Bryan Jay
260738764

ECSE 443 - Assignment 4

Question 1

The following function was given to calculate the integral using the various methods.

$$I = \int_0^{\pi} \ln(5 - 4\cos(x))dx$$

The actual value was found in Matlab to be $actual = 4.355172180607203$. For each part we used the relative error as the stopping condition.

$$Relative\ Error = \frac{|calculated - theoretical|}{theoretical}$$

a) Refer to Appendix A, part a for the corresponding Matlab code that was written. The following formula was followed to calculate he area under the curve at each segment using the midpoint rule.

$$M_n = \frac{b-a}{n}(f(m_1) + f(m_2) + \ldots f(m_n)),$$

$$m_k = \frac{t_k + t_{k-1}}{2} = a + \frac{2k-1}{2n}(b-a).$$

Number of Segments (Midpoint)	9
I	4.355173512185210
Relative Error	3.057463517027002e-07

b) Refer to Appendix A, part b for the corresponding Matlab code that was written. The following formula was followed to calculate he area under the curve at each segment using the trapezoidal rule.

$$T(f) = \frac{b-a}{2}(f(a) + f(b))$$

$$\int_a^b f(x)\,dx \approx \sum_{k=1}^{N} \frac{f(x_{k-1}) + f(x_k)}{2}\Delta x_k$$

$$\Delta x_k = \Delta x = \frac{b-a}{N}$$

where:

Number of Segments (Trapezoidal)	9
I	4.355170849024120
Relative Error	3.057475180346966e-07

c) Refer to Appendix A, part c for the corresponding Matlab code that was written. The following formula was followed to calculate he area under the curve at each segment using the Simpsons 1/3 rule:

$$\int_a^b P(x)\,dx = \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right].$$

Number of Segments (Simpsons)	8
I	4.355174177932306
Relative Error	4.586099053484477e-07

Question 2

The following function was given to calculate the integral using the various methods.

$$I = \int_0^3 \int_x^{2x^3} (x^2 + y)dydx$$

The actual value was found in Matlab to be $actual = \ 7.905357142861429\ e + 02$. For each part we used the relative error as the stopping condition.

$$Absolute\ Error = |calculated - theoretical|$$

For the following 3 parts the code was written with the same formulas referred to above.

a) Refer to Appendix B, part a for the corresponding Matlab code that was written. The following formula was followed to calculate he area under the curve at each segment using the midpoint rule. To account for the double integral, we repeated the loop to find respect sum under the curve corresponding to each axis. Therefore, the integration was done twice, and a new function was created after the first integration that was used for the second integration.

$$M_n = \frac{b-a}{n}(f(m_1) + f(m_2) + \ldots f(m_n)),$$

$$m_k = \frac{t_k + t_{k-1}}{2} = a + \frac{2k-1}{2n}(b-a).$$

Number of Segments (Midpoint)	1149
I	790.535614364064539381218663817 86
Absolute Error	9.992207839443421e-05

b) Refer to Appendix b, part b for the corresponding Matlab code that was written. The following formula was followed to calculate he area under the curve at each segment using the trapezoidal rule. To account for the double integral, we repeated the loop to find respect sum

under the curve corresponding to each axis. Therefore, the integration was done twice, and a new function was created after the first integration that was used for the second integration.

$$T(f) = \frac{b-a}{2}(f(a) + f(b))$$

$$\int_a^b f(x)\,dx \approx \sum_{k=1}^{N} \frac{f(x_{k-1}) + f(x_k)}{2} \Delta x_k$$

$$\Delta x_k = \Delta x = \frac{b-a}{N}$$

where:

Number of Segments (Trapezoidal)	1625
I	790.535814198928248063879234500 47
Absolute Error	9.991278531424845e-05

c) Refer to Appendix B, part c for the corresponding Matlab code that was written. The following formula was followed to calculate he area under the curve at each segment using the Simpsons 1/3 rule. To account for the double integral, we repeated the loop to find respect sum under the curve corresponding to each axis. Therefore, the integration was done twice, and a new function was created after the first integration that was used for the second integration.

$$\int_a^b P(x)\,dx = \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right].$$

Number of Segments (Simpsons)	12
I	790.535800684408668409922267946 96
Absolute Error	8.639826573459449e-05

Question 3

a) To calculate the fifth order backward difference with first order accuracy. We begin with the known function.

$$f'(x) = \frac{f(x) - f(x-h)}{h}$$

The fifth order derivative is calculated by integrating this formula recursively.

$$f''(x) = \frac{f'(x) - f'(x-h)}{h}$$

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

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

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

$$f'''(x) = \frac{\frac{f(x) - f(x-h)}{h} - 2\frac{f(x-h) - f(x-2h)}{h} + \frac{f(x-2h) - f(x-3h)}{h}}{h^2}$$

$$f'''(x) = \frac{f(x) - 3f(x-h) + 3f(x-2h) - f(x-3h)}{h^3}$$

$$f^{(4)}(x) = \frac{f'(x) - 3f'(x-h) + 3f'(x-2h) - f'(x-3h)}{h^3}$$

$$f^{(4)}(x) = \frac{\frac{f(x) - f(x-h)}{h} - 3\frac{f(x-h) - f(x-2h)}{h} + 3\frac{f(x-2h) - f(x-3h)}{h} - \frac{f(x-3h) - f(x-4h)}{h}}{h^3}$$

$$f^{(4)}(x) = \frac{f(x) - 4f(x-h) + 6f(x-2h) - 4f(x-3h) + f(x-4h)}{h^4}$$

$$f^{(5)}(x) = \frac{f'(x) - 4f'(x-h) + 6f'(x-2h) - 4f'(x-3h) + f'(x-4h)}{h^4}$$

$$f^{(5)}(x) = \frac{\frac{f(x) - f(x-h)}{h} - 4\frac{f(x-h) - f(x-2h)}{h} + 6\frac{f(x-2h) - f(x-3h)}{h} - 4\frac{f(x-3h) - f(x-4h)}{h} + \frac{f(x-4h) -}{h}}{h^4}$$

$$\boldsymbol{f^{(5)}(x) = \frac{f(x) - 5f(x-h) + 10f(x-2h) - 10f(x-3h) + 5f(x-4h) - f(x-5h)}{h^5}}$$

b) In order to solve the first derivative function, we begin with the following the function that we will fill its coefficients in using linear algebra.

$$\frac{df}{dx} = \frac{(\alpha_1 f(x) + \alpha_2 f(x+h) + \alpha_3 f(x+2h) + \alpha_4 f(x+3h))}{h} + O(h^4)$$

To solve the function, we will plug in the multiple Taylor series function below.

$$f(x) = f(x)$$

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{6}f'''(x) + O(h^4)$$

$$f(x+2h) = f(x) + 2hf'(x) + 2h^2 f''(x) + \frac{4h^3}{3}f'''(x) + O(h^4)$$

$$f(x+3h) = f(x) + 3hf'(x) + \frac{9h^2}{2}f''(x) + \frac{27h^3}{6}f'''(x) + O(h^4)$$

Then we plug these functions into the original function.

$$\frac{(\alpha_1 f(x) + \alpha_2 f(x+h) + \alpha_3 f(x+2h) + \alpha_4 f(x+3h))}{h}$$
$$= \frac{\alpha_1 f(x)}{h} + \frac{\alpha_2 f(x) + \alpha_2 h f'(x) + \frac{\alpha_2 h^2}{2} f''(x) + \frac{\alpha_2 h^3}{6} f'''(x)}{h}$$
$$+ \frac{\alpha_3 f(x) + \alpha_3 2h f'(x) + \alpha_3 2h^2 f''(x) + \frac{\alpha_3 4}{3} h^3 f'''(x)}{h}$$
$$+ \frac{\alpha_4 f(x) + \alpha_4 3h f'(x) + \frac{\alpha_4 9}{2} h^2 f''(x) + \frac{\alpha_4 27}{6} h^3 f'''(x)}{h}$$

$$\frac{(\alpha_1 f(x) + \alpha_2 f(x+h) + \alpha_3 f(x+2h) + \alpha_4 f(x+3h))}{h}$$
$$= \left(\frac{\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4}{h}\right) f(x) + (\alpha_2 + 2\alpha_3 + 3\alpha_4) f'(x) + \left(\frac{\alpha_2}{2} + 2\alpha_3 + \frac{9}{2}\alpha_4\right) h f''(x)$$
$$+ \left(\alpha_2 + \frac{4}{3}\alpha_3 + \frac{27}{6}\alpha_4\right) h^2 f'''(x)$$

If we go ahead and break the big equation into the respective system of equations for each derivative, we can plug it into a matrix to solve the system of equations.

$$\frac{\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4}{h} = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 0$$

$$\alpha_2 + 2\alpha_3 + 3\alpha_4 = 1$$

$$\left(\frac{\alpha_2}{2} + 2\alpha_3 + \frac{9}{2}\alpha_4\right) = 0$$

$$\left(\alpha_2 + \frac{4}{3}\alpha_3 + \frac{27}{6}\alpha_4\right) = 0$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & \frac{1}{2} & 2 & \frac{9}{2} \\ 0 & \frac{1}{2} & \frac{4}{3} & \frac{27}{6} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

In Matlab, the coefficient matrix a is found:

```
A = [1 1 1 1; 0 1 2 3; 0 1/2 2 9/2; 0 1/6 4/3 27/6];
B = [0;1;0;0];
coeffs = inv(A)*B
```

```
coeffs = 4×1
  -1.833333333333333
   3.000000000000000
  -1.500000000000000
   0.333333333333333
```

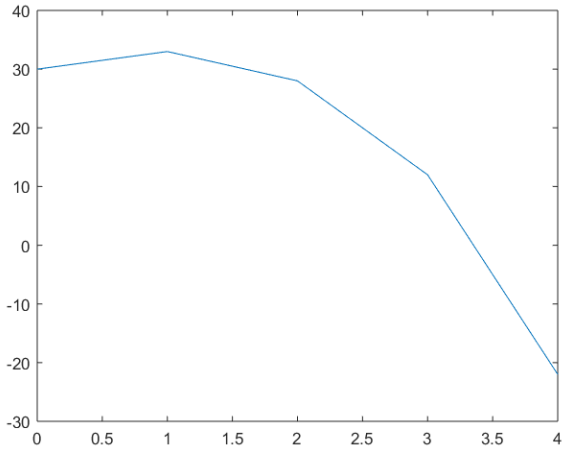
Therefore, the following equation is found:

$$\frac{df}{dx} = \frac{-\frac{11}{6}f(x) + 3f(x+h) - \frac{3}{2}f(x+2h) + \frac{1}{3}f(x+3h)}{h} + O(h^4)$$

Question 4

***In each of the following examples we only displayed the final equation, as the process to solve for the equation is the same process outlines in **question 3**. Please see above for the functions.

The given points correspond to the plot below:



a) Refer to Appendix C, part a. To calculate $f'(0)$, accurate to the second order h^2, we are unable to use the central difference method because we do not have the values before x=0. We will use the second order accurate forward difference, which uses the forward difference method with h and 2h points. This takes the Taylor series for the derivative and keeps values up until h^2 so we can be second order accurate.

$$f'(x) = \frac{-f(x+2) + 4f(x+h) - 3f(x)}{2h} + O(h^2)$$

The following is calculated.

$$\boldsymbol{f''(0) = 7}$$

b) Refer to Appendix C, part b. To calculate $f'(2)$, accurate to the second order h^2, we use the central difference method since we have the coordinates around it.

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + O(h^2)$$

The following is found:

$$\boldsymbol{f''(2) = -10.5}$$

c) Refer to Appendix C, part c. To calculate $f'(4)$, accurate to the second order h^2, we are unable to use the central difference method because we do not have the values before x=0. We will use the second order accurate backward difference, which uses the forward difference method with

h and 2h points. This takes the Taylor series for the derivative and keeps values up until h^2 so we can be second order accurate.

$$f'(x) = \frac{f(x-2h) - 4f(x-h) + 3f(x)}{2h} + O(h^2)$$

The following is calculated.

$$\boldsymbol{f''(0) = -43}$$

d) Refer to Appendix C, part d. To calculate $f''(0)$, accurate to the second order h^2, we are unable to use the central difference method because we do not have the values before x=0. We will use the second order accurate forward difference.

$$f''(x) = \frac{-f(x+3h) + 4f(x+2h) - 5f(x+h) + 2f(x)}{h^2} + O(h^2)$$

The following is calculated to.

$$\boldsymbol{f''(0) = -5}$$

Appendix

Appendix A – Question 1 Matlab Code

Question 1

```
syms f(t)
format long
% create the function variable and its integration endpoints
f(t) = log(5-4*cos(t));
% create a function that can be passed to functions
p = @(t) log(5-4*cos(t));
a = 0;
b = pi;
% set threshold to the error
thresh = 10^-6;
% actual answer
actual = int(f(t), t, a, b)
```

actual = $2\pi\log(2)$

```
actual1 = integral(p, a, b)
```

```
actual1 =
   4.355172180607203
```

part a, the midpoint rule

```
% the number of segments/points that will be used , assume >1
N_m = 1;
% set the sum of areas to 0
M = 0;
% check if the value vary by a large enough threshold
while(abs(M-actual)/actual > thresh)
    % reset the sum of areas to 0
    M = 0;
    % find the step size
    dx = (b-a)/N_m;
    for i=1:N_m
        % calculate the midpoint formula at each segment and sum the area
        % under the curve
        x = a + 0.5*(2*i-1)*dx;
        M = M + f(x)*dx;
    end
    % increment the number of segmenets
    N_m = N_m+1;
end
N_m = N_m - 1
```

```
N_m =
     9
```

```
relativeError = double(abs(M-actual)/actual)
```

```
relativeError =
     3.057463517027002e-07
```

```
double(M)
```

```
ans =
   4.355173512185210
```

```
% using my function
% [M,N_M] = MidpointIntegration(p,a,b,thresh)
```

part b, the trapezoid rule

```
% the number of segments/points that will be used , assume >1
N_t = 1;
% T starting point to 0
T = 0;
% check if the value vary by a large enough threshold
while(abs(T-actual)/actual > thresh)
    % find the step size
    dx = (b-a)/N_t;
    % we want x+1 and x for we need to x vectors
    x1 = a:dx:b-dx;
    x2 = a+dx:dx:b;

    % calculate the trapzoid method using vectors
    y = f(x2) +f(x1);
    T = 0.5*sum(y*dx);
%     T = 0;
%     T = 0.5*(f(a) + f(b));
%     for i=1:N_t-1
%         T = T + f(a + i*dx);
%     end
%     T = dx*T;

    % increment the number of segments
    N_t = N_t+1;
end
N_t = N_t - 1
```

```
N_t =
     9
```

```
relativeError = double(abs(T-actual)/actual)
```

```
relativeError =
     3.057475180346966e-07
```

```
double(T)
```

```
ans =
   4.355170849024120
```

part c, the simpsons rule

```
% the number of segments/points that will be used , assume >1
N_s = 1;
% set S = 0
S = 0;
% check if the value vary by a large enough threshold
while(abs(S-actual)/actual > thresh)
    % reset the sum of of area segments under the curve
```

```
    S = 0;
    % find the step size
    dx = (b-a)/N_s;
    for i=1:N_s
        % calculate the integral using the simpsons method
        x = a + (i-1)*dx;
        y = (dx/6)*(f(x) + 4*f(x+dx/2) + f(x + dx));
        S = S + y;
    end
    % increment the number of segments
    N_s = N_s + 1;
end
N_s = N_s - 1
```

```
N_s =
     8
```

```
relativeError = double(abs(S-actual)/actual)
```

```
relativeError =
     4.586099053484477e-07
```

```
double(S)
```

```
ans =
   4.355174177932306
```

Appendix B – Question 2 Matlab Code

Question 2

```
syms f(s,t) s h(s)
format long
% create the function variable and its integration endpoints
f(s,t) = s^2 + t;
a = 2;
b = 3;
c = s;
d = 2*s^3;
c_ = @(x) x;
d_ = @(x) 2*x.^3;
% create a function that can be passed to functions
p = @(x,y) x.^2 + y;
% set threshold to the error
thresh = 10^-4;
% actual answer
actual = integral2(p, a, b, c_, d_)
```

```
actual =
     7.905357142861429e+02
```

```
f(s,t)
```

ans = $s^2 + t$

part a, the midpoint rule

```
% the number of segments/points that will be used , assume >1148
N_m = 1149; % answer is 1149
% set the sum of areas to 0
M = 0;
% check if the value vary by a large enough threshold
while(abs(M-actual) > thresh)
    % reset the sum of areas to 0
    M = 0;
    temp = 0;
    % find the step size
    dy = (d-c)/N_m;
    for i=1:N_m
        % calculate the midpoint formula at each segment and sum the area
        % under the curve for the first part section under y
        y = c + 0.5*(2*i-1)*dy;
        temp = temp + f(s,y)*dy;
    end

    % find the step size
    dx = (b-a)/N_m;
    % keep the function in a symbolic function
    h(s) = temp;

    % repeat the algorithm for the area under the curve on the other x-axis
    for i=1:N_m
        x = a + 0.5*(2*i-1)*dx;
        M = M + h(x)*dx;
    end
    % increment the number of segmenets
    N_m = N_m+1;
end
M
```

```
M =
 1746275838931903522306 25
 2208978074108237376 96
```

```
N_m = N_m - 1
```

```
N_m =
        1149
```

```
absoluteError = double(abs(M-actual))
```

```
absoluteError =
     9.992207839443421e-05
```

part b, the trapezoidal rule

```
% the number of segments/points that will be used , assume >1624
N_t = 1625; % actual answer is 1625
% T starting point to 0
T = 0;
% check if the value vary by a large enough threshold
while(abs(T-actual) > thresh)
```

```
    % find the step size corresponding to the y-axis
    dy = (d-c)/N_t;
    % compute the area under the curve of the y-axis
    temp = 0.5*(f(s,c) + f(s,d));
    for i=1:N_t-1
        temp = temp + f(s,c + i*dy);
    end
    temp = dy*temp;

    % find the step size corresponding to the x-axis
    dx = (b-a)/N_t;
    h(s)= temp;

    % compute the area under the curve of the x-axis
    T = 0.5*(h(a) + h(b));
    for i=1:N_t-1
        T= T + h(a + i*dx);
    end
    T = dx*T;
    % increment the number of segments
    N_t = N_t+1;
end
N_t = N_t - 1
```

```
N_t =
        1625
```

```
T
```

```
T =
582239590880998940751 49
 73651260375976562500
```

```
absoluteError = double(abs(T-actual))
```

```
absoluteError =
     9.991278531424845e-05
```

part c, the simpsons rule

```
% the number of segments/points that will be used , assume >1
N_s = 1;
% set S = 0
S = 0;
% check if the value vary by a large enough threshold
while(abs(S-actual) > thresh)
    % reset the sum of of area segments under the curve
    S = 0;
    temp = 0;
    % find the step size on the y-axis
    dy = (d-c)/N_s;
    % calculate the integral using the simpsons method for hte y-axis
    for i=1:N_s
        y = c + (i-1)*dy;
        temp = temp + (dy/6)*(f(s,y) + 4*f(s,y+dy/2) + f(s,y+dy));
```

```
    end

    h(s) = temp;
    % find the step size on the x-axis
    dx = (b-a)/N_s;

    % calculate the integral using the simpsons method for x-axis
    for i=1:N_s
        x = a + (i-1)*dx;
        S = S + (dx/6)*(h(x) + 4*h(x+dx/2) + h(x+dx));
    end

    % increment the number of segments
    N_s = N_s + 1;
end
N_s = N_s - 1
```

```
N_s =
    12
```

```
S
```

```
S =
113305308109
143327232
```

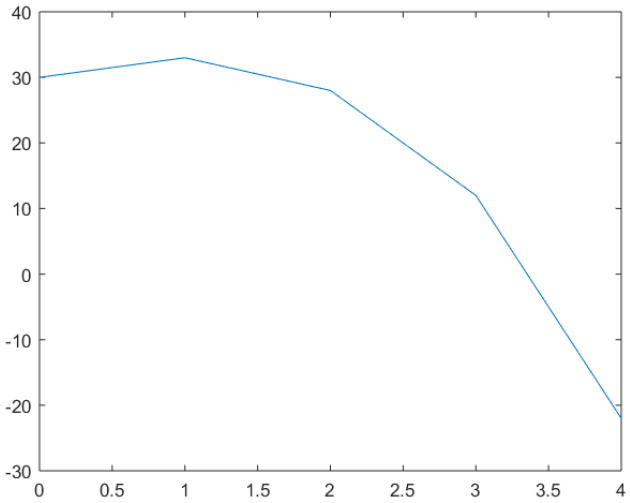
```
absoluteError = double(abs(S-actual))
```

```
absoluteError =
     8.639826573459449e-05
```

Appendix C – Question 4 Matlab Code

Question 4

```
format long
% import the data to be used
data = importdata('Ass_4_data_functQ4.txt');
x = data(:,1);
y = data(:,2);
plot(x,y)
```




```
% find the step size
h = x(2)-x(1)
```

```
h =
     1
```

Calculate f'(0)

```
% the point is zero, but to plug into the function, the array data starts
% at index 1
x_0 = 0+1;
% find the second order forward approximation
fd_0 = (-y(x_0 + 2*h) + 4*y(x_0 + h) - 3*y(x_0))/(2*h)
```

```
fd_0 =
     7
```

Calculate f'(2)

```
% the point is 2, but to plug into the function, the array data starts
% at index 1
x_1 = 2+1;
% find the second order centered approximation
cd_1 = (y(x_1 + h) - y(x_1 - h))/(2*h)
```

```
cd_1 =
 -10.500000000000000
```

Calculate f'(4)

```
% the point is 4, but to plug into the function, the array data starts
% at index 1
x_2 = 4+1;
% find the second order backward approximation
bd_2 = (y(x_2 - 2*h) - 4*y(x_2 - h) + 3*y(x_2))/(2*h)
```

```
bd_2 =
   -43
```

Calculate f''(0)

```
% the point is zero, but to plug into the function, the array data starts
% at index 1
x_3 = 0+1;
% find the second order forward approximation
fd_3 = (-y(x_3 + 3*h) + 4*y(x_3 + 2*h) - 5*y(x_3 + h) + 2*y(x_3))/h^2
```

```
fd_3 =
   -5
```