HW 3 - 151A

ASHER CHRISTIAN 006-150-286

1. Exercise 1

(a) Let $a > 0$. Verify that $\sqrt{a}$ is a fixed point of the function

$$g(x) = \frac{1}{2}(x + \frac{a}{x}).$$

clearly

$$g(\sqrt{a}) = \frac{1}{2}(\sqrt{a} + \frac{a}{\sqrt{a}}) = \frac{1}{2}(\sqrt{a} + \sqrt{a}) = \sqrt{a}.$$

(b) Assume that, for some p_0 the fixed point iteration $p_{n+1} = g(p_n)$ converges to $p = \sqrt{a}$ as $n \to \infty$ Determine the order of convergence. This is the same question as the order of convergence for fixed point iteration which was shown to converge linearly in lecture.

2. Exercise 2

Consider the function $f(x) = e^x - 1 - x - \frac{x^2}{2}$
(a) show that $x = 0$ is a root of f. What is the multiplicity of this root?
$f(0) = e^0 - 1 - 0 - \frac{0^2}{2} = 0$ so 0 is a root

$$f'(x) = e^x - 1 - x \to f'(0) = e^0 - 1 - 0 = 0.$$
$$f''(x) = e^x - 1 \to f''(0) = e^0 - 1 = 0.$$
$$f'''(x) = e^x \to f'''(0) = e^0 \neq 0.$$

Thus f has a root at $x = 0$ of multiplicity 3
Implement both Newton's method and modified Newton's method. Then, use both the modified Newton's method and the usual newton's method to calculate the root with initialization $x_0 = 1$. Stop the iteration when $|f(x)| < 10^{-10}$ or $|f(x)| > 10^5$ or number of iterations = 100. report iterations and calculated root with 8 sig digits

```
#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;


double EPSILON = pow(10,-10);
double DELTA = pow(10, 5);
double MAX_ITERATIONS = 100;

double newtons_method(double p0, double (*f)(double),
    double (*fp)(double));
```

Date: 01.02.25.

```
double mod_newton_method(double p0, double (*f)(double)
    , double (*fp)(double), double (*fpp)(double));
double f(double x);
double fp(double x);
double fpp(double x);

int main(void){
    double s0 = 1;
    cout << setprecision(16);
    newtons_method(s0,f,fp);
    mod_newton_method(s0,f,fp,fpp);
}

double f(double x){
    return exp(x)-1-x-(pow(x,2)/2);
}

double fp(double x){
    return exp(x)-1-x;
}

double fpp(double x){
    return exp(x) - 1;
}

double newtons_method(double p0, double (*f)(double),
    double (*fp)(double))
{
    double p = p0;
    int iterations = 0;
    while(abs(f(p)) >= EPSILON && abs(f(p)) <= DELTA &&
         iterations < 100){
        p = p - (f(p)/fp(p));
        iterations += 1;
    }
    cout << "Newton's Method starting at p0 = " << p0
        << ", iterations: " << iterations;
    cout << " p estimate: " << p << " f(p) final: " <<
        f(p) << endl << endl;
    return p;
}
double mod_newton_method(double p0, double (*f)(double)
    , double (*fp)(double), double (*fpp)(double)){
    double p = p0;
    int iterations = 0;
    while(abs(f(p)) >= EPSILON && abs(f(p)) <= DELTA &&
         iterations < 100){
        p = p - ((f(p)*fp(p))/(pow(fp(p),2) - (f(p)*fpp
            (p))));
        iterations += 1;
    }
    cout << "Modified Newton's Method starting at p0 =
        " << p0 << ", iterations: " << iterations;
```

```
        cout << " p estimate: " << p << " f(p) final: " <<
            f(p) << endl << endl;
        return p;
    }
```

With the following output

```
Newton's Method starting at p0 = 1, iterations: 18 p
    estimate: 0.0007743951257744384 f(p) final:
    7.741430383911001e-11
Modified Newton's Method starting at p0 = 1, iterations
    : 3 p estimate: -8.80281240268465e-08 f(p) final:
    2.583358545641916e-17
```

Since $f'(0)$, newton's method converges linearly whereas modified newton's method converges quadratically This explains why newton's method took 6 times as many iterations without converging as close as the modified newton's method converged.

3. Exercise 3

The sequence defined by $p_n = \log(1 + 2^{-n})$ $n \geq 0$ *converges linearly to* $p = 0$ *report the first four terms of* p_n *and the results from the first two iterations of the sequence* $\hat{p_n}$ *using Aitken's* Δ^2 *method with 4 digits after decimal. which converges faster?* Using the sequence

$$
\begin{aligned}
p_0 &= 0.3010 \\
p_1 &= 0.1761 \\
p_2 &= 0.0969 \\
p_3 &= 0.0512
\end{aligned}
$$

Using aitken's Δ^2 method.

$$
\begin{aligned}
\hat{p_0} &= p_0 - \frac{(p_1 - p_0)^2}{p_2 - 2p_1 + p_0} = -0.0401 \\
\hat{p_1} &= p_1 - \frac{(p_2 - p_1)^2}{p_3 - 2p_2 + p_1} = -0.0115
\end{aligned}
$$

Aitken's method converges much faster.

4. Exercise 4

Let $f(x) = ln(x)$ *(a) write down the lagrange polynomial for* f *passing through the points* $(1, \ln 1), (2, \ln 2), (3, \ln 3)$

$$
\begin{aligned}
p(x) &= \frac{(x-2)(x-3)}{(1-2)(1-3)} \ln 1 + \frac{(x-1)(x-3)}{(2-1)(2-3)} \ln 2 + \frac{(x-1)(x-2)}{(3-1)(3-2)} \ln 3. \\
&= x^2 \left(\ln \left(\frac{3^{\frac{1}{2}}}{2} \right) \right) + x \left(\ln \left(2^4 \cdot 3^{-\frac{3}{2}} \right) \right) + \ln \left(3 \cdot 2^{-3} \right).
\end{aligned}
$$

Approximate $\ln(1.5), \ln(2.4)$ *and report absolute error*

$$p(1.5) = 0.3825 \quad \ln(1.5) = 0.4055 \quad |p(1.5) - \ln(1.5)| = 0.022931.$$

$$p(2.4) = 0.8898551 \quad \ln(2.4) = 0.875469 \quad |p(2.4) - \ln(2.4)| = 0.014386.$$

Give an upper bound for the absolute error on the interval [1,3]
We know that for each $x \in [1,3]$ there exists a number $\xi(x)$ between 1 and 3 such that

$$\ln(x) = p(x) + \frac{f^{(3)}(\xi(x))}{3!}(x-1)(x-2)(x-3).$$
$$|\ln(x) - p(x)| = |\frac{2}{3!\xi(x)^3}(x-1)(x-2)(x-3)|.$$

with $1 < \xi(x) < 3$

$$\leq |\frac{1}{3}(x-1)(x-2)(x-3)|.$$

This function achieves its absolute maximum which can be verified by taking the derivative and solving for 0 at $x = 2 \pm \frac{\sqrt{12}}{6}$

$$\leq 0.1283000598.$$

5. Exercise 5

(a)

$$\{(1,1),(2,3),(3,1)\}.$$

$p(x) = a_0 + a_1 x + a_2 x^2$

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix} \vec{a} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} \vec{f} = \begin{pmatrix} 1 \\ 3 \\ 1 \end{pmatrix} A\vec{a} = \vec{f}.$$

using row reduction we get

$$\vec{a} = \begin{pmatrix} -5 \\ 8 \\ -2 \end{pmatrix}.$$

$n+1$ *data points* $\{(x_i, f(x_i))\}_{i=0}^{n}$ *and*

$$p(x) = \sum_{k=0}^{n} a_k x^k.$$

Similar to before

$$A = \begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{pmatrix} \vec{a} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} \vec{f} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}.$$

If all x_i are unique then by the existence of the lagrange interpolant polynomial we know there exists some $a_0, a_1, ..., a_n$ that satisfy the equation, additionally we have shown that such a polynomial is unique because if it were not then subtracting two polynomials that satisfy the equation would create a polynomial with $n+1$ roots despite being of degree less than or equal to n and so it would be 0. Thus the equation has one and only one solution and the corresponding matrix is invertible.