Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное   
учреждение высшего образования

Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

**Отчет по лабораторной работе**

**«Вычисление функций sin(x), cos(x), exp^(x), ln(1+x) в окрестности 0»**

**Выполнил**:

студент/ка группы 3821Б1ПМ2

Василевский А.П.

**Проверил**:

преподаватель каф. МОСТ,

Волокитин В.Д.

Нижний Новгород

2022

**Содержание**

[Постановка задачи 3](#_Toc98783025)

[Метод решения 4](#_Toc98783026)

[Руководство пользователя 6](#_Toc98783027)

[Описание программной реализации 8](#_Toc98783028)

[Подтверждение корректности 11](#_Toc98783029)

[Заключение 13](#_Toc98783030)

# Постановка задачи

Задачами лабораторной работы являются реализация алгоритмов вычисления функций sin(x), cos(x), exp^(x), ln(1+x) в окрестности 0, суммирование в прямом и обратном порядке, а также попарное суммирование на языке программирования Си, измерение абсолютной и относительной ошибки вычислений.

# Метод решения

**Сортировка выбором**

Суть всех алгоритмов заключается в вычислении приблизительного значения функции при помощи разложения её в ряд Маклорена.

Функцию , имеющую (n+1) производных в точке x = 0, можно представить по формуле Маклорена:

Ряд Маклорена для функций sin(x), cos(x), exp^(x), ln(1+x):

Точность вычисления этих функций повышается с увеличением количества элементов в ряде Маклорена, однако в знаменателе элементов присутствует факториал, которой быстро возрастает и поэтому может принимать значения больше чем может хранить double даже при небольших х, из-за этого при вычислении значения элементов по формуле может возникнуть большая погрешность.

Для того чтобы появилась возможность вычислить как можно больше элементов с сохранением наибольшей точности прибегнем к следующему методу, который рассмотрим на примере функции exp^(x).

Первый элемент равняется 1, а каждый последующий можно получить из предыдущего для этого достаточно поделить предыдущий элемент на порядковый номер текущего и умножить на значение х. За счёт такого способа вычисления элементов удаётся избежать переполнения факториалом переменной типа double. Однако точность вычислений этим способом зависит также и от порядка суммирования, например, при суммировании по порядку по рядку точность будет меньше чем при суммировании с конца.

# Руководство пользователя

При запуске программы будет выводиться сообщение

“select the function:

1 - sin(x)

2 - cos(x)

3 - exp^(x)

4 - ln(1+x)“, после чего пользователю необходимо ввести цифру, соответствующую функции, если введённое число не соответствует ни одной функции, то программа прервётся.

Далее будет выведено сообщение

“choose the summation method:

1 - summation in order

2 - summing from the end with an array

3 - summation in pairs

4 - print”, предлагающие выбрать способ суммирования.

Вне зависимости от выбранного числа выведется “x = ” после чего нужно ввести чисто приблизительное значение которого нужно вычислить.

Если введённое число не соответствует ни одному способу суммирования, то программа выведет “ ----” и завершится.

Если введённое число соответствует способу суммирования от 1 до 3, то программа выведет

“ sin(<значение х>)~<вычисленное значение синуса от х>

sin(<значение х>)=< значение библиотечного синуса от х >

absolute error = <абсолютная погрешность> relative error = <относительная погрешность>”.

Если введённое число соответствует способу суммирования 4, то программа (после ввода х) выведет ”step =” b , и будет ожидать ввода числа, после чего выведет следующее сообщение

“error:

1 - absolute

2 - relative”, после чего в зависимости от выбранного типа ошибки запишет в файл (при его отсутствии создаст) “table.txt” в папке проекта значения погрешностей суммирования в следующем виде:

<прямое> <обратное> <попарное>

Для значений равных от -|x| до |x| с шагом в |step|, а в конце файла выведет в таком же виде сумму соответствующих погрешностей

Если введённое число не соответствует ни одному типу ошибки, то программа завершит своё выполнение и выведет “ ----”.

# Описание программной реализации

void sinchoose() – вызывается при выборе вычисления . Предоставляет выбор числа и способа суммирования на основе введённых данных вызывает функцию для подсчёта и выводит приблизительное значение, значение библиотечной функции и ошибку (абсолютную и относительную).

void coschoose() – вызывается при выборе вычисления . Предоставляет выбор числа и способа суммирования на основе введённых данных вызывает функцию для подсчёта и выводит приблизительное значение, значение библиотечной функции и ошибку (абсолютную и относительную).

void expchoose() – вызывается при выборе вычисления . Предоставляет выбор числа и способа суммирования на основе введённых данных вызывает функцию для подсчёта и выводит приблизительное значение, значение библиотечной функции и ошибку (абсолютную и относительную).

void lnchoose() – вызывается при выборе вычисления . Предоставляет выбор числа и способа суммирования на основе введённых данных вызывает функцию для подсчёта и выводит приблизительное значение, значение библиотечной функции и ошибку (абсолютную и относительную).

double \_1sin(double num) - на вход функция принимает число типа double и возвращает примерное значение синуса от этого числа используя метод прямого суммирования.

double \_2sin(double num) - на вход функция принимает число типа double и возвращает примерное значение синуса от этого числа используя метод суммирования с конца при помощи массива.

double \_3sin(double num) - на вход функция принимает число типа double и возвращает примерное значение синуса от этого числа используя метод попарного суммирования.

double \_1cos(double num) - на вход функция принимает число типа double и возвращает примерное значение косинуса от этого числа используя метод прямого суммирования.

double \_2cos(double num) - на вход функция принимает число типа double и возвращает примерное значение косинуса от этого числа используя метод суммирования с конца при помощи массива.

double \_3cos(double num) - на вход функция принимает число типа double и возвращает примерное значение косинуса от этого числа используя метод попарного суммирования.

double \_1exp(double num) - на вход функция принимает число типа double и возвращает примерное значение от этого числа используя метод прямого суммирования.

double \_2exp(double num) - на вход функция принимает число типа double и возвращает примерное значение от этого числа используя метод суммирования с конца при помощи массива.

double \_3exp(double num) - на вход функция принимает число типа double и возвращает примерное значение от этого числа используя метод попарного суммирования.

double \_1ln(double num) - на вход функция принимает число типа double и возвращает примерное значение ln(1+num) от этого числа используя метод прямого суммирования.

double \_2ln(double num) - на вход функция принимает число типа double и возвращает примерное значение ln(1+num) от этого числа используя метод суммирования с конца при помощи массива.

double \_3ln(double num) - на вход функция принимает число типа double и возвращает примерное значение ln(1+num) от этого числа используя метод попарного суммирования.

# Подтверждение корректности

Для подтверждения корректности во время работы программы при выборе метода суммирования выберем четвёртый вариант, введём “x” и “step”, а также выберем интересующую нас ошибку. Программой будет создан файл в котором будет записана ошибка для трёх способов суммирования на интервале от -|x| до |x| с шагом в step, а в последней строчке будет записана суммарная погрешность для каждого из способов суммирования. На основе суммарной погрешности можно сделать вывод о точности вычислений.

Абсолютная ошибка вычисляется как модуль разности между значением реализованной функции и её библиотечным аналогом, корректность которого подтверждена. Относительная вычисляется как абсолютная делённая на число.

Суммарная абсолютная и относительная ошибка для Sin(x), Cos(x) и Exp^(x) вычислялась на интервале от -3 до 3 с шагом в 0,001, а для Ln(1+x) ошибка вычислялась на интервале от -0,7 до 0,7 с шагом в 0,0003.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Функция | Тип суммарной ошибки | Прямое суммирование | Обратное суммирование | попарное суммирование |
| Sin(x) | абсолютная | 5566\*10^(-16) | 4070\*10^(-16) | 4895\*10^(-16) |
| относительная | 3965\*10^(-16) | 1867\*10^(-16) | 3244\*10^(-16) |
| Cos(x) | абсолютная | 5166\*10^(-16) | 4118\*10^(-16) | 5570\*10^(-16) |
| относительная | 7589\*10^(-16) | 2165\*10^(-16) | 3066\*10^(-16) |
| Exp^(x) | абсолютная | 37400\*10^(-16) | 11639\*10^(-16) | 24172\*10^(-16) |
| относительная | 31028 \*10^(-16) | 6224\*10^(-16) | 12903\*10^(-16) |
| Ln(1+x) | абсолютная | 5382\*10^(-16) | 2121\*10^(-16) | 4202\*10^(-16) |
| относительная | 33300\*10^(-16) | 26755\*10^(-16) | 30847\*10^(-16) |

На основе этих данных можно сделать вывод о том, что функции вычисляются с достаточной точностью и прямое суммирование обладает большей точностью чем попарное, которое в свою очередь обладает большей точностью по сравнению с прямым суммированием.

# Заключение

В ходе лабораторной работы были реализованы алгоритмы вычисления функций sin(x), cos(x), exp^(x), ln(1+x) в окрестности 0 по ряду Маклорена с использованием прямого, обратного и попарного суммирования на языке программирования Си и проведена проверка корректности. Были описаны алгоритмы работы данных сортировок и проведено измерение погрешности.