MA202 LAB6

Name: Dipean Dasgupta ID:202151188

Task 1:

1. Compute the integral of e^{-x^2} from -10 to 10, using Trapezoidal rule, Simpson's 1/3 rule, and Simpson's 3/8 rule. Plot a graph of absolute value of the change in the integral as a function of no. of sampling points. Also plot a graph of absolute value of the relative change in the integral as a function of no. of sampling points. Comment on the results obtained.

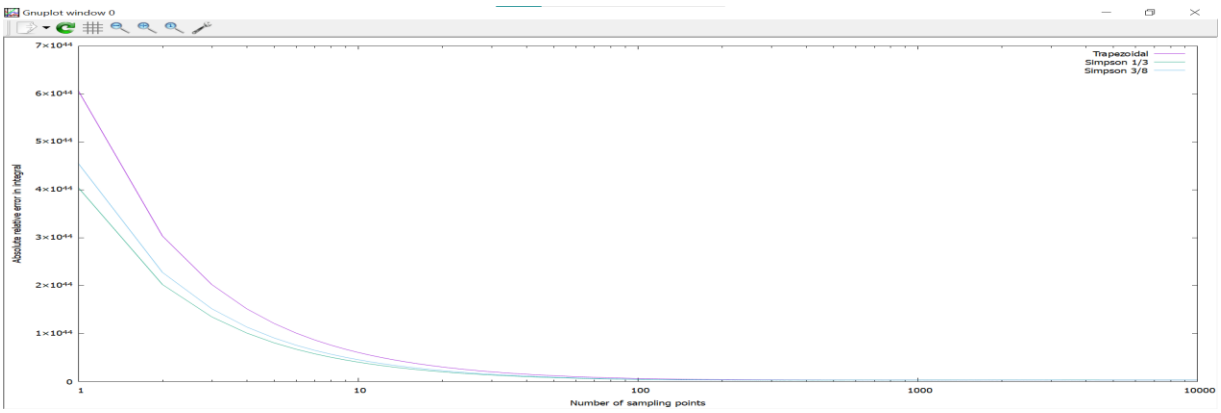
Solution Code:

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

double f(double x) {
    return exp(-(x * x));
}

double trapezoidal_rule(double a, double b, int n) {
    double h = (b - a) / n;
    double sum = 0.5 * (f(a) + f(b));
    for (int i = 1; i < n; i++) {
        double x = a + i * h;
        sum += f(x);
    }
    return h * sum;
}

double simpson_13_rule(double a, double b, int n) {
    double h = (b - a) / n;
    double sum = f(a) + f(b);
    for (int i = 1; i < n; i += 2) {
        double x = a + i * h;
        sum += 4.0 * f(x);
    }
    for (int i = 2; i < n; i += 2) {
        double x = a + i * h;
        sum += 2.0 * f(x);
    }
    return h * sum / 3.0;
}
```

```
double simpson_38_rule(double a, double b, int n) {
    double h = (b - a) / n;
    double sum = f(a) + f(b);
    for (int i = 1; i < n; i += 3) {
        double x = a + i * h;
        sum += 3.0 * f(x);
    }
    for (int i = 2; i < n; i += 3) {
        double x = a + i * h;
        sum += 3.0 * f(x);
    }
    for (int i = 3; i < n; i += 3) {
        double x = a + i * h;
        sum += 2.0 * f(x);
    }
    return 3.0 * h * sum / 8.0;
}


int main() {
    double a = -10.0;
    double b = 10.0;
    double true_value = sqrt(M_PI) * 0.5 * erf(10.0);
    int max_n = 10000;
    double trapezoidal[max_n];
    double simpson_13[max_n];
    double simpson_38[max_n];
    double abs_error_trapezoidal[max_n];
    double abs_error_simpson_13[max_n];
    double abs_error_simpson_38[max_n];
    double rel_error_trapezoidal[max_n];
    double rel_error_simpson_13[max_n];
    double rel_error_simpson_38[max_n];
    for (int n = 1; n <= max_n; n++) {
        trapezoidal[n - 1] = trapezoidal_rule(a, b, n);
        simpson_13[n - 1] = simpson_13_rule(a, b, n);
        simpson_38[n - 1] = simpson_38_rule(a, b, n);
        abs_error_trapezoidal[n - 1] = fabs(trapezoidal[n - 1] - true_value);
        abs_error_simpson_13[n - 1] = fabs(simpson_13[n - 1] - true_value);
    abs_error_simpson_38[n - 1] = fabs(simpson_38[n - 1] - true_value);
    rel_error_trapezoidal[n - 1] = fabs((trapezoidal[n - 1] - true_value) /
true_value);
    rel_error_simpson_13[n - 1] = fabs((simpson_13[n - 1] - true_value) /
true_value);
```

```
    rel_error_simpson_38[n - 1] = fabs((simpson_38[n - 1] - true_value) /
true_value);
}
FILE *abs_error_file = fopen("abs_error.txt", "w");
FILE *rel_error_file = fopen("rel_error.txt", "w");
for (int n = 1; n <= max_n; n++) {
    fprintf(abs_error_file, "%d %e %e %e\n", n, abs_error_trapezoidal[n - 1],
abs_error_simpson_13[n - 1], abs_error_simpson_38[n - 1]);
    fprintf(rel_error_file, "%d %e %e %e\n", n, rel_error_trapezoidal[n - 1],
rel_error_simpson_13[n - 1], rel_error_simpson_38[n - 1]);
}




fclose(abs_error_file);
fclose(rel_error_file);
FILE *gnuplot_pipe = popen("gnuplot -persist", "w");
fprintf(gnuplot_pipe, "set logscale x\n");
fprintf(gnuplot_pipe, "set xlabel 'Number of sampling points'\n");
fprintf(gnuplot_pipe, "set ylabel 'Absolute error in integral'\n");
fprintf(gnuplot_pipe, "plot 'abs_error.txt' using 1:2 with lines title
'Trapezoidal', 'abs_error.txt' using 1:3 with lines title 'Simpson 1/3',
'abs_error.txt' using 1:4 with lines title 'Simpson 3/8'\n");
fprintf(gnuplot_pipe, "set ylabel 'Absolute relative error in integral'\n");
fprintf(gnuplot_pipe, "plot 'rel_error.txt' using 1:2 with lines title
'Trapezoidal', 'rel_error.txt' using 1:3 with lines title 'Simpson 1/3',
'rel_error.txt' using 1:4 with lines title 'Simpson 3/8'\n");
fflush(gnuplot_pipe);
getchar();
pclose(gnuplot_pipe);
return 0;
}
```

OUTPUT:



Task 2:

2. Compute the integral of $\frac{\sin x^2}{x}$ from -10 to 10, using Trapezoidal rule, Simpson's 1/3 rule, and Simpson's 3/8 rule. Plot a graph of absolute value of the change in the integral as a function of no. of sampling points. Also plot a graph of absolute value of the relative change in the integral as a function of no. of sampling points. Comment on the results obtained.

```
# include<stdio.h>

# include<math.h>

float e = 2.18;

float sinpowx(float  x){

    return (sin(x*x))/x;

}

int main(){

    // Trapezium

      double curr = 0.0;

      double prev = 0.0;

FILE * ptr  = fopen("lab6_q_2.dat","w");

    float y0;

    float yn;

    float sum = 0.0f;

    float intv = 0.0f;

    for(int n =1;n<100;n++){

        intv = 1.0/n;

    for(float i = -10.0f;i<=10.0f;i += intv){

        if(i == 0)continue;

        float a = sinpowx(i);

        if(i == -10.0f){

            y0 = a;
```

```
        }
        else if(i == 10.0f){
            yn = a;
        }else{
            sum += a;
        }
    }
    sum += (y0+yn)/2.0f;
    sum *= intv; // interval taken is 0.2
    printf("sum = %f\n",sum);
      curr = sum;
    double req = sum-prev;
    if(prev!=0.0);  fprintf(ptr, "%d %1.30lf\n",n,req);
    prev = curr;

//simpson 1/3;
    long counter = 0;
    float sum_even = 0.0f;
    float sum_odd = 0.0f;
    float sum_simpson = 0.0f;
    for(float i = -10.0f;i<=10;i += intv){
                if(i == 0)continue;
        float a = sinpowx(i);
         if(i == -10.0f){
            y0 = a;
        }
        else if(i == 10.0f){
```

```
            yn = a;
        }else{
            if(counter % 2 == 0)    {
                sum_even += a;
            }else{
                sum_odd += a;
            }
        }
        counter++;
    }
    sum_even *= 2.0f;
    sum_odd *= 4.0f;
    sum_simpson = sum_even+sum_odd + y0+ yn;
    sum_simpson *= (intv)/3.0f;
    printf("sum_simpson = %f\n" , sum_simpson);


    // Simpson 3/8
    long counter1 = 0;
    float sum_even1 = 0.0f;
    float sum_odd1 = 0.0f;
    float sum_simpson1 = 0.0f;
    for(float i = -10.0f;i<=10;i += intv){
                if(i == 0)continue;


        float a = sinpowx(i);
         if(i == -10.0f){
            y0 = a;
```

```
        }else if(i == 10.0f){

            yn = a;

        }else{

            if(counter1 % 3 == 0)    {

                sum_even1 += a;

            }else{

                sum_odd1 += a;

            }}

        counter1++;

    }

    sum_even1 *= 2.0f;

    sum_odd1 *= 3.0f;

    sum_simpson1 = sum_even1+sum_odd1 + y0+ yn;

    sum_simpson1 *= (intv*3.0)/8.0f;

    printf("sum_simpson_3_over_8 = %f" , sum_simpson1);

    }

    return 0;

}
```

OUTPUT: