

MA 322: Lab Assignment #6

Due on Sunday, September 13, 2015

Jiten Chandra Kalita

Tushar Sircar - 130123038

PROBLEM 1

```
#include<iostream>
#include<stdio.h>
#include<cmath>
using namespace std;

5
double getF(double x)
{
    return 1/(1 + (x*x));
}

10
double trapizoidalValue(double *x,int n,double h)
{
    double ans = 0;
    for(int i=1; i<n; i++)
15        ans += getF(x[i]);
    ans = ans*2;
    ans += getF(x[0]) + getF(x[n]);
    return (h*ans)/2;
}

20
double simpsonsOneThirdValue(double *x,int n,double h)
{
    double ans = 0;
    ans = getF(x[0]) + getF(x[n]);
25    for(int i=1; i<=n-1; i++)
    {
        if(i%2 == 0)
            ans += 2*getF(x[i]);
        else ans += 4*getF(x[i]);
30    }
    return (ans*h)/3;
}

double simpsonsThreeEighthValue(double *x,int n,double h)
35
{
    double ans = 0;
    ans = getF(x[0]) + getF(x[n]);
    for(int i=1; i<n; i++)
    {
40        if(i%3 == 0)
            ans += 2*getF(x[i]);
        else ans += 3*getF(x[i]);
    }
    return (3*h*ans)/8;
45 }

void printValuesAndErrors(double *x,int n,double h)
{
    cout<<endl;
50    double actual = 3.141/4;
    printf("Actual value of intergal: %f\nApproximation using %d intervals...\n",actual,n);
```

```
double trapizoidal = trapizoidalValue(x,n,h);
double sonethird = simpsonsOneThirdValue(x,n,h);
double sthreeeighth = simpsonsThreeEighthValue(x,n,h);
55 printf("Trapizoidal           : %f .... Error: %f\n",trapizoidal,abs(actual-trapizoidal));
printf("Simpsons One Third       : %f .... Error: %f\n",sonethird,abs(actual-sonethird));
printf("Simpsons Three Eighth Value: %f .... Error: %f\n",sthreeeighth,abs(actual-sthreeeighth));
}

60 int main()
{
    double left,right;
    cout<<"Enter end points: ";
    cin>>left>>right;

65     int n;
    cout<<"Enter number of intervals: ";
    cin>>n;

70     double *x = new double[(20*n)+1];
    double h = (right-left)/n;

    x[0] = left;
    for(int i=1; i<=n; i++)
75         x[i] = x[i-1] + h;
    printValuesAndErrors(x,n,h);

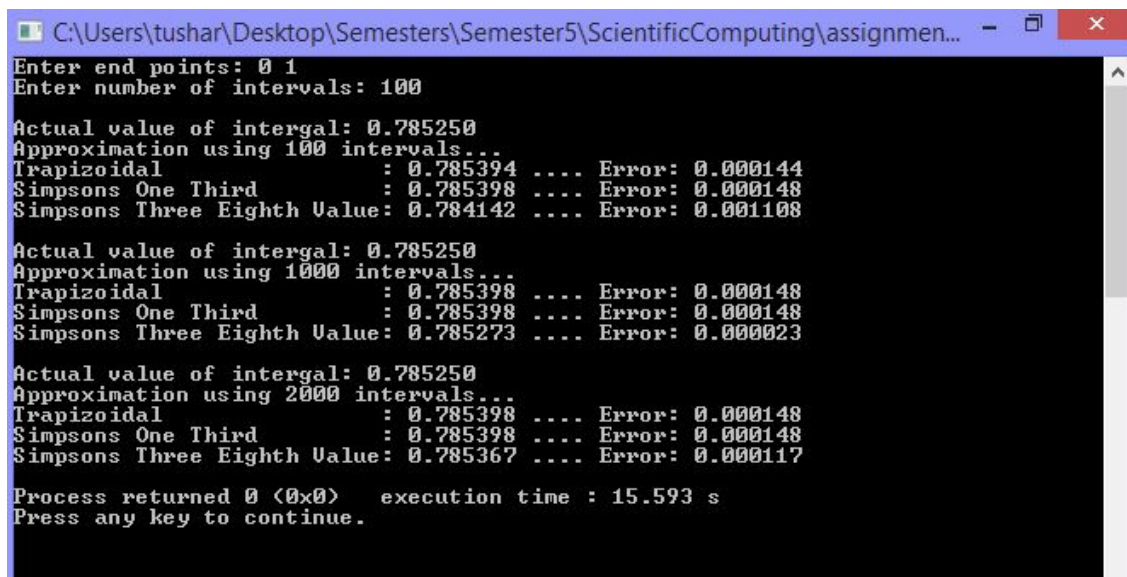
    x[0] = left;
80     h = h/(double)10;
    for(int i=1; i<=10*n; i++)
        x[i] = x[i-1] + h;
    printValuesAndErrors(x,10*n,h);

85     x[0] = left;
    h = h/(double)2;
    for(int i=1; i<=20*n; i++)
        x[i] = x[i-1] + h;
    printValuesAndErrors(x,20*n,h);

90

95     return 0;
}
```

OUTPUT



```
C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...
Enter end points: 0 1
Enter number of intervals: 100

Actual value of intergal: 0.785250
Approximation using 100 intervals...
Trapezoidal          : 0.785394 .... Error: 0.000144
Simpsons One Third   : 0.785398 .... Error: 0.000148
Simpsons Three Eighth Value: 0.784142 .... Error: 0.001108

Actual value of intergal: 0.785250
Approximation using 1000 intervals...
Trapezoidal          : 0.785398 .... Error: 0.000148
Simpsons One Third   : 0.785398 .... Error: 0.000148
Simpsons Three Eighth Value: 0.785273 .... Error: 0.000023

Actual value of intergal: 0.785250
Approximation using 2000 intervals...
Trapezoidal          : 0.785398 .... Error: 0.000148
Simpsons One Third   : 0.785398 .... Error: 0.000148
Simpsons Three Eighth Value: 0.785367 .... Error: 0.000117

Process returned 0 (0x0)   execution time : 15.593 s
Press any key to continue.
```

RESULTS

- (a) Number of intervals taken are 100,1000 and 2000.
- (b) Error decreases as the number of intervals are increased.
- (c) In terms of accuray Simpsons Three-Eighth > Simpsons One-Third > Trapezoidal, as expected based on theory.

PROBLEM 2(a)

```
#include<iostream>
#include<stdio.h>
#include<cmath>
5 using namespace std;

double getF(double x)
{
    return (double)4/(1 + (x*x));
10 }

double simpsonsOneThirdValue(double *x,int n,double h)
{
15     double ans = 0;
    ans = getF(x[0]) + getF(x[n]);
    for (int i=1; i<=n-1; i++)
    {
        if (i%2 == 0)
20         ans += 2*getF(x[i]);
        else ans += 4*getF(x[i]);
    }
    return (ans*h)/3;
25 }

double printValuesAndErrors(double *x,int n,double h)
{
30     double actual = 4*atan(1);
    double sonethird = simpsonsOneThirdValue(x,n,h);
    printf("%d\t\t\t%f\t%f\n",n,sonethird,abs(sonethird-actual));
    return abs(sonethird-actual);
35 }

double printIntervalsVsError(double left,double right,int n)
{
40     double h = (right-left)/n;
    double *x = new double[n+1];
    x[0] = left;
    for (int i=1; i<=n; i++)
        x[i] = x[i-1] + h;
    return printValuesAndErrors(x,n,h);
45 }

int main()
{
    double left,right;
50     cout<<"Enter end points: ";
    cin>>left>>right;
```

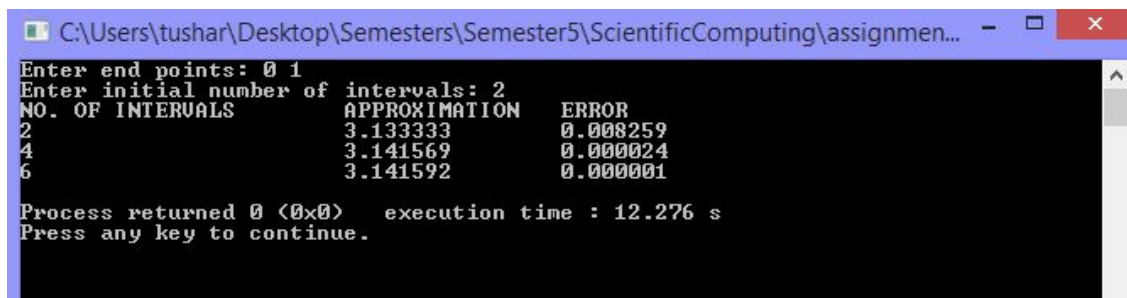
```
int n;
cout<<"Enter initial number of intervals: ";
cin>>n;

double *x = new double[(50*n)+1];
double h = (right-left)/n;

printf("NO. OF INTERVALS\tAPPROXIMATION\tERROR\n");
int i = n;
while(1)
{
    double error = printIntervalsVsError(left,right,i);
    if(error < 0.5*pow(10,-5))
        break;
    i+=2;
}

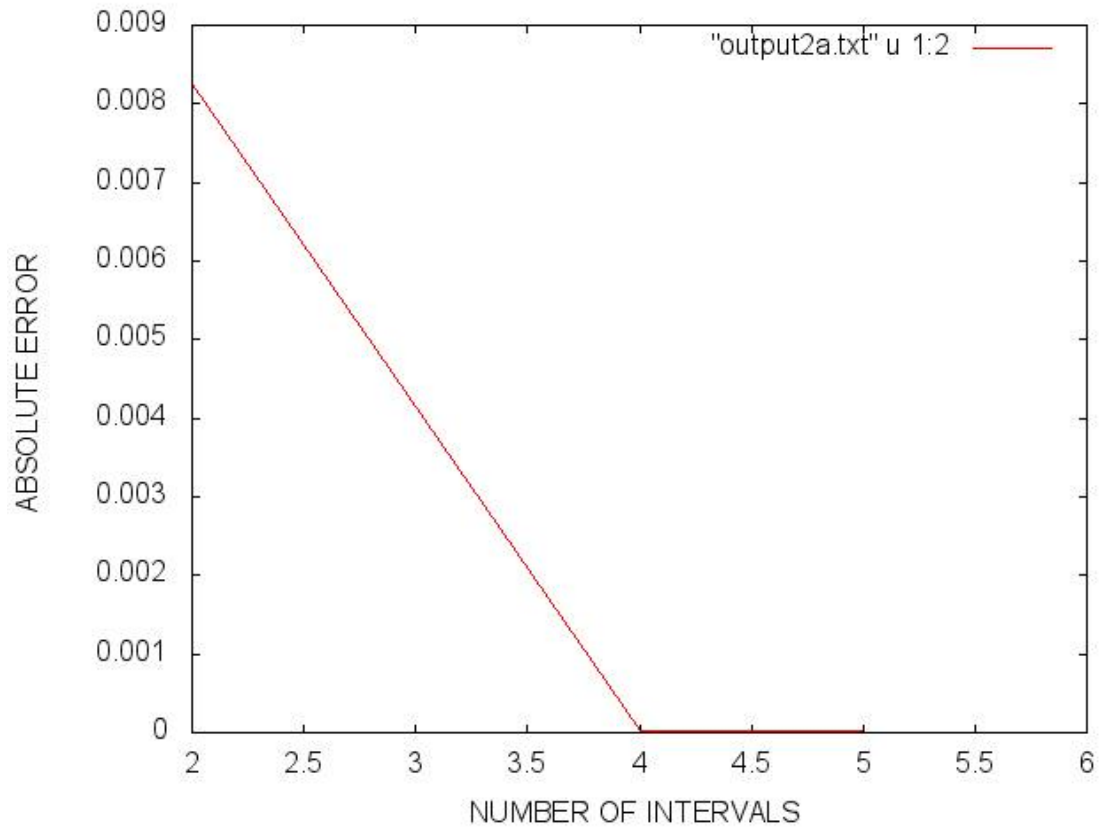
return 0;
}
```

OUTPUT



```
C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...
Enter end points: 0 1
Enter initial number of intervals: 2
NO. OF INTERVALS    APPROXIMATION    ERROR
2                   3.133333        0.008259
4                   3.141569        0.000024
6                   3.141592        0.000001

Process returned 0 (0x0)   execution time : 12.276 s
Press any key to continue.
```



RESULTS

- (a) Error decreases as we increase the number of intervals.
- (b) Number of intervals required is 6 for desired accuracy.

PROBLEM 2(b)

```
#include<iostream>
#include<stdio.h>
#include<cmath>
5 using namespace std;

double getF(double x)
{
    return sqrt(1 - (x*x)) - x;
10 }

double simpsonsOneThirdValue(double *x,int n,double h)
{
15     double ans = 0;
    ans = getF(x[0]) + getF(x[n]);
    for(int i=1; i<=n-1; i++)
    {
        if(i%2 == 0)
20         ans += 2*getF(x[i]);
        else ans += 4*getF(x[i]);
    }
    return (ans*h)/3;
25 }

double printValuesAndErrors(double *x,int n,double h)
{
30     double actual = 0.5*atan(1);
    double sonethird = simpsonsOneThirdValue(x,n,h);
    printf("%d\t\t\t%f\t%f\n",n,sonethird,abs(sonethird-actual));
    return abs(sonethird-actual);
35 }

double printIntervalsVsError(double left,double right,int n)
{
40     double h = (right-left)/n;
    double *x = new double[n+1];
    x[0] = left;
    for(int i=1; i<=n; i++)
        x[i] = x[i-1] + h;
    return printValuesAndErrors(x,n,h);
45 }

int main()
{
    double left,right;
50     cout<<"Enter end points: ";
    cin>>left>>right;
```



```
int n;
cout<<"Enter initial number of intervals: ";
cin>>n;

double *x = new double[(50*n)+1];
double h = (right-left)/n;

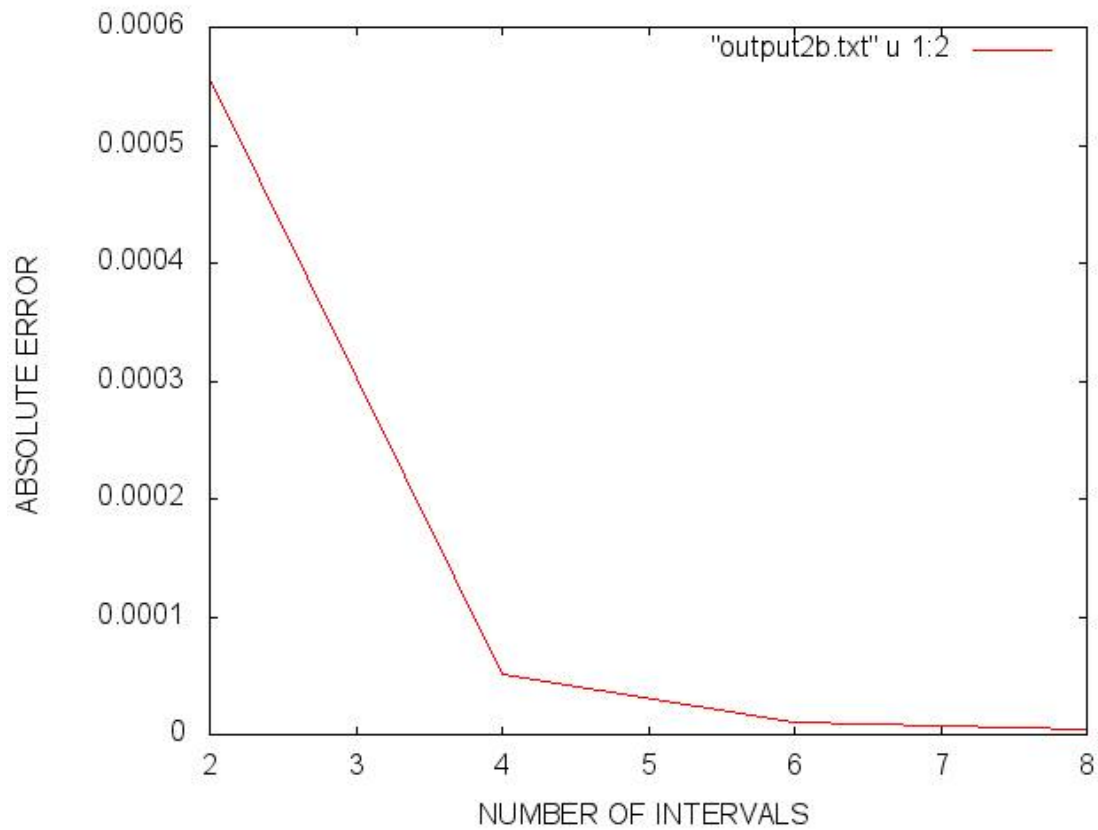
printf("NO. OF INTERVALS\tAPPROXIMATION\tERROR\n");
int i = n;
while(1)
{
    double error = printIntervalsVsError(left,right,i);
    if(error < 0.5*pow(10,-5))
        break;
    i+=2;
}

return 0;
}
```

OUTPUT

```
C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...
Enter end points: 0 0.707106
Enter initial number of intervals: 2
NO. OF INTERVALS      APPROXIMATION      ERROR
2                      0.392143          0.000556
4                      0.392648          0.000051
6                      0.392688          0.000011
8                      0.392695          0.000004

Process returned 0 (0x0)   execution time : 35.761 s
Press any key to continue.
```



RESULTS

- (a) Error decreases as we increase the number of intervals.
- (b) Number of intervals required is 8 for desired accuracy.

PROBLEM 3(b) - Part 1

```
#include<iostream>
#include<stdio.h>
#include<cmath>
using namespace std;

5
double getF(double x)
{
    return pow(x, (double)5);
}

10
double convertValue(double x, double a, double b)
{
    return ((b-a)*x + (b+a))/(double)2;
}

15
double threePointGaussian(double left, double right)
{
    double x1, x2, x3;
    double c1, c2, c3;

20
    c1 = 0.555555;
    c2 = 0.888888;
    c3 = 0.555555;

25
    x1 = convertValue(0.7745966692, left, right);
    x2 = convertValue(0, left, right);
    x3 = convertValue(-0.7745966692, left, right);

    return (c1*getF(x1) + c2*getF(x2) + c3*getF(x3));

30
}

void evaluateByDividing(double left, double right, int n)
35
{
    double h = (right-left)/n;
    double ans = 0;
    for(int i=0; i<n; i++)
        ans += (h/(double)2)*threePointGaussian(left + (i*h), left + ((i+1)*h));
40
    printf("Approximation: %f\n", ans);
}

int main()
{
45
    double left, right;
    cout<<"Enter end points: ";
    cin>>left>>right;

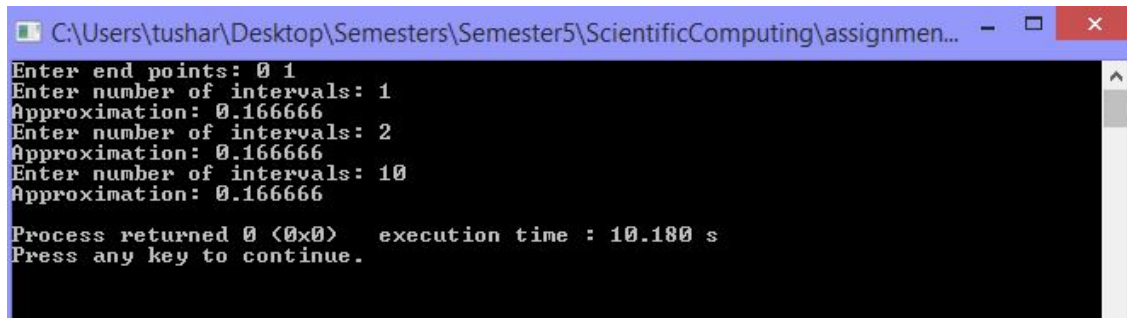
    int n;

50
    int i =1;
```

```
55     while ((i++) < 4)
    {
        cout << "Enter number of intervals: ";
        cin >> n;
        evaluateByDividing(left, right, n);
    }

60     return 0;
}
```

OUTPUT



```
C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...
Enter end points: 0 1
Enter number of intervals: 1
Approximation: 0.166666
Enter number of intervals: 2
Approximation: 0.166666
Enter number of intervals: 10
Approximation: 0.166666

Process returned 0 (0x0)   execution time : 10.180 s
Press any key to continue.
```

PROBLEM 3(b) - Part 2

```
#include<iostream>
#include<stdio.h>
#include<cmath>
using namespace std;

5
double getF(double x)
{
    return sin(x)/x;
}

10
double convertValue(double x,double a,double b)
{
    return ((b-a)*x + (b+a))/2;
}

15
double threePointGaussian(double left,double right)
{
    double x1,x2,x3;
    double c1,c2,c3;

20
    c1 = 0.555555;
    c2 = 0.888888;
    c3 = 0.555555;

25
    x1 = convertValue(0.7745966692,left,right);
    x2 = convertValue(0,left,right);
    x3 = convertValue(-0.7745966692,left,right);

    return (c1*getF(x1) + c2*getF(x2) + c3*getF(x3));

30
}

void evaluateByDividing(double left,double right,int n)
35
{
    double h = (right-left)/n;
    double ans = 0;
    for(int i=0; i<n; i++)
        ans += (h/(double)2)*threePointGaussian(left + (i*h),left + ((i+1)*h));
40
    printf("Approximation: %f\n",ans);
}

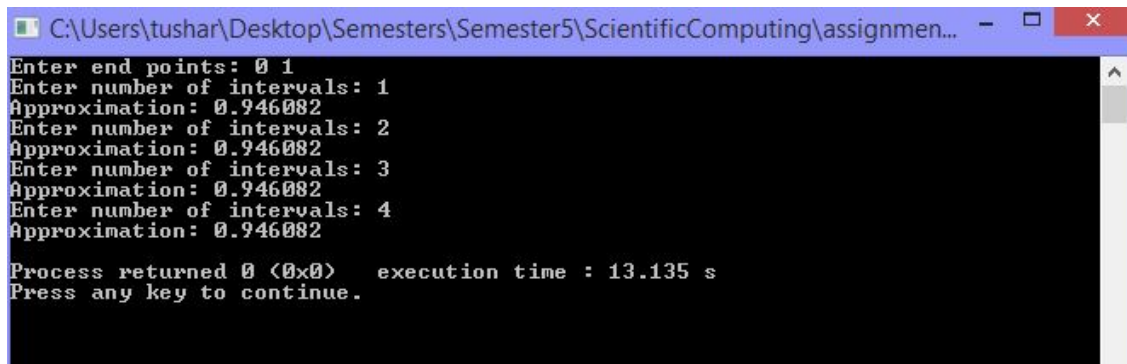
int main()
{
45
    double left,right;
    cout<<"Enter end points: ";
    cin>>left>>right;

    int n;

50
```

```
int i = 1;
while ((i++) < 5)
{
    cout << "Enter number of intervals: ";
55     cin >> n;
    evaluateByDividing(left, right, n);
}

60     return 0;
}
```



```
C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...
Enter end points: 0 1
Enter number of intervals: 1
Approximation: 0.946082
Enter number of intervals: 2
Approximation: 0.946082
Enter number of intervals: 3
Approximation: 0.946082
Enter number of intervals: 4
Approximation: 0.946082
Process returned 0 (0x0)   execution time : 13.135 s
Press any key to continue.
```