# APPM 4600 Homework 4

Becca Blum

2/8/2024

Please note, this homework is incomplete due to me using most of my time to work on a research paper for an AIAA conference. I'll be back on track next week!

## Github

https://github.com/BeccaBlum/APPM-4600/tree/main/Homework/HW4

## Question 1

We want to find a solution located near (x, y) = (1, 1) to the nonlinear set of equations

$$f(x, y) = 3x^2 - y^2 = 0$$
$$g(x, y) = 3xy^2 - x^3 - 1 = 0$$

**(a)** We are given the algorithm below to iterate on to find the solution to this system, starting with x = y = 1. How well does it converge?

$$\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} - \begin{bmatrix} 1/6 & 1/18 \\ 0 & 1/6 \end{bmatrix} \begin{bmatrix} f(x_n, y_n) \\ g(x_n, y_n) \end{bmatrix}, \quad n = 0, 1, 2, \ldots$$

```
Root is x = 0.500000000040521, y = 0.8660254038535676
Number of iterations: 33
Error message: 0
```

**Fig. 1 Output of given iteration method**

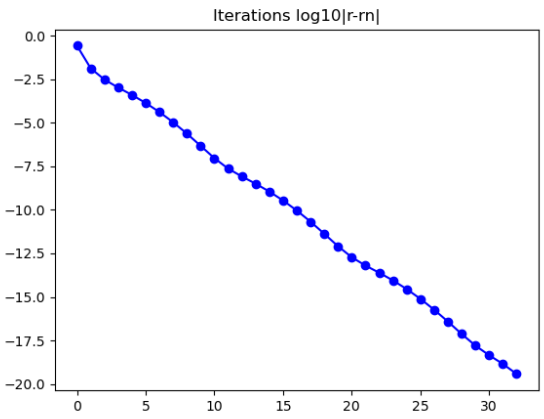



**Fig. 2 Error of given integration method**

The method converges linearly, in 33 iterations.

**(b)** Provide some motivation for the particular choice of the numerical 2×2 matrix in the equation above. My guess was that the matrix is the inverse of the Jacobian evaluated at the initial point, x=y=0.

$$\begin{bmatrix} 6x & -2y \\ 3y^2 + 3x^2 & 6xy \end{bmatrix}$$

$$\begin{bmatrix} 6(1) & -2(1) \\ 3(1)^2 - 3(1)^2 & 6(1)(1) \end{bmatrix}$$

$$\begin{bmatrix} 6 & -2 & 1 & 0 \\ 0 & 6 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1/3 & 1/6 & 0 \\ 0 & 1 & 0 & 1/6 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 1/6 & 1/18 \\ 0 & 1 & 0 & 1/6 \end{bmatrix}$$

After calculating the inverse of the initial Jacobian, it appears this is the matrix in the given iteration formula.

**(c)** Iterate on the system of equations using Newton's method, using the same starting approximation x0 = y0 = 1. How well does this converge?

```
Newton Method
Root is x = 0.5, y = 0.8660254037844386
Number of iterations: 5
Error message: 0
```

**Fig. 3**

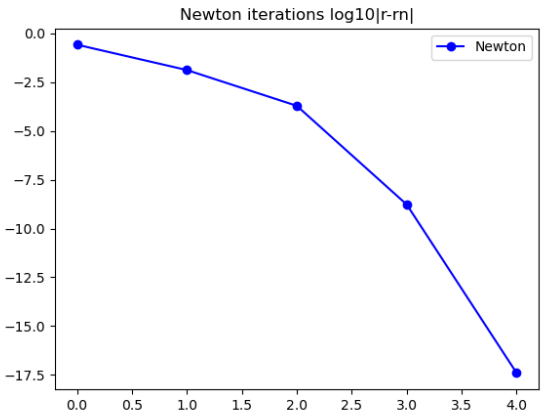



**Fig. 4 Error of Newton method**

Newton's method converges quadratically and in only 5 iterations. This makes sense as the Jacobian is recalculated every iteration. However, this is more expensive computationally.

**(d)** Here we are asked to spot the exact solution from the numerical result and then verify it analytically.

The approximated roots imply that the exact solution is:

$$x = \cos(\pi/3) = 0.5$$
$$y = \sin(\pi/3) = \sqrt{3}/2 \approx 0.866025403784$$

Of course the 0.5 could simply be 0.5, but I immediately thought of the sine and cosine of $\pi/3$ from using them over and over again for the past 10 years. We can confirm that this is the solution by entering the points into the original functions:

$$\begin{aligned} 3x^2 - y^2 &= 0 \\ 3(0.5)^2 - (\sqrt{3}/2)^2 &= 0 \\ \frac{3}{4} - \frac{3}{4} &= 0 \\ 0 &= 0 \end{aligned}$$

$$\begin{aligned} 3xy^2 - x^3 - 1 &= 0 \\ 3(0.5)(\sqrt{3}/2)^2 - (0.5)^3 - 1 &= 0 \\ 3(0.5)(\frac{3}{4}) - (\frac{1}{8}) - 1 &= 0 \\ (\frac{9}{8}) - (\frac{1}{8}) - 1 &= 0 \\ 1 - 1 &= 0 \\ 0 &= 0 \end{aligned}$$

Both equations evaluate to zero at the assumed exact solution, so x* = 0.5 and y* = $\sqrt{3}/2$.

## Question 2

Here we are considering the nonlinear system below which has two real solutions. We are asked to use the Newton, Lazy Newton, and Broyden to approximate the solutions with the given initial guesses. Which methods perform the best?

$$f(x, y) = x^2 + y^2 - 4 = 0$$
$$g(x, y) = e^x + y - 1 = 0$$

**(i)** $x_0 = 1, y_0 = 1$

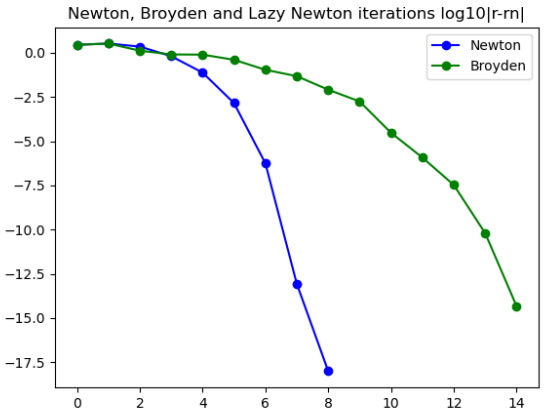



**Fig. 5 Error comparison**

```
Q2.i

Newton Method
Root is x = -1.8162640688251506, y = 0.8373677998912478
Number of iterations: 8
Error message: 0
```

**Fig. 6 Output of Newton method**

```
Broyden method converged, n=15, |F(xn)|=2.2e-16
```

**Fig. 7 Output of Broyden method**

**(ii)** $x_0 = 1, y_0 = -1$

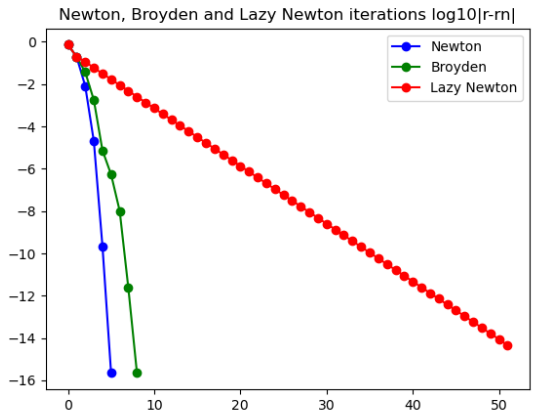



**Fig. 8 Error comparison**

```
Q2.ii

Newton Method
Root is x = 1.0041687384746594, y = -1.72963728702587
Number of iterations: 5
Error message: 0

Lazy Newton Method
Root is x = 1.004168738474661, y = -1.7296372870258745
Number of iterations: 50
Error message: 0
```

**Fig. 9 Output of Newton and Lazy Newton methods**

```
Broyden method converged, n=9, |F(xn)|=9.9e-16
```

**Fig. 10 Output of Broyden method**

**(iii)** $x_0 = 0, y_0 = 0$

Both the Newton and Lazy Newton methods fail due to the initial Jacobian being singular. The Broyden method fails due to an overflow or underflow error (inf or NAN, according to the error message).

Overall, we can see that the Newton method performs the best as it converges faster than both the Lazy Newton and Broyden methods. The Lazy Newton converges linearly and is both the slowest and least successful for the given initial guesses. The Broyden method converges almost as fast as Newton method.

## Question 3

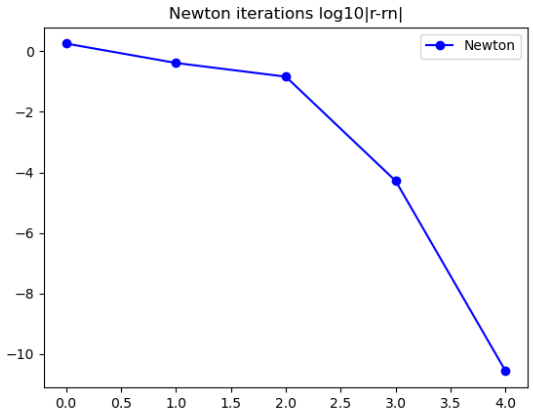
Here we are considering the nonlinear system

$$\begin{aligned} x + \cos(xyz) - 1 &= 0 \\ (1-x)^{1/4} + y + 0.05z^2 - 0.15z - 1 &= 0 \\ -x^2 - 0.1y^2 + 0.01y + z - 1 &= 0 \end{aligned}$$

We are asked to test the following three techniques for approximating the solution to the nonlinear system to within 10-6.

**(a)** Newton method

```
Newton Method
Newton, root is x = 4.0246940060813654e-17, y = 0.09999999999999988,z = 1.0
Number of iterations: 5
Error message: 0
```




**(b)** Steepest descent method

**(c)** First Steepest descent method with a stopping tolerance of 5 × 10-2. Use the result of this as the initial guess for Newton's method.

**(d)** Using the same initial guess, which technique converges the fastest? Try to explain the performance

## Question 4

We are given the following interpolation data with a degree 4 polynomial p(x):

| $x_j$ | 0 | 2 | 3 | 5 | 8 |
|---|---|---|---|---|---|
| $y_j$ | -125 | -27 | -8 | 0 | 27 |

**(a)** Write p(x) using the Lagrange polynomial basis for this problem.

$$p(x) = -125\left(\frac{\prod_{j\neq 0}(x-x_j)}{\prod_{j\neq 0}(2-x_j)}\right) - 27\left(\frac{\prod_{j\neq 1}(x-x_j)}{\prod_{j\neq 1}(3-x_j)}\right) - 8\left(\frac{\prod_{j\neq 2}(x-x_j)}{\prod_{j\neq 2}(5-x_j)}\right) + 0 + 27\left(\frac{\prod_{j\neq 4}(x-x_j)}{\prod_{j\neq 4}(8-x_j)}\right)$$

**(b)** Find the coefficients for Newton interpolation, and write p(x) as a linear combination of Newton polynomials.

**(c)** This data was sampled from $f(x) = (x-5)^3$. Explain what this means in terms of differences of order 4 or higher (like the last coefficient above) using Newton interpolation.

## Python code

Python code is located in my GitHub as it is very long.