solving the above equation. Explain why $\lim_{k\to\infty} x_k = \xi$ if x_0 is sufficiently near ξ and show that the sequence converges with at least order $q = 3$.*

Solution:

To see that (x_k) indeed will converge for a sufficient x_0 we need to analyse how $\arctan(x)$ behaves in regards to contractivity. Indeed, $f'(x) = \frac{1}{1+x^2} \leq 1, \quad \forall x \in \mathbb{R}$ and $|\arctan(x)| \leq |x|$ which by the Mean Value Theorem tells us that

$$|f(x) - f(y)| \leq |f'(z)||x - y|, \qquad \text{for } z \in (x, y).$$

Where WLOG $y > x$. Now notice that if $x \to \infty$ then $f'(x) \to 0$, implying that $\arctan(x)$ would violate the contraction criteria given large enough x. It is also easy seeing that since $f'(x)$ is rapidly decreasing on $\mathbb{R}$, namely $0 < f'(x) < |\frac{1}{x}|$, then as $x \to 0 \Rightarrow f'(x) \to 1$. Thus if $x \in (0, r)$ where $r << \infty$ then $|f'(x) < 1|$. Now let $z \in (\xi, r)$, and denote $f'(z) = L$, meaning $0 < L < 1$, then

$$\begin{aligned} |f(\xi) - f(r)| &\leq |f'(z)||\xi - r| \\ &= L|\xi - r|. \end{aligned}$$

which shows that $\arctan(x)$ is a contraction for a sufficient r which we can set as an upper boundary for x_0.

Order of convergence

When showing the order of convergence we start off by denoting $h(x) = x_n - \frac{f(x_n}{f'(x_n)}$. This gives us the functional iteration

$$x_{n+1} = h(x_n)$$

which if x_0 is sufficiently close to ξ will converge to ξ (i.e part (1) of this exercise). To express the order of convergence we firstly look at the difference

$$x_{n+1} - \xi = h(x_n) - h(\xi)$$

which if we Taylor expand $h(x)$ about the point x_n we get that

$$x_{n+1} - \xi = h(x_n) - h(\xi) = \frac{1}{q!} h^{(q)}(\eta_n)(x_n - \xi)^q, \quad q \in \mathbb{N}$$

where by the Squeeze Theorem will ensure that when $x_n \to \eta \Rightarrow \eta_n \to \xi$. Moreover, by continuity we then have that

$$\frac{|x_{n+1} - \xi|}{|x_n - \xi|^q} = \frac{1}{q!}|h^{(q)}(\xi)|.$$

Now since $r = h(r)$ only if $f(r) = 0$ we always have that $h'(r) = 0$ since

$$h'(x) = \frac{f(x)f''(x)}{f'(x)^2}.$$

Thus, for the order of convergence to be at least k the expression

$$\frac{1}{q!}|h^{(q)}(\xi)| \neq 0$$

for at least q. Now if we use (1.24) from (S.M) we have that

$$\frac{|x_{n+1} - \xi|}{|x_n - \xi|^q} = \frac{1}{2}\left|\frac{f''(\xi)}{f'(\xi)}\right| = 0$$

$$\Downarrow$$

$$h''(r) = 0$$

which means the order of convergence is at least cubic.[1] ■

b.) *Show that*

$$g(x) := \begin{cases} f(x)/xf'(x) & \text{if } x \neq 0 \\ 1 & \text{if } x = 0 \end{cases}$$

is a continuous and symmetric function (meaning that $g(x) = g(-x)$), and that Newton's method (for this particular problem) can be written as

$$x_{k+1} = x_k(1 - g(x_k))$$

Solution:

Continuity

Since we are dealing with a piecewise function we check that the limits as $x \to 0$ coincide.

$$\lim_{x\to 0} \frac{f(x)}{xf'(x)} = \lim_{x\to 0} \frac{\arctan(x)(1+x^2)}{x} \overbrace{=}^{L'H} \frac{1 - \arctan(x)(\frac{2x}{(1+x^2)^2}}{1} = 1 \quad (1)$$

$$\lim_{x\to 0} 1 = 1 \quad (2)$$

Meaning $g(x)$ is continous.

[1] In this specific problem it is actually exactly of order 3

Symmetry

By definition we have that a symmetric function can be expressed by the equality $g(x) = g(-x)$. Thus to check that this holds we simply calculate $g(x) - g(-x)$. This gives

$$\begin{aligned} g(x) - g(-x) &= \frac{f(x)}{xf'(x)} - \frac{f(-x)}{-xf'(-x)} \\ &= \frac{\arctan(x)(1+x^2)}{x} + \frac{\arctan(-x)(1+(-x)^2)}{x} \\ &= \frac{\arctan(x)(1+x^2) - \arctan(x)(1+x^2)}{x} \\ &= 0 \end{aligned}$$

where we have used that $\arctan(-x) = -\arctan(x)$ since it is an odd function.

Newtons Method

To show that Newtons method can be written thaty way we simply write it out as

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

and we are done. ■

c.) *Show that there exists a unique point $x^* \in (0, \infty)$ such that $g(x^*) = 2$ and that*

$$g(x) > 2, \quad \forall x \in (x^*, \infty).$$

To show the uniqueness of x^* we will proceed by doing a proof by contradiction.
Suppose there exists two points $x_1, x_2 \in (0, \infty)$ such that $g(x_1) = g(x_2) = 2$. As defined we know that

$$g(x) = f(x)/xf'(x) = \frac{\arctan(x)(1+x^2)}{x} > x \quad , x \in (0, \infty)$$

meaning $g(x)$ is strictly increasing on $(0, \infty)$. Since $g(x)$ is strictly increasing that means

$$g(x_1) < g(x_2) < \cdots < g(x_n), \quad \text{for } x_1 < x_2 < \cdots < x_n \in (0, \infty)$$

but we defined $g(x_1) = g(x_2)$ for two arbitrary values, meaning we arrive at a contradiction and there must then exist a unique point x^* such that $g(x^*) = 2$. Thus, must all points $x_1, x_2, \ldots, x_n \in (x^*, \infty)$ yield $g(x_k) > 2$, for $k \in \mathbb{N}$. ■

d.) *Approximate the value of x^* numerically to 5 correct decimal digits using an iteration method.*

Solution:

We can approximate x^* by the following script

```python
import numpy as np
import scipy.linalg as linalg
import sys
import numpy as np


def g(x):
    return ((x**2 + 1) * np.arctan(x)) / (x)


def dg(x):
    return 1 / x + np.arctan(x) - np.arctan(x) / (x**2)


x_0 = 1.0
epsilon = 1
error = 1e-5
x_n = 0

while epsilon > error:
    x_n = x_0 - (g(x_0) - 2) / dg(x_0)
    epsilon = abs(x_n - x_0)
    x_0 = x_n

print(f"x_star: {x_0:.6f}")
```

where we have implemented Newtons method on $k(x) = g(x) - 2$. This gives ut the following approximated value

```
print(f"x_star: {x_0:.6f}")
x_star: 1.391745
```

e.) *Use the results in 1 b) and c) to show that the Newton sequence diverges if x_0 is far from ξ, and describe the largest possible non-empty interval (a, b) that contains ξ and satisfies that*

$$\lim_{k\to\infty} x_k = \xi \quad \text{for all Newton sequences with} \quad x_0 \in (a, b)$$

Solution:

Let $\varphi(x) = x - \frac{f'(x)}{(f(x)} = x - (x^2 - 1)\arctan(x)$, and define the functional iteration $x_{n+1} = \varphi(x_n)$ (as in (a.)). Remember that φ is odd, $\varphi(0) = 0$, and $\varphi'(x) < 0$ meaning $\varphi(x)$ is strictly decreasing.

Now let x^* be a unique positive integer (like in (c)) satisfying $\varphi(x^*) = -x^*$. (since φ is odd) If $0 < |x| < |x^*| \Rightarrow |\varphi(x) \leq |x|$. Hence if $|x_0| < |x^*| \Rightarrow |x_{n+1}| \leq |x_n|$, $\forall n$ and $|x_n| \to \xi$ since φ is strictly decreasing and $\varphi(0) = 0$ which by continuity of φ implies that $\varphi(\xi) = \xi$.

If $|x_0| = |x^*|$ then $\{x_n\} = \{x_0, -x_0, x_0, \ldots, \}$ which would neither converge nor diverge.

In the last case suppose now that $0 < |x^*| < |x|$, which would imply $\varphi(x) \geq |x|$, meaning x_n is now non-decreasing. Nor is x_n is bounded (neither from below or above) which then implies that $|x_n| \to \infty$.

Hence to conclude must $x_0 \in (-x^*, x^*)$ for $|x_n| \to \xi$. ■

f.) *Compute the sequence* $(x_k)_{k=0}^6$ *with Newton's method*

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

We compute $(x_k)_{k=0}^6$ by the following script

```
import numpy as np
import scipy.linalg as linalg
import sys
import numpy as np
import math

def f(x):
   return np.arctan(x)


def f_prime(x):
   return 1 / (1 + x**2)


def newton_method_arctan(x0, tolerance=1e-13, max_iterations=100):
   x = x0
   for i in range(max_iterations):
      fx = f(x)
      if abs(fx) < tolerance:
         return x, i
      x = x - fx / f_prime(x)
   return None, max_iterations


# Initial guess for the root
x0 = 1.3

# Call Newton's method to find the root
root, iterations = newton_method_arctan(x0)

if root is not None:
   print(f"Approximate root: {root}")
   print(f"Number of iterations: {iterations}")
else:
   print("Newton's method did not converge within the specified tolerance.")
```

which gives the following output .

```
x_0 = 1.3
---------
Approximate root: 1.2045171040057576e-14
Number of iterations: 6

x_0 = 1.4
---------
Newton's method did not converge within the specified tolerance.
```

This aligns well with the theoretical results from a.), c.) and e.) where we can clearly see that $x^* \in (1.3, 1.4)$ somewhere.[2]

Problem 2

Compute the QR factorization of the matrix

$$A = \begin{bmatrix} 1 & 1 \\ 0 & -1 \\ 1 & 1 \end{bmatrix}$$

and use the factorization to find the least squares solution of the following equation

$$Ax = \underbrace{\begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}}_{=b}$$

Solution:

Let $a_1 = (1, 0, 1)^T, a_2 = (1, -1, 1)^T$. By applying Gram-Schmidt it follows that

$$\begin{aligned}
\boldsymbol{u}_1 &= a_1 = (1, 0, 1) \\
\boldsymbol{e}_1 &= \frac{\boldsymbol{u}_1}{\|\boldsymbol{u}_1\|} = (\sqrt{2}^{-1}, 0, \sqrt{2}^{-1}) \\
\boldsymbol{u}_2 &= a_2 - (a_2 \cdot \boldsymbol{e}_1)\boldsymbol{e}_1 \\
&= (1, -1, 1) - ((1, -1, 1) \cdot ((\sqrt{2}^{-1}, 0, \sqrt{2}^{-1})(\sqrt{2}^{-1}, 0, \sqrt{2}^{-1}) \\
&= (1, -1, 1) - \left(\frac{2}{\sqrt{2}^2}, 0, \frac{2}{\sqrt{2}^2}\right) \\
&= (1, -1, 1) - (1, 0, 1) \\
&= (0, -1, 0) \\
\boldsymbol{e}_2 &= \boldsymbol{u}_2.
\end{aligned}$$

[2] To be completely exact, the value $x^* \in (-1.391745..., 1.391745...)$

Then by the principles of QR factorization we have that

$$Q = \begin{bmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 \end{bmatrix}$$

$$R = \begin{bmatrix} a_1\boldsymbol{e}_1 & a_2\boldsymbol{e}_1 \\ 0 & a_2\boldsymbol{e}_2 \end{bmatrix}.$$

To solve the least squares problem we remember that Q is an orthonormal matrix which means that $Q^T = Q^{-1}$, so we can thus re-write the problem as

$$Ax = b \iff QRx = b \iff Rx = Q^T b.$$

Then insert for Q, R, b

$$\begin{bmatrix} \sqrt{2} & \sqrt{2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \sqrt{2}/2 & 0 & \sqrt{2}/2 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}$$

and solve the following linear set of equations

$$\sqrt{2}x_1 + \sqrt{2}x_2 = \sqrt{2} + \frac{3}{2}\sqrt{2} \tag{3}$$

$$x_2 = -1 \tag{4}$$

which gives that

$$x = \begin{bmatrix} 3.5 \\ -1 \end{bmatrix}.$$

Can also solve this on the computer by the following script

```
import numpy as np
import scipy.linalg as linalg
import sys
import numpy as np

A = np.array([[1.0, 0, 1], [1, -1, 1]])
B = np.array([2, 1, 3])
x = np.linalg.lstsq(A.T, B)
print(x[0])
```

which gives the following output .

```
 x = np.linalg.lstsq(A.T, B)
[ 3.5 -1.]
```

Problem 3

Let $PA = LU$ be an LU factorization of A, where P is a permutation matrix, L is a unit lower triangular matrix and U is upper triangular. Explain how the matrices P, L and

U can be used to construct a more efficient method for computing det(A) and compute the computational cost of your approach. You may use that the cost of the LU factorization of A is $\frac{2n^3}{3} + \mathcal{O}(n^2)$ *arithmetic operations, the cost of det(P) is* $\mathcal{O}(n^2)$ *arithmetic operations and that* $det(P) \neq 0$ *(it always takes one of the values* ± 1*).*

Solution:

Given that A has an LU factorisation we can write the determinant of A as

$$\det(A) = \det(P)\det(L)\det(U)$$

where L is a unit lower triangular matrix, U an upper triangular matrix and P a permutation matrix. The cost of the product $\det(L)\det(U) = u_{1,1} \cdot u_{2,2} \cdots u_{n,n}$ is only $\mathcal{O}(n)$ since it is the product of the main diagonal of U, and by definition we had that the cost of $\det(P)$ is $\mathcal{O}(n^2)$. Hence will the product yield a total of $\mathcal{O}(n^3)$ calculations, which if we take the LU factorization into account gives a totat cost of

$$\begin{aligned}
\mathcal{C}_T &= \mathcal{C}(LU) + \mathcal{C}(\det(P)\det(L)\det(U)) \\
&= \frac{2n^3}{3} + \mathcal{O}(n^2) + \mathcal{O}(n^2)\mathcal{O}(n) \\
&= \frac{2n^3}{3} + \mathcal{O}(n^2) + \mathcal{O}(n^3) \\
&= \frac{2n^3}{3} + \mathcal{O}(n^3 + n^2) \\
&= \frac{2n^3}{3} + \mathcal{O}(n^3).
\end{aligned}$$

[3] The cost of determining the determinant of A through PLU decomposition is thus of cubic time complexity and

$$\frac{2n^3}{3} + \mathcal{O}(n^3) << e^1 n! + \mathcal{O}((n-1)!)$$

which shows that the method is a lot more efficient. ■

[3] I have decided to write $\mathcal{O}(n^3)$ and not $\mathcal{O}(n^3 + n^2)$ since $\mathcal{O}(n^3)$ dominates $\mathcal{O}(n^2)$