Sarah Mahl

EXAM #1

→ READ INSTRUCTIONS BETTER, CHECK ALGEBRA

1. $x^4 - x - 2 = 0$

(a) ① $a = 0, b = 2 \Rightarrow c = \frac{1}{2}(a+b) = \frac{1}{2}(0+2) = 1$

*plugging c into f(x) checks how close c is to the exact root

$f(c) = (1)^4 - (1) - 2 = 1 - 1 - 2 = -2$

② $b_2 = 1 \Rightarrow c = \frac{1}{2}(a+b_2) = \frac{1}{2}(0+1) = \frac{1}{2}$

$f(\frac{1}{2}) = (\frac{1}{2})^4 - (\frac{1}{2}) - 2 = \frac{1}{16} - \frac{1}{2} - \frac{4}{2} = \frac{1}{16} - \frac{3}{2} = \frac{1}{16} - \frac{24}{16}$

$= \boxed{-\frac{23}{16}} \approx \boxed{-1.4375}$

(b) $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$ for $n = 0, 1, 2, \ldots$

$x_0 = 0$

$f'(x) = 4x^3 - 1$

$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 0 - \frac{0^4 - 0 - 2}{4(0)^3 - 1} = -\left(\frac{-2}{-1}\right) = -2$

$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$

$f(x_1) = f(-2) = (-2)^4 - (-2) - 2 = 16 + 2 - 2 = 16$

$f'(x_1) = f'(-2) = 4(-2)^3 - 1 = 4(-8) - 1 = -32 - 1 = -33$

$x_2 = -2 - \left(\frac{16}{-33}\right) = -\frac{2}{1} + \frac{16}{33} = -\frac{66}{33} + \frac{16}{33} = \boxed{-\frac{50}{33}} \approx \boxed{-1.515}$

(c) Just using two approximations of the bisection method and Newton's method, I found that the bisection method was the most accurate (since $-\frac{23}{16} \approx -1.4375$ is closer to the root of $f(x)$ at $x \approx 1.3521$ than $-\frac{50}{33} \approx -1.515$).

(d) $f''(x) = 12x^2$, $f'''(x) = 24x$

$\hookrightarrow f''(-1) = 12(-1)^2 = 12 \neq 0$

f(x) has roots of multiplicity one

(b) $|r - x_{n+1}| \leq k|r - x_n|^2 \Rightarrow |1.3521 + 2| \leq k|1.3521 - 0|^2$

$3.3521 \leq 1.828k$

(a) $|1.3521 + 1.4375| \leq k|1.3521 + 2|^2$

$2.7896 \leq 11.2366k$

X Newton converges quadratically. Bisection converges linearly

Assuming that both methods converge to the exact value of I, the bisection method will converge more rapidly as the number of iterations increase. This is because the bisection method for $f(x) = x^4 - x - 2$ converges quadratically, as shown above for its root $x \approx 1.3521$.

* ALGEBRA

2. $I = \int_1^5 \frac{24}{3+x}\,dx = 24\int_1^5 \frac{1}{x+3}\,dx$

(a) $N=2$

$I = \Delta x\left[\frac{1}{2}(f(a)+f(b)) + \sum_{j=1}^{N-1} f(x_j)\right]$, $\Delta x = \frac{b-a}{N}$

$\Delta x = \frac{5-1}{2} = \frac{4}{2} = 2$

$f(a) = f(1) = \frac{24}{3+1} = \frac{24}{4} = 6$

$f(b) = f(5) = \frac{24}{3+5} = \frac{24}{8} = 3$

$I = 2\left[\frac{1}{2}(6+3) + \sum_{j=1}^{2-1} f(x_j)\right] = 9 + 2\sum_{j=1}^{1} f(x_j)$

$= 9 + 2f(x_1)$

↳ let $x_0 = 0 \Rightarrow x_1 = 2$

$= 9 + 2f(2) = 9 + 2\left(\frac{24}{2+3}\right) = 9 + 2\left(\frac{24}{5}\right) = 9 + \frac{48}{5}$

$= \frac{45}{5} + \frac{48}{5} = \boxed{\frac{93}{5}} = \boxed{18.6}$

(b) error $< 1.0\times10^{-8}$

$\text{error}_{trap} = \frac{(b-a)^3}{12N^2}|f''(c^*)|$

$1.0\times10^{-8} > \frac{(5-1)^3}{12N^2}k$, $|f''(c^*)| \doteq k$

$12.0\times10^{-8}N^2 > 64k$

$N^2 > \frac{64}{12.0\times10^{-8}}k$

$k = \max_{c^*\in[a,b]} |f''(c^*)| = \frac{3}{4}$

↳ $f'(x) = -\frac{24}{(x+3)^2}$

$f''(x) = \frac{48}{(x+3)^3}$

$f''(a) = f''(1) = \frac{48}{4^3} = \frac{48}{64} = \frac{3}{4}$

$f''(b) = f''(5) = \frac{48}{8^3} = \frac{48}{512} = \frac{3}{32}$

$N^2 > \frac{64}{12.0\times10^{-8}}\left(\frac{3}{4}\right)$

$N^2 > 4.0\times10^{9}$

$N > \sqrt{4.0\times10^{9}} \approx 63{,}245.6$

To ensure that the trapezoidal rule provides an approximation to I that has an error less than 1.0×10^{-8}, the number of subintervals N must be greater than $\sqrt{4.0\times10^{9}} \approx 63{,}245.6$.

(c) Because the trapezoid rule converges quadratically while the left-endpoint rule only converges linearly, the trapezoid rule will converge more rapidly.

3. $J = \int_1^3 \int_2^7 \cos^2(x^2+y^3)\,dx\,dy$

```
import numpy as np
def f(x,y):
    return (np.cos(x**2 + y**3))**2

def MonteCarlo2D(a,b,c,d, Nx, Ny):
    # a = 2, b=7, c=1, d=3
    deltax = (b-a)/Nx
    deltay = (d-c)/Ny
    sum=0.0
    x=a
    y=c

    for i in range(Ny):
        for j in range(Nx):
            sum += f(random.uniform(~~x~~ a,b),
                     random.uniform(~~y~~ c, d))
            x ± deltax
        y ± deltay

    return deltax * deltay*sum
```

* setting to x & y will shorten interval the random # is taken from, which we don't want

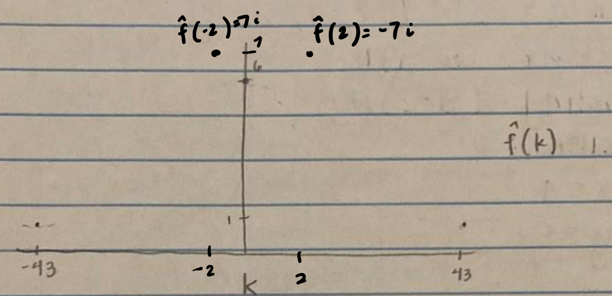
4. $g(x) = 6 + 2\cos(43x) + 14\sin(2x)$, $x \in [0, 2\pi]$

(a) $\hat{f}(k) = \frac{1}{L}\int_0^L f(x) e^{-\frac{2\pi i k x}{L}}$

→ used Mathematica because I ran out of time

$$\hat{f}(k) = \begin{cases} -7i, & k = 2 \\ 7i, & k = -2 \\ i, & k = \pm 43 \\ 6, & k = 0 \end{cases}$$

(b)



(c)
```
import numpy as np
N = 100
L = 2*np.pi
X_n = [n*(L/N) for n in range(0, N)]
fvec = [(6+np.cos(43*x_n[j]+12* np.sin(2*x_n[j]))
        for j in range((len(x_n)))]
Fhats = np.fft.fft(fvec)
```

(d) To accurately reconstruct the discrete version of g, ~~infinitely~~ 87 many Fourier coefficients are required. This is because g is periodic. 87 points in $k \in [-43, 43]$ (not including imag. $\hat{f}$'s)

(e) The Fourier coefficients obtained with an FFT would ~~converge to the exact values depending on the amount of iterations N used to compute the Fourier coefficients.~~ differ by roundoff error since g is periodic (g would need more than the 87 $\hat{f}$'s to exactly reconstruct g)

False. Size of a matrix doesn't determine convergence (though it does affect speed)

5. (a) ~~True~~. Newton's method for a system of equations requires the inverse of the Jacobian matrix, which is an $\mathcal{O}(N^3)$ operation. Because Sally is solving a system of $N = 2000$ equations and Jimmy is solving a system of $N = 1000$ equations, sally will see a slower rate of convergence.

True

(b) For $Ax = b$ where neither Jacobi nor Gauss-Seidel converge, an example could include an A that is neither diagonally dominant nor positive definite.

$$\begin{pmatrix} 1 & 6 \\ 3 & 10 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

positive diagonal elements, but are the evals positive & real? also not symmetric

ex.
$$\begin{pmatrix} 1 & 5 \\ 5 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$