Homework 1

Rakesh Nath

February 12, 2013

Problem 1

We need to integrate the function

$$f(x) = \int_0^{1000} x^3 \sin(x) \exp(-x)$$

We use the simpson's formula which is generalized by

$$\int_a^b f(x)dx = \frac{3h}{8}[f(x_0 + 3f(x_1) + 3f(x_2) + 2f(x_3) + 3f(x_4).... + f(x_n)]$$

where

$$h = \frac{b-a}{n}$$

where n is the number of bins

$$x_i = a + i \times h$$

The convergence of this function determines if the integral has been solved or not. Convergence is measured when the difference between the values produced by the current bin size and the previous bin size differ by a relative value of 10^{-5}. If I is the interval size then the error E is

$$E = \frac{I_n - I_{n-1}}{I_n}$$

The integral does not converge and the answer is . The Fig(1) shows the plot of convergence of the error over the value of the number of bins

The error values have a lot of bins so it does not make sense to include them as a table. So it has not been included.The code is included in the appendices

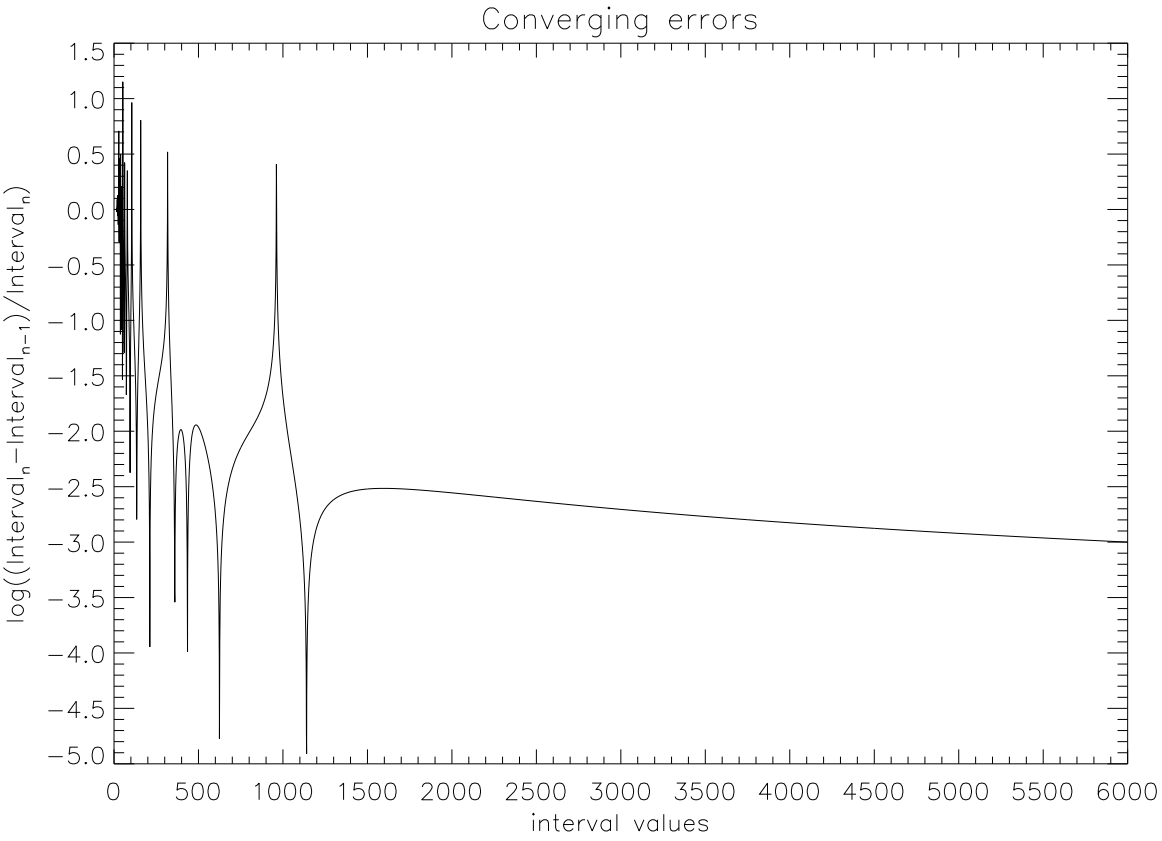

Figure 1: Error convergences over bin size

Appendices

```
/*****************************************************************
****Program to compute the integral of x^3sin(x)e^(-x) between the
limits 0 to 1000. This program uses the simpson's 3/8th rule to compute
the integral and does it to an interval of 6000. Each interval has 2 bins
so the total number of bins can go upto 12000.
Functions: simpsonthree declared implicitly and used as a separate
piece of code
Author: Rakesh Nath
*****************************************************************/
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#include<float.h>
#include<string.h>
#include<file.h>
#include<omp.h>
//#include<thread.h>
/*Precompiler definitions*/
#define N 6000/*Number of interval N=2xnumber of bins*/
#define ERR 1.0e-5//error tolerance
/*formal declaration of the function */
double simpsonthree(double (*func)(double),double,double,int);
/*************************************************************
Function: f
Description: Function merely returns double value of the math function
declared within it.
Returns: double
Input: Double values which will be supplied by the main program
*************************************************************/

double f(double x)
        {
                return pow(x,3)*sin(x)*exp(-x);
                //return sin(x);
                //return exp(pow(x,2));
        }
/*************************************************************
Function Name:main()
Description: The main program does the integration using Simpson's 3/8th rule
by calling the simpsonthree() function. The error values are calculated using
relative error and are saved to a file named error_values.dat.
Returns: If successful 0
Input:none
*************************************************************/
int main()
{       FILE *fp;
        /*file pointer fp*/
        double *dSim= malloc((N+1)*sizeof(double));
        /*size is allocated based on the number of intervals,
this will have the array of values of the result of the
```

```
simpson's integration.*/
        double iLowerLimit,iUpperLimit;
        //upper and lower limits
        int i;
        //loop variable
        double dRes,dErr[N],dFinres;
        //results, errors and intermediate results

        iLowerLimit=0;
        iUpperLimit=1000;
/*The lower limit is set to 0 and the upper limit to a 1000*/

        fp=fopen("error_values.dat","w+");
        //open the file error_values.dat
        for(i=1;i<N;i+=1)
{
                dSim[i]=simpsonthree(f,iLowerLimit,iUpperLimit,i);
//the first set of values computed with an interval i
                dFinres=simpsonthree(f,iLowerLimit,iUpperLimit,i-1);
//the preveious value with interval i-1
                dErr[i]=fabs((dSim[i]-dFinres)/dSim[i]);
//error calculation
                fprintf(fp,"%e\n",dErr[i]);
/*if the error is less than the 10^-5 then print,
this is the convergence of the integral*/
                if(dErr[i]<=ERR)
                {
                printf("at i=%d\t",i);
                printf("The value using Simpson's 3/8th rule is %e\n",dSim[i]);
                printf("the value of error is %e\n",dErr[i]);
                //break;
                }
        }

fclose(fp);
free(dSim);
        return 0;
}

/*************************************************************************
Function Name: Simpsonthree
Description: The simpsonthree works using the 3/8th rule of simpson that
replicates the number of bins using the formula for simpsons rule.
output: Double value corresponding to the sum of the individual integral
results from the bins
Input: Function to be integrated, upper limit, lower limit, number of bins
*************************************************************************/


#include<stdio.h>
```

```
#include<math.h>

double simpsonthree(double (*func)(double),double a,double b,int n)
        {
                double dH,dRes;
                int N=n;
        //This is the number of intervals

                double *x;
                x=malloc((N+1)*sizeof(double));
//malloc to allocate the memory for the size of the output to be summed
                int i;
//loop variable
                x[0]=a;
//intial value at which function is evaluated
                x[N]=b;
//Final value at which function is evaluated
                dH=(b-a)/(double)N;
//bin size
                dRes=3*dH/8*func(a);
//variable initialized to integral value
        omp_set_num_threads(100);
/*this sets the number of threads that will be used for
the program*/
        for(i=1;i<N;i++)
        {
                        x[i]=a+i*dH;
        }

        #pragma omp parallel reduction(+:dRes) //pragma!!

        #pragma omp for
        //for loop is being parallized
                for(i=1;i<N;i+=3)
                {
                        //split the bins
                    dRes+=3*dH/8*(3*func(x[i])+3*func(x[i+1])+2*func(x[i+2]));
                        /*compute integral at all the values summing up
all the bins*/
                }
                return dRes;
        }
```