Московский государственный университет
имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики

Отчет по заданию №6

«Сборка многомодульных программ. Вычисление корней уравнений и определенных интегралов.»

Вариант 10 / 3 / 2

Выполнил:
студент 106 группы
Оганисян Э. Г.

Преподаватель:
Корухова Л. С.

Москва
2019

Содержание

Постановка задачи **2**

Математическое обоснование **3**

Результаты экспериментов **5**

Структура программы и спецификация функций **6**

Сборка программы (Make-файл) **7**

Отладка программы, тестирование функций **8**

Программа на Си и на Ассемблере **9**

Анализ допущенных ошибок **10**

Список цитируемой литературы **11**

Постановка задачи

В задании требовалось с заданной точностью ε вычислить площадь плоской фигуры, ограниченной тремя кривыми: $y = \frac{4}{x^2+1} + 1$, $y = x^3$ и $y = 2^{-x}$. Для этого необходимо было:

- с некоторой точностью ε_1, определенной аналитически, вычислить абсциссы точек пересечения кривых, используя метод Ньютона или метод секущих приближенного решения уравнения $\mathrm{F(x)} = 0$,
- аналитически определить отрезки, на которых программа ищет точки пересечения, с учетом области применимости используемых методов и некоторой грубой оценки значений функции в отдельных точках,
- представить площадь заданной фигуры как алгебраическую сумму определенных интегралов и вычислить эти интегралы с некоторой точностью ε_2 по квадратурной формуле - метод трапеций,

Величины ε_1 и ε_2 подобраны аналитически с использованием оценки погрешности, взятой из литературы, и знаний и навыков, полученных в курсе математического анализа, так, чтобы гарантировалось вычисление площади фигуры с точностью ε

Математическое обоснование

Для применимости метода Ньютона и метода секущих нахождения приближенных корней уравнения $f = g$ на промежутке [a, b] необходимо, чтобы функция $f - g$ удовлетворяла следующим требованиям:

- функция $f - g$ непрерывна на [a, b] и имеет на нем непрерывную первую производную,
- $(f(a) - g(a)) \cdot (f(b) - g(b)) < 0$,
- $(f - g)$' монотонна на [a, b],
- $f - g$ дважды дифференцируема на [a, b].

Подберем отрезки, на которых будем искать точки пересечения выберем, опираясь на приведенные выше требования. Выберем a = -5 и b = 5 и убедимся, что наши функции удовлетворяют перечисленным выше требованиям.

Рассмотрим уравнение $f_1 - f_2 = 0$. Данная функция непрерывна, как композиция непрерывных фунцкий. Её производная равна $\frac{-8x}{(x^2+1)^2} - 3x^2$. Она так же непрерывна и сохраняет отрицательный знак на промежутке [a, b]. Вторая производная у данной функции тоже существует. Таким образом все необходимые условия выполнены

Аналогично рассмотрим уравнение $f_1 - f_3 = 0$. Данная функция непрерывна, как композиция непрерывных фунцкий. Её производная равна $\frac{-8x}{(x^2+1)^2} - ln(2) * 2^{-x}$. Она так же непрерывна и сохраняет отрицательный знак на промежутке [a, b]. Вторая производная у данной функции тоже существует. Таким образом все необходимые условия выполнены.

Рассмотрим уравнение $f_2 - f_3 = 0$. Данная функция непрерывна, как композиция непрерывных фунцкий. Её производная равна $3x^2 - ln(2) * 2^{-x}$. Она так же непрерывна и сохраняет положительный знак на промежутке [a, b]. Вторая производная у данной функции тоже существует. Таким образом все необходимые условия выполнены.

Следовательно, мы можем искать наши корни на промежутке [-5, 5]. Ниже приведены графики кривых (рис. 1).

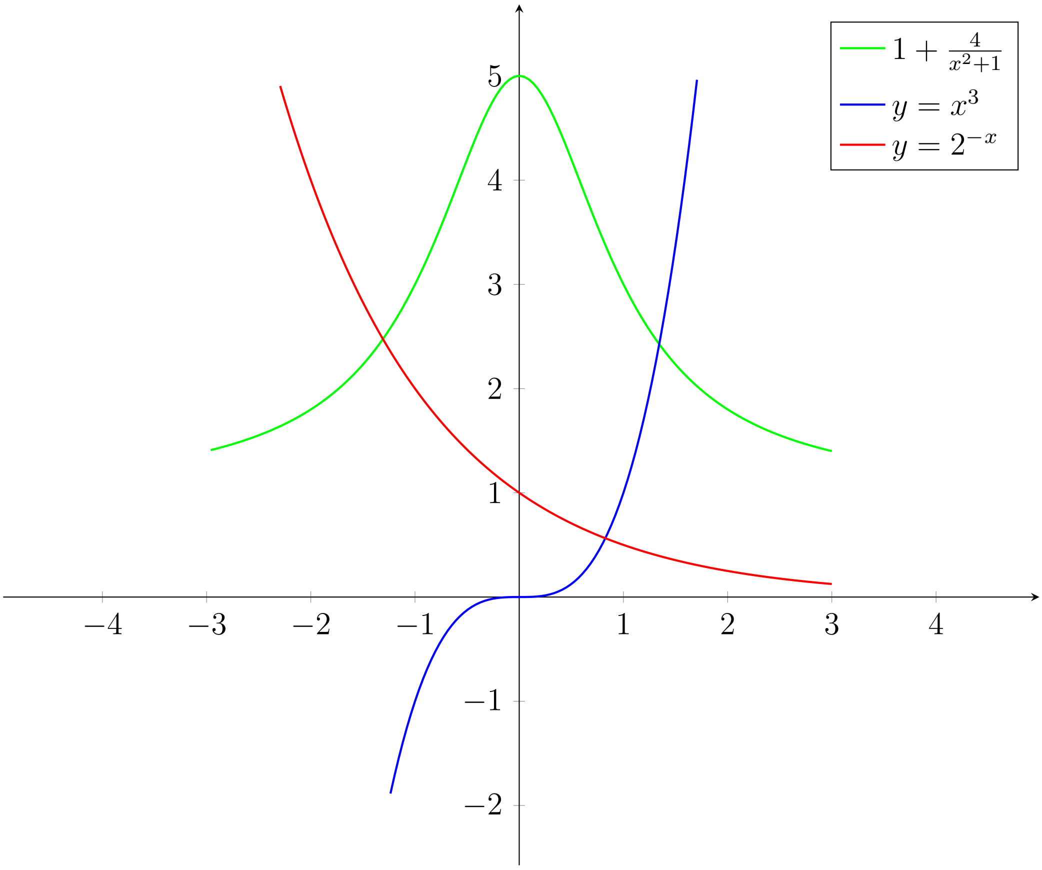

Рис. 1: Плоская фигура, ограниченная графиками заданных уравнений

Для выбора ε_1 и ε_2, оценим сначала отклонение n-го приближения от точного значения корня с при использовании методов хорд и касательных в отдельности. Применим к выражению $f(x_n) = f(x_n) - f(c)$ формулу Лагранжа, будем иметь $f(x_n) = (x_n - c) \cdot f'(\xi_n)$. Отсюда получим следующую оценку: $|x_n - c| \leq \frac{|f(x_n)|}{m}$, где m - минимальное значение $|f'(x_n)|$ на сегменте [a, b]. [1] Для этого достаточно взять $\varepsilon_1 = 0.001$.

Теперь оценим погрешность приближенного вычисления интеграла при использовании метода трапеций. На самом деле он не дает улучшения по сравнению с методом прмоугольников, поэтому погрешность вычеслений такая же. Возьмем оценку из книги [1]. Погрешность равна $\frac{(b-a)^3}{24n^2} f''(\xi), a \leq \xi \leq b$. Вычисляя интеграл с точностью ε_2, получаем $b - a = n \cdot \varepsilon_2$. Так как $f''(\xi) \leq 1$, то для достижения погрешности ε, достаточно взять такое ε_2, чтобы $\frac{(b-a) \cdot \varepsilon_2^2}{24} \leq 0.001$. Так как на промежутке, которыq я рассматриваю $b - a \leq 10$, достаточно взять $\varepsilon_2^2 \leq 0.0024$, то есть возьмем $\varepsilon_2 = 0.001$.

Результаты экспериментов

В данном разделе приведены результаты проведенных вычислений: координаты точек пересечения (таблица 1) и площадь полученной фигуры.

Кривые	x	y
1 и 2	1.3435	2.4258
1 и 3	-1.3075	2.4757
2 и 3	0.8258	0.5640

Таблица 1: Координаты точек пересечения

Ниже результаты проиллюстрированы графиком (рис. 2).

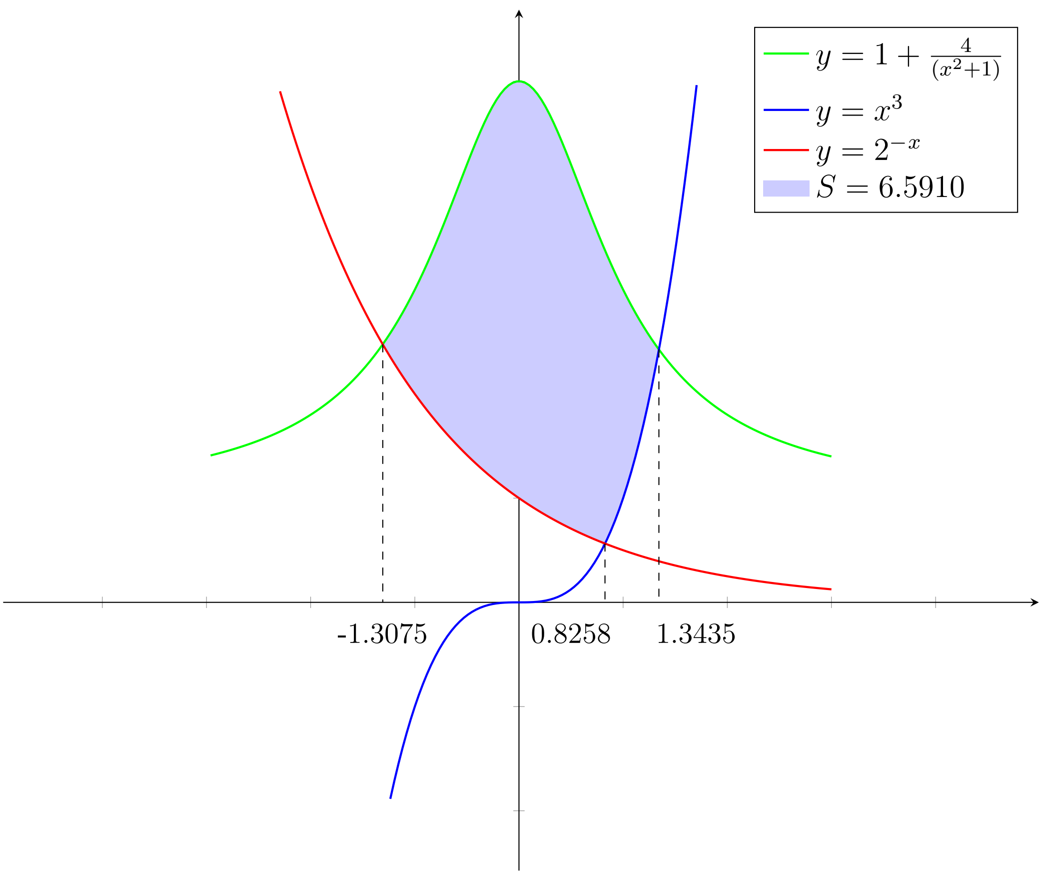

Рис. 2: Плоская фигура, ограниченная графиками заданных уравнений

Структура программы и спецификация функций

Функции написанные на языке ассемблер и находящиеся в файле function.asm и derivative.asm:

- f1 - функция, вычисляющая значение $y = 1 + \frac{4}{x^2+1}$ в точке x. Принимает значение x типа double, возвращает значение функции типа double. Соглашение cdecl.
- f2 - функция, вычисляющая значение $y = x^3$ в точке x. Принимает значение x типа double, возвращает значение функции типа double. Соглашение cdecl.
- f3 - функция, вычисляющая значение $y = 2^{-x}$ в точке x. Принимает значение x типа double, возвращает значение функции типа double. Соглашение cdecl.
- df1 - функция, вычисляющая значение производной функции f1 в точке x. Принимает значение x типа double, возвращает значение функции типа double. Соглашение cdecl.
- df2 - функция, вычисляющая значение производной функции f2 в точке x. Принимает значение x типа double, возвращает значение функции типа double. Соглашение cdecl.
- df3 - функция, вычисляющая значение производной функции f3 в точке x. Принимает значение x типа double, возвращает значение функции типа double. Соглашение cdecl.

Функции, написанные на языке Си:

Функция root-newton-method, root-secant-method, integral находящиеся в файле calc.c. Имеет прототип:

- double root-newton-method(double (*)(double), double (*)(double), double (*)(double), double (*)(double), double, double, double)
  Принимает указатели на функции и их производные(требуются для нахождения корней уравнения), интервал на котором необходимо искать корень и погрешность вычисления. Возвращает приближенное значение найденного корня.
- double root-secant-method(double (*)(double), double (*)(double), double, double, double)
  Принимает указатели на функции, интервал на котором необходимо искать корень и погрешность вычисления. Возвращает приближенное значение найденного корня.
- double integral(double (*)(double), double, double , double)
  Принимает указатели на функции, интервал на котором необходимо искать интеграл и погрешность вычисления. Возвращает приближенное значение интеграла.

Сборка программы (Make-файл)

Итоговый проект project.exe собирается из 4 объктных модулей:
main.o calc.o function.o derivative.o

Они в свою очередь собираются из 4 файлов:
main.c calc.c function.asm derivative.asm

Необходимо добавить, что файлы main.c и calc.c компилируются с ключом -D (-Dhord или -Dnewton), который определяет с помощью какого метода приближенных решений уравнений будут находиться корни.

Все фунцкии, использующиеся в программе описаны в библиотеке: library.h

Итого проект сосотоит их следующих модулей:

- main.c : оболочка программы. В ней происходит обработка опций, вводимых в командной строке
- calc.c : в данном файле находятся функции, вычисляющие корни уравнений, а также функция нахождения интеграла
- function.asm : в данном файле написаны функции, предлагаемые условием задачи
- derivtive.asm : в данно файле находятся производные функйи из файла function.asm
- library.h : описаны прототипы всех используемых функций

Для сборки проекта написан makefile, в котором отражены все зависисмости и прописаны все необходимые для компиляции ключи

Отладка программы, тестирование функций

Для того, чтобы удоствоериться, что программа работате правильно, необходиом было протестировать 3 функции:

- root-newton-method
- root-secant-method
- integral

Протестировать данные функции позволяют опции командной строки -root и -integral, после которых следут номер(или номера) функций, границы промежутка [a, b], на котором будут производиться вычисления и число ε ,определяющее точность вычеслений.

Проверка производится путем сравнения аналитических расчетов и ответов, которые выводит программа. Аналитические расчеты производятся на сайте wolframalpha.com

Функции integral и root тестировались на функция данных в условии, а также на дополнительных функцих, описанных в файле test.c , который не входит в сборку основной программы и требуется лишь для дополнительного тестирования.

Таким образом, убедившись, что аналитически расчеты совпадают с расчетами программы, можно сделать вывод, что функции, описанные в файлах calc.c, function.asm, derivative.asm работают верно!

Программа на Си и на Ассемблере

Исходные тексты программы, написанные на языках Си и Ассемблер имеются в архиве, который приложен к этому отчету.

Анализ допущенных ошибок

В ходе написания проекта были допущены 2 ошибки:

- Первая ошибка была допущена при написании функций в файле function.asm. Была попытка загрузить передаваемую в качестве аргумета переменную double x на стек (для вызова функции pow) с помощью команды push qword[ebp+8]. Однако я забыл, что программа собираетсяи работает на 32битной системе. Поэтому при компиляции выходила логичная ошибка о недопустимости применения данной команды. Решение оказалось следущим: класть переменную x на стек по 4 байта.

- Вторая ошибка заключалась в передаче количества иттераций в основную программу. По ошибке переменная iterations была объялена локально в файле calc.c, и при попытке сделать extern int iterations ничего не происходило. Проблема была решена двумя способами:
  1) можно было просто объявить переменную iterations глобально в файле calc.c, а потом, как и предполагалось, сделать extern int iterations в оболочку программы main.c
  2) Или же можно было объявить переменную iterations в библиотеке library.h, и так как библиотека подключается в обоих Си-файлах, то переменная была бы видна, и никакой ошибки не возникало бы.

Список литературы

[1] Ильин В. А., Садовничий В. А., Сендов Бл. Х. Математический анализ. Т. 1 — Москва: Наука, 1985.