Computational Engineering Assignment report

> *The whole details of this assignment is managed in my Github:*
>
> *Site: https://github.com/CHANShu0508/Engineering_Compute_Assignment*

1.0 The problem

The problems I have chosen is:

- For integrals:
  - The problem 2 of EXERCISE SET 4.3 on page 202 of the text book:

    Approximate the following integrals using the Trapezoidal rule.

    a. $\int_{-0.25}^{0.25} (\cos x)^2 \, dx$ **b.** $\int_{-0.5}^{0} x \ln(x+1) \, dx$

    c. $\int_{0.75}^{1.3} \left((\sin x)^2 - 2x \sin x + 1\right) \, dx$ **d.** $\int_{e}^{e+1} \frac{1}{x \ln x} \, dx$

  - The problem 22 of EXERCISE SET 4.4 on page 211 of the text book:

    Determine to within 10^{-6} the length of the graph of the ellipse with equation $4x^2 + 9y^2 = 36$.

    A car laps a race track in 84 seconds. The speed of the car at each 6-second interval is determined by using a radar gun and is given from the beginning of the lap, in feet/second, by the entries in the following table.

Time	0	6	12	18	24	30	36	42	48	54	60	66	72	78	84
Speed	124	134	148	156	147	133	121	109	99	85	78	89	104	116	123

    How long is the track?

- For differentials:
  - The problem 1 of EXERCISE SET 4.1 on page 182 of the text book:

    Use the forward-difference formulas and backward-difference formulas to determine each missing entry in the following tables.

    a.

x	$f(x)$	$f'(x)$
0.5	0.4794	
0.6	0.5646	
0.7	0.6442	

    b.

x	$f(x)$	$f'(x)$
0.0	0.00000	
0.2	0.74140	
0.4	1.3718	

2.0 The ideas

2.1 The integral problems

2.1.1 The first problem

There are for questions in this problem. For C language, I want to let users enter the upper limit, the lower limit of the integral. Additionally, because the basic frame of the algorithm is the same, we can make the progress of evaluating the value of the function into a function of C language, by which we can solve the four different questions by just change the content of the function.

And for MATLAB, I decided to solve them in two ways: first use `intgral()` function to solve the precise result, and then solve them by composed trapezoidal method, which means using `trapz()` function, for the conditions $n = 1\sim8$, get a result in respect to each condition.

2.1.2 The second problem

The second problem is some discrete data, so for C language, I decided to use the composed Simpuson's method to get result by these discrete numbers.

But for MATLAB, because there is no function expression is given, we have to use the trapezoidal method to approach the precise result.

2.2 The differential problem

For C language, I decided to use forward-difference and backward-difference formula to approach the value of derived function. And by enter different data, this program can output different result to solve different questions

And for MATLAB, I decided to use interpolation method to approximate the real function, and then evaluate it's derived function as the result.

3.0 Coding

3.1 The integral problems

3.1.1 The first problem

I want to give the code to question ***a*** as an example: *(the whole code is put in Appendix 1.0)*

```c
#include <stdio.h>
#include <math.h>

double functionValue(double x);

int main()
{
    int n, i;                          //Separate the domain into n parts
    double iniPoint_a, finPoint_b, h; //h = (b - a)/n
    double midValue, result;

    printf("Please enter the begin point: ");
    scanf("%lf", &iniPoint_a);
    printf("Please enter the end point: ");
    scanf("%lf", &finPoint_b);
    printf("\n");

    for (n = 1; n <= 8; n++) {
        h = (finPoint_b - iniPoint_a) / n;
        midValue = 0;
        for (i = 1; i <= n - 1; i++) {
            midValue += functionValue(iniPoint_a + i * h);
        }
        result = functionValue(iniPoint_a) + functionValue(finPoint_b);
        result *= 0.5;
        result += midValue;
        result *= h;
        printf("When n = %d, the integral is: %.6f\n", n, result);
    }

    return 0;
}

double functionValue(double x)
{
    double result;
```

```
    result = cos(x);
    result = pow(result, 2.0);

    return result;
}
```

For each question, we just change the content of the function "`functionValue()`" and we can use it to solve different questions.

And in the main function, we use loop body to achieve the features of the composed trapezoidal formula:

$$T_n = h\left(\frac{1}{2}[f(a)+f(b)]+\sum_{i=1}^{n-1} f(x_i)\right)$$

After calculating, this program will output the result from $n = 1$ to $n = 8$.

The output is:

> *When n = 1, the integral is: 0.469396*
> *When n = 2, the integral is: 0.484698*
> *When n = 3, the integral is: 0.487489*
> *When n = 4, the integral is: 0.488463*
> *When n = 5, the integral is: 0.488913*
> *When n = 6, the integral is: 0.489158*
> *When n = 7, the integral is: 0.489305*
> *When n = 8, the integral is: 0.489401*

And the MATLAB code is: *(list a. as example, the whole code is put in Appendix 1.0)*

```
clear;
aimFunc = @(x) cos(x).^2;
preciseResult = integral(aimFunc,-0.25,0.25);
result = zeros(1,8);
for n = 1:8
    h = (0.25+0.25)/n;
    x = linspace(-0.25,0.25,n+1);
    y = cos(x).^2;
    result(n) = trapz(x,y);
end
fprintf('The precise result is: %.4f\n',preciseResult);
for i = 1:8
   fprintf('When n = %d, the result by trapezoidal method is %.4f\n',i,result(i));
end
```

After computing, this program will output the precise value and the result by trapezoidal method from $n = 1$ to $n = 8$.

The output is:

> *The precise result is: 0.4897*
> *When n = 1, the result by trapezoidal method is 0.4694*
> *When n = 2, the result by trapezoidal method is 0.4847*
> *When n = 3, the result by trapezoidal method is 0.4875*
> *When n = 4, the result by trapezoidal method is 0.4885*
> *When n = 5, the result by trapezoidal method is 0.4889*
> *When n = 6, the result by trapezoidal method is 0.4892*

> *When n = 7, the result by trapezoidal method is 0.4893*
> *When n = 8, the result by trapezoidal method is 0.4894*

3.1.2 The second problem

The C code is as following:

```c
#include <stdio.h>

int main()
{
    int i, n = 7; //Because the given data, we got n = 14
    double h = 84.0 / n;
    double time[15] = {0}, sum1 = 0, sum2 = 0, result = 0;
    double speed[15] = {124, 134, 148, 156, 147, 133, 121,
    109, 99, 85, 78, 89, 104, 116, 123};
    for (i = 0; i < 15; i++) time[i] = 6 * i;

    for (i = 0; i < n; i++) sum1 += 4 * speed[i * 2 + 1];
    for (i = 0; i < n - 1; i++) sum2 += 2 * speed[(i + 1) * 2];

    result = sum1 + sum2 + speed[0] + speed[14];
    result *= (h / 6);
    printf("\nThe track is %.0f feets.\n", result);

    return 0;
}
```

The two loop in the code the two sums in the composed Simpuson's formula:

$$S_n = \sum_{i=0}^{n-1} \frac{h}{6}\left[f\left(x_i\right) + 4f\left(x_{i+\frac{1}{2}}\right) + f\left(x_{i+1}\right)\right] = \frac{h}{6}\left[f(a) + 4\sum_{i=0}^{n-1} f\left(x_{i+\frac{1}{2}}\right) + 2\sum_{i=1}^{n-1} f\left(x_i\right) + f(b)\right]$$

After calculating, this program will output the approximate result.

The output is:

> *The track is 9858 feet.*

And the MATLAB code are below:

```
clear;
time = 0:6:84;
speed = [124 134 148 156 147 133 121 109 99 85 78 89 104 116 123];
result = trapz(time,speed);
fprintf('The track is %.0f feet.\n',result);
```

By the trapezoidal method to approach the real result.

The output of this command is:

> *The track is 9855 feet.*

3.2 The differential problem

To solve this problem, the C code I have wrote is:

```c
#include <stdio.h>

int main()
{
    int i;
    double xArray[3] = {0}, yArray[3] = {0}, difference[3] = {0};
    printf("Please enter values of x: ");
    for (i = 0; i < 3; i++) scanf("%lf", &xArray[i]);
    for (i = 0; i < 3; i++) {
        printf("Please enter f(%.1f) = ", xArray[i]);
        scanf("%lf", &yArray[i]);
    } //Initialize the array
    double h = xArray[1] - xArray[0];

    for (i = 0; i < 2; i++) {
        difference[i] = (yArray[i + 1] - yArray[i]) / h;
    }
    difference[2] = (yArray[2] - yArray[1]) / h;

    printf("  x   |  f(x)   |  f'(x)  \n");
    printf("------+--------+---------\n");
    for (i = 0; i < 3; i++) {
        printf(" %.1f | %.4f | %.4f\n", xArray[i], yArray[i],difference[i]);
    }

    return 0;
}
```

This program uses forward-difference and backward-difference formula to approach the value of derived function. And user can input different data to solve these two questions.

And it's output is:

x	$f(x)$	$f'(x)$	and	x	$f(x)$	$f'(x)$
0.5	0.4794	0.8520		0.0	0.0000	3.7070
0.6	0.5646	0.7960		0.2	0.7414	3.1520
0.7	0.6442	0.7960		0.4	1.3718	3.1520

The MATLAB code of this problem is:

```m
clear;
x1 = 0.5:0.1:0.7;
x2 = 0:0.2:0.4;
y1 = [0.4794 0.5646 0.6442];
y2 = [0.0000 0.7414 1.3718];
p1 = polyfit(x1,y1,2);
p2 = polyfit(x2,y2,2);
syms a b;
f1(a) = p1(1).*a.^2+p1(2).*a+p1(3);
f2(b) = p2(1).*b.^2+p2(2).*b+p2(3);
difference1(a) = diff(f1(a),a);
difference2(b) = diff(f2(b),b);
result1 = difference1(x1);
result2 = difference2(x2);
for i = 1:3
```

```
    fprintf('In the first question: f''(%.1f)=%.6f\n',x1(i),result1(i));
end
for i = 1:3
    fprintf('In the second question: f''(%.1f)=%.6f\n',x2(i),result2(i));
end
```

In this program, we first use interpolation method to approximate the function by "`polyfit()`" and then we get the result by derivate the approximate function.

And it's output is:

> *In the first question: f'(0.5)=0.880000*
> *In the first question: f'(0.6)=0.824000*
> *In the first question: f'(0.7)=0.768000*
> *In the second question: f'(0.0)=3.984500*
> *In the second question: f'(0.2)=3.429500*
> *In the second question: f'(0.4)=2.874500*

4.0 Appendix: the whole code

The whole codes are managed in my Github:

> *https://github.com/CHANShu0508/Engineering_Compute_Assignment*

4.1 The integral problems

4.1.1 The first problem

- The question **2a**: (first C code, then MATLAB code)

```c
#include <stdio.h>
#include <math.h>

double functionValue(double x);

int main()
{
    int n, i;                          //Separate the domain into n parts
    double iniPoint_a, finPoint_b, h; //h = (b - a)/n
    double midValue, result;

    printf("Please enter the begin point: ");
    scanf("%lf", &iniPoint_a);
    printf("Please enter the end point: ");
    scanf("%lf", &finPoint_b);
    printf("\n");

    for (n = 1; n <= 8; n++) {
        h = (finPoint_b - iniPoint_a) / n;
        midValue = 0;
        for (i = 1; i <= n - 1; i++) {
            midValue += functionValue(iniPoint_a + i * h);
        }
```

```
        result = functionValue(iniPoint_a) + functionValue(finPoint_b);
        result *= 0.5;
        result += midValue;
        result *= h;
        printf("When n = %d, the integral is: %.6f\n", n, result);
    }

    return 0;
}

double functionValue(double x)
{
    double result;
    result = cos(x);
    result = pow(result, 2.0);

    return result;
}
```

```
clear;
aimFunc = @(x) cos(x).^2;
preciseResult = integral(aimFunc,-0.25,0.25);
result = zeros(1,8);
for n = 1:8
    h = (0.25+0.25)/n;
    x = linspace(-0.25,0.25,n+1);
    y = cos(x).^2;
    result(n) = trapz(x,y);
end
fprintf('The precise result is: %.4f\n',preciseResult);
for i = 1:8
   fprintf('When n = %d, the result by trapezoidal method is %.4f\n',i,result(i));
end
```

- The question **2b**: (first C code, then MATLAB code)

```
#include <stdio.h>
#include <math.h>

double functionValue(double x);

int main()
{
    int n, i;                          //Separate the domain into n parts
    double iniPoint_a, finPoint_b, h; //h = (b - a)/n
    double midValue, result;

    printf("Please enter the begin point: ");
    scanf("%lf", &iniPoint_a);
    printf("Please enter the end point: ");
    scanf("%lf", &finPoint_b);
    printf("\n");

    for (n = 1; n <= 8; n++) {
        h = (finPoint_b - iniPoint_a) / n;
```

```
        midValue = 0;
        for (i = 1; i <= n - 1; i++) {
            midValue += functionValue(iniPoint_a + i * h);
        }
        result = functionValue(iniPoint_a) + functionValue(finPoint_b);
        result *= 0.5;
        result += midValue;
        result *= h;
        printf("When n = %d, the integral is: %.6f\n", n, result);
    }

    return 0;
}

double functionValue(double x)
{
    double result;
    result = x * log(x + 1);

    return result;
}
```

```
clear;
aimFunc = @(x) x.*log(x+1);
preciseResult = integral(aimFunc,-0.5,0);
result = zeros(1,8);
for n = 1:8
    h = (0+0.5)/n;
    x = linspace(-0.5,0,n+1);
    y = x.*log(x+1);
    result(n) = trapz(x,y);
end
fprintf('The precise result is: %.4f\n',preciseResult);
for i = 1:8
   fprintf('When n = %d, the result by trapezoidal method is %.4f\n',i,result(i));
end
```

- The question **2c**: (first C code, then MATLAB code)

```
#include <stdio.h>
#include <math.h>

#define PI 3.1415926

double functionValue(double x);

int main()
{
    int n, i;                          //Separate the domain into n parts
    double iniPoint_a, finPoint_b, h; //h = (b - a)/n
    double midValue, result;

    printf("Please enter the begin point: ");
    scanf("%lf", &iniPoint_a);
    printf("Please enter the end point: ");
```

```
    scanf("%lf", &finPoint_b);
    printf("\n");

    for (n = 1; n <= 8; n++) {
        h = (finPoint_b - iniPoint_a) / n;
        midValue = 0;
        for (i = 1; i <= n - 1; i++) {
            midValue += functionValue(iniPoint_a + i * h);
        }
        result = functionValue(iniPoint_a) + functionValue(finPoint_b);
        result *= 0.5;
        result += midValue;
        result *= h;
        printf("When n = %d, the integral is: %.6f\n", n, result);
    }

    return 0;
}

double functionValue(double x)
{
    double result;
    result = pow(sin(x*PI/180), 2.0) - 2 * x * sin(x*PI/180) + 1;

    return result;
}
```

```
clear;
aimFunc = @(x) sin(x*pi/180).^2-2*x.*sin(x*pi/180)+1;
preciseResult = integral(aimFunc,0.75,1.3);
result = zeros(1,8);
for n = 1:8
    h = (0.3-0.75)/n;
    x = linspace(0.75,1.3,n+1);
    y = sin(x*pi/180).^2-2*x.*sin(x*pi/180)+1;
    result(n) = trapz(x,y);
end
fprintf('The precise result is: %.4f\n',preciseResult);
for i = 1:8
   fprintf('When n = %d, the result by trapezoidal method is %.4f\n',i,result(i));
end
```

- The question **2d**: (first C code, then MATLAB code)

```
#include <stdio.h>
#include <math.h>

double functionValue(double x);

int main()
{
    int n, i;                          //Separate the domain into n parts
    double iniPoint_a, finPoint_b, h; //h = (b - a)/n
    double midValue, result;
```

```c
    printf("Please enter the begin point: ");
    scanf("%lf", &iniPoint_a);
    printf("Please enter the end point: ");
    scanf("%lf", &finPoint_b);
    printf("\n");

    for (n = 1; n <= 8; n++) {
        h = (finPoint_b - iniPoint_a) / n;
        midValue = 0;
        for (i = 1; i <= n - 1; i++) {
            midValue += functionValue(iniPoint_a + i * h);
        }
        result = functionValue(iniPoint_a) + functionValue(finPoint_b);
        result *= 0.5;
        result += midValue;
        result *= h;
        printf("When n = %d, the integral is: %.6f\n", n, result);
    }

    return 0;
}

double functionValue(double x)
{
    double result;
    result = 1 / (x * log(x));

    return result;
}
```

```c
clear;
aimFunc = @(x) 1./(x.*log(x));
preciseResult = integral(aimFunc,exp(1),exp(1)+1);
result = zeros(1,8);
for n = 1:8
    h = 1/n;
    x = linspace(exp(1),exp(1)+1,n+1);
    y = 1./(x.*log(x));
    result(n) = trapz(x,y);
end
fprintf('The precise result is: %.4f\n',preciseResult);
for i = 1:8
   fprintf('When n = %d, the result by trapezoidal method is %.4f\n',i,result(i));
end
```

4.1.2 The second question

The code is: (first C code, then MATLAB code)

```c
#include <stdio.h>

int main()
{
    int i, n = 7; //Because the given data, we got n = 14
    double h = 84.0 / n;
```

```c
    double time[15] = {0}, sum1 = 0, sum2 = 0, result = 0;
    double speed[15] = {124, 134, 148, 156, 147, 133, 121,
    109, 99, 85, 78, 89, 104, 116, 123};
    for (i = 0; i < 15; i++) time[i] = 6 * i;

    for (i = 0; i < n; i++) sum1 += 4 * speed[i * 2 + 1];
    for (i = 0; i < n - 1; i++) sum2 += 2 * speed[(i + 1) * 2];

    result = sum1 + sum2 + speed[0] + speed[14];
    result *= (h / 6);
    printf("\nThe track is %.0f feets.\n", result);

    return 0;
}
```

```m
clear;
time = 0:6:84;
speed = [124 134 148 156 147 133 121 109 99 85 78 89 104 116 123];
result = trapz(time,speed);

fprintf('The track is %.0f feet.\n',result);
```

4.2 The differential problem

The code is: (first C code, then MATLAB code)

```c
#include <stdio.h>

int main()
{
    int i;
    double xArray[3] = {0}, yArray[3] = {0}, difference[3] = {0};
    printf("Please enter values of x: ");
    for (i = 0; i < 3; i++) scanf("%lf", &xArray[i]);
    for (i = 0; i < 3; i++) {
        printf("Please enter f(%.1f) = ", xArray[i]);
        scanf("%lf", &yArray[i]);
    } //Initialize the array
    double h = xArray[1] - xArray[0];

    for (i = 0; i < 2; i++) {
        difference[i] = (yArray[i + 1] - yArray[i]) / h;
    }
    difference[2] = (yArray[2] - yArray[1]) / h;

    printf("  x  |  f(x)  |  f'(x)  \n");
    printf("-----+--------+---------\n");
    for (i = 0; i < 3; i++) {
        printf(" %.1f | %.4f | %.4f\n", xArray[i], yArray[i],difference[i]);
    }

    return 0;
}
```

```m
clear;
```

```
x1 = 0.5:0.1:0.7;
x2 = 0:0.2:0.4;
y1 = [0.4794 0.5646 0.6442];
y2 = [0.0000 0.7414 1.3718];
p1 = polyfit(x1,y1,2);
p2 = polyfit(x2,y2,2);
syms a b;
f1(a) = p1(1).*a.^2+p1(2).*a+p1(3);
f2(b) = p2(1).*b.^2+p2(2).*b+p2(3);
difference1(a) = diff(f1(a),a);
difference2(b) = diff(f2(b),b);
result1 = difference1(x1);
result2 = difference2(x2);
for i = 1:3
    fprintf('In the first question: f''(%.1f)=%.6f\n',x1(i),result1(i));
end
for i = 1:3
    fprintf('In the second question: f''(%.1f)=%.6f\n',x2(i),result2(i));
end
```