Bryan Jay
260738764

ECSE 443 - Assignment 1

***** USED ROUNDING UP FOR 5-9 NUMBERS AND ROUND DOWN FOR 0-4 NUMBERS.**

<u>Question 1</u> – a) MATLAB values, Refer to **appendix A**, section a

x =	10	1000	1000000
F(x)	1.6228	15.8153	500.0001

b) Calculator values: For these calculations I used my computer calculator which kept the most significant digits when compared to a standard calculator. This is due to the fact I wanted to keep as many digits as possible before rounding down.

$$f(x) = x * \left(\sqrt{x} - \sqrt{x-1}\right)$$

For x = 10:

$$f(10) = 10 * \left(\sqrt{10} - \sqrt{9}\right)$$

$$f(10) = 10 * (3.16228 - 3.00000)$$

$$f(10) = 10 * (0.16228)$$

For x = 1000:

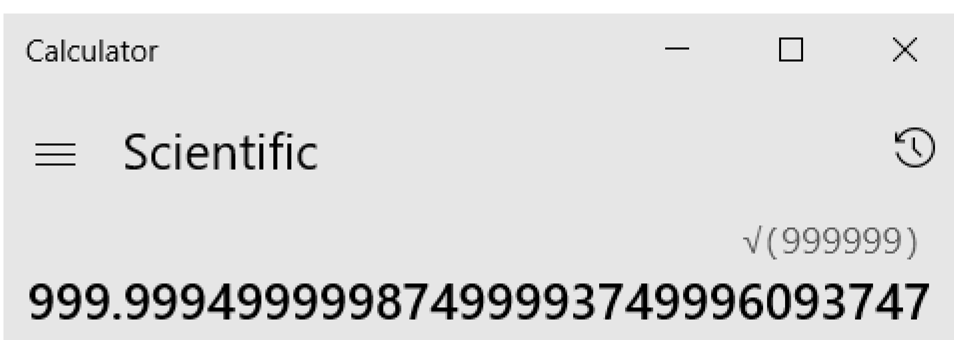
$$f(1000) = 1000 * \left(\sqrt{1000} - \sqrt{999}\right)$$

$$f(1000) = 1000 * (31.6228 - 31.6070)$$

$$f(1000) = 1000 * (0.0158000)$$

For x = 1000000:

I used my computer calculator and calculated 999.9994… which is a round down to 999.999 for 6 significant figures.



$$f(1000000) = 1000000 * \left(\sqrt{1000000} - \sqrt{999999}\right)$$

$$f(1000000) = 1000000 * (1000 - \ 999.999)$$

$$f(1000000) = 1000000 * (0.001)$$

x =	10	1000	1000000
F(x)	1.6228	15.8000	1000

c) Matlab Error Results with Calculator results. Refer to **appendix A**, section c.

abs_error = 1×3
0.0 -0.0153 499.9999

per_relative_err = 1×3
0.0014 -0.0970 99.9999
The error associated with the results calculated with the calculator is from the limited number of significant figures that we can use in order to do the calculations. Therefore, we lose precision throughout the operations.

d) Refer to **appendix A**, section d for calculation.

Starting Function:

$$f(x) = x\left(\sqrt{x} - \sqrt{x-1}\right)\frac{\sqrt{x} + \sqrt{x-1}}{\sqrt{x} + \sqrt{x-1}}$$

Simplified:

$$f(x) = \frac{x}{\sqrt{x} + \sqrt{x-1}}$$

For X = 10:

$$f(10) = \frac{10}{\sqrt{10} + \sqrt{10-1}}$$

$$f(10) = \frac{10}{3.16228 + 3.0}$$

$$f(10) = \frac{10}{6.16228}$$

For X = 1000:

$$f(1000) = \frac{1000}{\sqrt{1000} + \sqrt{1000-1}}$$

$$f(1000) = \frac{1000}{31.6228 + 31.6070}$$

$$f(1000) = \frac{1000}{63.2298}$$

For X = 1000000:

Bryan Jay
260738764

$$f(1000000) = \frac{1000000}{\sqrt{1000000} + \sqrt{1000000 - 1}}$$

$$f(1000000) = \frac{1000000}{1000 + 999.999}$$

$$f(1000000) = \frac{1000000}{1999.999}$$

x =	10	1000	1000000
F(x)	1.62278	15.8153	500.000

e) Matlab Error Results compared with modified function results in d). Refer to **appendix A**, section e.

abs_error = 1×3
10^{-3} ×
0.0034 -0.0431 -0.1250

per_relative_err = 1×3
10^{-3} ×
0.2094 -0.2727 -0.0250

The calculated error between the Matlab calculation and the modified function is lower for most inputs due to the elimination of subtraction in the function. When the subtraction operation occurs, it leads to a lost of significant figures when the two operands are close to one another. Therefore, the removal of this operation allows for a more precise result.

<u>Question 2</u> – a) MATLAB values, Refer to **appendix B,** section a

X=	0.007
F(x)	0.003500014291736696180563008388486

b) Again, I used my computers calculator to perform the calculations due to its extra significant figures therefore when I perform my rounding later it less likely to be affected by initial rounding made by my standard calculator.

$$f(x) = \frac{1 - \cos(x)}{\sin(x)}$$

$$f(0.007) = \frac{1 - \cos(0.007)}{\sin(0.007)}$$

$$f(0.007) = \frac{1 - 0.999976}{0.00699994}$$

$$f(0.007) = \frac{0.000024}{0.00699994}$$

X=	0.007
F(x)	0.00342860

c) Refer to **appendix B**, section c.

abs_error =
0.000071414291736696216917261329836799

per_relative_err =
2.040400003660004142025675973315

The error associated with the results is from the limiting number of significant figure which limits the precision of the result compared to when we use all the significant figures.

d) Refer to **appendix B**, section d for calculation.

Original Function:

$$f(x) = \frac{1-\cos(x)}{\sin(x)}\frac{1+\cos(x)}{1+\cos(x)}$$

$$f(x) = \frac{1-\cos^2(x)}{\sin(x)\,(1+\cos(x))}$$

$$f(x) = \frac{\sin^2(x)}{\sin(x)\,(1+\cos(x))}$$

Simplified Function:

$$f(x) = \frac{sin(x)}{1+\cos(x)}$$

For X = 0.007:

$$f(0.007) = \frac{sin(0.007)}{1+\cos(0.007)}$$

$$f(0.007) = \frac{0.00699994}{1+0.999976}$$

$$f(0.007) = \frac{0.00699994}{1.999976}$$

X=	0.007
F(x)	0.00350001

e) Matlab Error Results compared with modified function results in d). Refer to **appendix B**, section e.

```
abs_error =
  0.0000000042917366961670926724307090239394

per_relative_err = 0.00012262054775889347305826475631838
```

Similarly, to above in Q1, section e, the absolute and percent relative error are both much smaller due to the removal of the subtraction operation. The subtraction operation results in a loss of significant figures when the operations are similar.

<u>Question 3</u> – a)

Initial Function:

$$f(x) = e^{-4x}\cos(6x)$$

$$f'(x) = e^{-4x}(-4)\cos(6x) + e^{-4x}(-\sin(6x))(6)$$

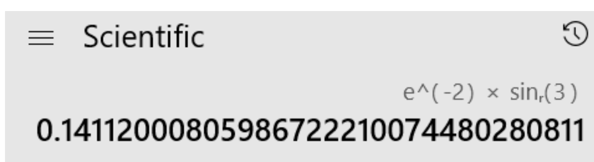
Derivative Function:

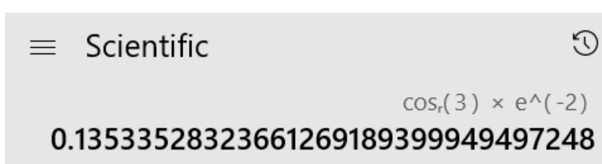
$$f'(x) = -4e^{-4x}\cos(6x) - 6e^{-4x}\sin(6x)$$

Solve for X = 0.5

$$f'(0.5) = -4e^{-4*0.5}\cos(6*0.5) - 6e^{-4*0.5}\sin(6*0.5)$$

$$f'(0.5) = -4e^{-2}\cos(3) - 6e^{-2}\sin(3)$$





$f'(0.5) = 0.5359236597181704538456210218446 2 - 0.11459109756681117859545744515281$

Final Answer:

$$f'(0.5) = 0.42133256215135927525016357669181$$

b) Refer to **appendix C**, section b.

Output from Matlab: `output = 0.43847880243653103711684144164038`

c)

Taylor Series of e^{-4x}:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots$$

$$e^{-4x} = 1 + (-4x) + \frac{(-4x)^2}{2!} + \frac{(-4x)^3}{3!} + \frac{(-4x)^4}{4!} + \cdots$$

$$e^{-4x} = 1 - 4x + \frac{16x^2}{2!} - \frac{64x^3}{3!} + \frac{256x^4}{4!} + \cdots$$

Taylor Series of $cos(6x)$:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} + \cdots$$

$$\cos(6x) = 1 - \frac{(6x)^2}{2!} + \frac{(6x)^4}{4!} - \frac{(6x)^6}{6!} + \frac{(6x)^8}{8!} + \cdots$$

Multiply the first 3 terms:

$$\mathrm{e}^{-4\mathrm{x}}\cos(6x) = (1 - \frac{36x^2}{2!} + \frac{1296x^4}{4!})(1 - 4x + \frac{16x^2}{2!})$$

$$\mathrm{e}^{-4\mathrm{x}}\cos(6x) = (1 - 18x^2 + 54x^4)(1 - 4x + 8x^2)$$

$$\mathrm{e}^{-4\mathrm{x}}\cos(6x) = 1 - 4x + 8x^2 - 18x^2 + 72x^3 - 144x^4 + 54x^4 - 216x^5 + 432x^6$$

$$f(x) = \mathrm{e}^{-4\mathrm{x}}\cos(6x) = 1 - 4x - 10x^2 + 72x^3 - 90x^4 - 216x^5 + 432x^6$$

$$f(x) = \mathrm{e}^{-4\mathrm{x}}\cos(6x) = 1 - 4x - 10x^2$$

Take the derivative:

$$x + h = 0.51$$

$$f(0.51) = 1 - 4(0.51) - 10(0.51)^2$$

$$f(0.51) =$$

$$f(0.5) = 1 - 4(0.5) - 10(0.5)^2$$

The work is also done in Matlab to compare answers in **appendix C**, section c. Final result is:

out_c = -14.1

d) Refer to **appendix C**, section d. The final output for the minimum step size is:

h = 1.000000000000000e-9

<u>Question 4</u> – a) Refer to **appendix D**, section a for the script and work done. The final answer is illustrated bellow.

```
x_a = 4×1
  -4.770833333333334
  -4.450757575757576
   3.757575757575756
   5.575757575757576
```

X_1	-4.770833333333334
X_2	-4.450757575757576
X_3	3.757575757575756
X_4	5.575757575757576

b) Refer to **appendix D**, section b and the function **gaussElim, in appendix E** for the implementation of the function.

```
x_b = 4×1
  -4.770833333333338
  -4.450757575757579
   3.757575757575760
   5.575757575757579
```

X_1	-4.770833333333338
X_2	-4.450757575757579
X_3	3.757575757575760
X_4	5.575757575757579

c) Refer to **appendix D**, section c. The absolute error for each entry was calculated as the following.

```
abs_error = 4×1
10^-14 ×
   0.444089209850063
   0.266453525910038
   0.444089209850063
   0.266453525910038
```

The maximum absolute error is:

```
max_abs =
     4.440892098500626e-15
```

<u>Question 5</u> – a) Refer to **appendix F**, section a to the steps in order to calculate the following Taylor Series.

f_t(x) = $\frac{x^2}{2} + x + 1$

b) Refer to **appendix F**, section b.

```
f_b = 1.010049999999999892352775532344821840524673461914062500000000000000
```

c)

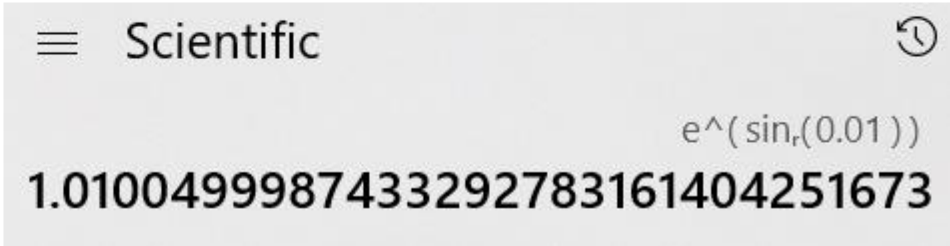
Starting Function:

$$f(x) = e^{sinx}$$

Solve x = 0.01:

$$f(0.01) = e^{\sin(0.01)}$$

Bryan Jay
260738764



$$f(0.01) = 1.01005$$

d) The results are relatively similar however, the result for c using the regular function will be more precise due to the fact that it is doing less approximation. In the case of the Taylor Series, we are approximating the value using the first 3 terms which leaves x^4 error in the answer.

Appendix

Appendix A – Question 1 Matlab Code

Q1 - a)

```
syms f(x)
input = [10, 1000, 1000000];
out_mat = zeros(size(input));
f(x) = x*(sqrt(x) - sqrt(x-1))
```

f(x) = $-x\left(\sqrt{x-1}-\sqrt{x}\right)$

```
for i = 1:length(input)
    x = input(i);
    out_mat(i) = vpa(f(x));
end
out_mat
```

```
out_mat = 1×3
10^2 ×
   0.016227766016838   0.158153431255768   5.000001250000625
```

c)

```
% values from calculator
out_calc = [1.6228, 15.8, 1000];
% absolute error
abs_error = out_calc - out_mat;
abs_error
```

```
abs_error = 1×3
10^2 ×
   0.000000233983162  -0.000153431255768   4.999998749999375
```

```
rel_error = zeros(size(input));
% relative error
for i=1:length(input)
    rel_error(i) = abs_error(i)/out_mat(i);
end
% Percent of Relative error
per_relative_err = rel_error*100
```

```
per_relative_err = 1×3
   0.001441869212467  -0.097014180817616  99.999949999987507
```

d) Calculation done in word with a calculator

e)

```
out_mat_new = [1.62278, 15.8153, 500];
```

```
% absolute error
abs_error = out_mat_new - out_mat;
abs_error
```

```
abs_error = 1×3
10^-3 ×
   0.003398316206660  -0.043125576773662  -0.125000062496383
```

```
rel_error = zeros(size(input));
% relative error
for i=1:length(input)
    rel_error(i) = abs_error(i)/out_mat(i);
end
% Percent of Relative error
per_relative_err = rel_error*100
```

```
per_relative_err = 1×3
10^-3 ×
   0.209413680424876  -0.272681891447036  -0.025000006249272
```

Appendix B - Question 2 Matlab Code

Question 2

a)

```
format long
syms f(x)
input = 0.007;
f(x) = (1-cos(x))/sin(x)
```

f(x) =

$$-\frac{\cos(x)-1}{\sin(x)}$$

```
x = input;
out_mat = vpa(f(input))
```

out_mat = $0.0035000142917366961805630083884 86$

b) Please see refer to written paper

F(x) = 0.00342860

c)

```
out_calc = 0.00342860;
% absolute error
abs_error = out_mat - out_calc
```

abs_error = $0.000071414291736696216917261329836799$

```
% relative error
rel_error = abs_error/out_mat;
per_relative_err = rel_error*100
```

per_relative_err = $2.0404000036600041420256759733 15$

d) Calculated using a calculator and demonstrated in word

e)

```
out_mat_new = 0.00350001;
% absolute error
abs_error = abs(out_mat_new - out_mat);
abs_error
```

abs_error = $0.0000000042917366961670926724307090239394$

```
% relative error
rel_error = abs_error/out_mat;
% Percent of Relative error
per_relative_err = rel_error*100
```

per_relative_err = $0.00012262054775889347305826475631838$

Appendix C - Question 3 Matlab Code

Question 3

a) refer to paper

b)

```
format long
syms p(x)
h = 0.01;
x_pt = 0.5;
p(x) = exp(-4*x)*cos(6*x)
```

p(x) = $\cos(6\,x)\,\mathrm{e}^{-4\,x}$

```
% p(x+h)
pin_xh = p(x_pt+h);
%p(x)
pin_x = p(x_pt);
p_out = vpa((pin_xh - pin_x)/h)
```

p_out = $0.43847880243653103711684144164038$

c)

```
syms f(x) f_t(x) k(x) k_t(x) g(x)
f(x) = exp(-4*x);
% use order 3 to get first 3 terms
f_t(x) = taylor(f, x, 'Order', 3)
```

f_t(x) = $8\,x^2 - 4\,x + 1$

```
% use order 5 to get the first 3 terms
k(x) = cos(6*x);
k_t(x) = taylor(k, x, 'Order', 5)
```

k_t(x) = $54\,x^4 - 18\,x^2 + 1$

```
g(x) = k_t(x)*f_t(x);
expand(g(x))
```

ans = $432\,x^6 - 216\,x^5 - 90\,x^4 + 72\,x^3 - 10\,x^2 - 4\,x + 1$

```
g(x) = 1-4*x-10*x^2;
% g(x+h)
g_xh = g(x_pt+h);
% g(x)
g_x = g(x_pt);
```

```
out_c = vpa((g_xh-g_x)/h)
out_c = −14.1
```

d) True value = 0.42133256215135927525016357669181 from a

Program to calculate the minimum step size DQA Dac DAQ GEDAAZx11

```
% true calue from part a
true_val = 0.42133256215135927525016357669181;
% starting step count
h = 0.01;
x = 0.5;
% find the initial value
min_abs_err = 1;
while(min_abs_err >= 0)
    p_out = (p(x+h)-p(x))/h;
    abs_error = vpa(abs(p_out-true_val));

    if abs_error >= min_abs_err
        % want to keep the minimum step size
        h_min = h*10;
        break;

    end
    % set new minimum
    min_abs_err = abs_error;
    h = h/10;
end
h_min
h_min =
     1.000000000000000e-09
```

Appendix D - Question 4 Matlab Code

Question 4

a)

```
format long
A = [4 -2 -3 6; 6 -7 6.5 -6; 1 7.5 6.25 5.5; -12 22 15.5 -1];
B = [12; -6.5; 16; 12];
x_a = inv(A)*B
x_a = 4×1
  -4.770833333333334
  -4.450757575757576
   3.757575757575756
   5.575757575757576
```

b)

```
X = [A B];
[n,m] = size(X);
```

```
X = gaussElim(X);
% get the last column
x_b = X(:,m)
```

```
x_b = 4×1
  -4.770833333333338
  -4.450757575757579
   3.757575757575760
   5.575757575757579
```

c)

```
% absolute error
abs_error = abs(x_b - x_a)
```

```
abs_error = 4×1
10^-14 ×
   0.444089209850063
   0.266453525910038
   0.444089209850063
   0.266453525910038
```

```
max_abs = max(abs_error(:))
```

```
max_abs =
     4.440892098500626e-15
```

Appendix E - Question 4 Gaussian Elimination function

```
% function to calculate Gaussian Elimination
function X = gaussElim(X)
% get the size of the array
[n, m] = size(X);


for i=1:n
    p = i;
    for k=i:n
        if(abs(X(p,i)) >= abs(X(k,i)))
            p = i;
        end
    end
    if(p ~= i)
        temp = X(p);
        X(p) = X(i);
        X(i) = temp;
    end
    % check if the diagonal is 0
    if X(i,i) == 0
        return
    end
    j = i;
    temp = X(i,j);
    X(i,:) = X(i,:)/temp;
    % loop through the matrix to get the triangular form
    for k=1:n
```

```
        if k ~= i
            X(k,:) = X(k,:)-X(i,:)*X(k,j);
        end
    end
end
```

Appendix F - Question 5 Matlab Code

Question 5

a)

```
format long
syms f(x) f_t(x)
f(x) = exp(sin(x));
f_t(x) = taylor(f, x, 'Order', 4)
```

f_t(x) =

$$\frac{x^2}{2} + x + 1$$

```
f_a = vpa(f_t(0))
```

f_a = 1.0

b)

```
f_b = vpa(f_t(0.01))
```

f_b = 1.01005

```
fprintf('%.64f', f_b);
```

1.0100499999999998923527755323448218405246734619140625000000000000

c) Calculation done in word and f_c =

d) Comparison completed in the word document.