ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ,
МЕХАНИКИ И ОПТИКИ»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

Метод деления пополам, метод хорд Метод простых итераций

Студент**:** Назирджанов Н.Ф

Преподаватель: Перл О.В.

Санкт-Петербург
2023г.

Описание метода

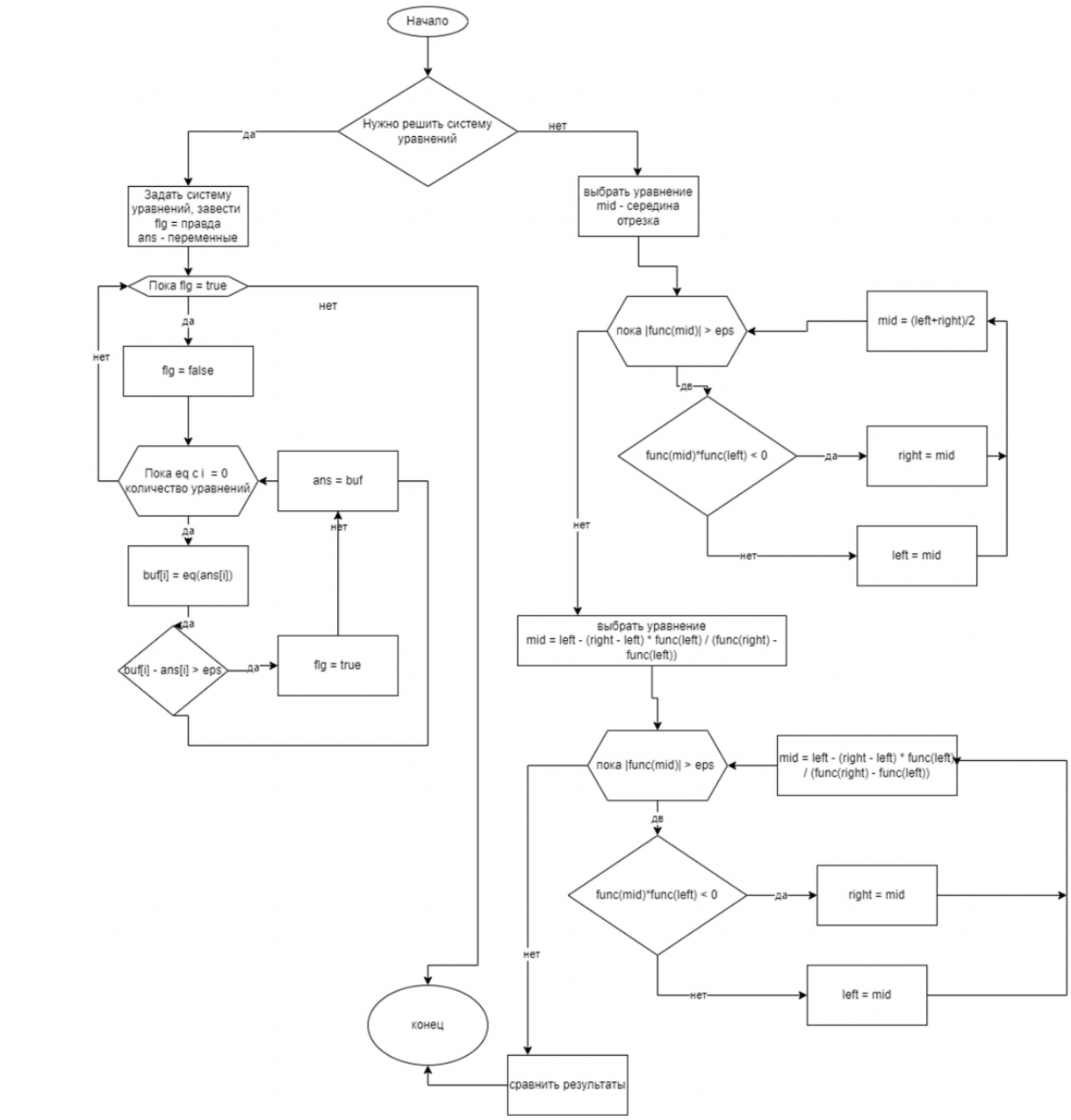
1) Метод деления пополам.

   Этот метод помогает найти решение уравнения, для которого нет аналитического решения, используя последовательное деление интервала на две половины. Мы начинаем с интервала, в котором мы предполагаем, что находится корень, и затем находим середину этого интервала. Мы затем смотрим, какое значение принимает функция в середине интервала и сравниваем его со значениями на концах интервала. Если значения на концах интервала имеют разные знаки, то мы продолжаем поиск корня в этом интервале, делая новый интервал равным половине первоначального интервала. Если значения на концах интервала имеют одинаковые знаки, то мы отбрасываем этот интервал и продолжаем поиск корня в другой половине интервала. Мы повторяем этот процесс до тех пор, пока не найдем корень или пока не достигнем заданной точности.

1. Метод хорд очень похож на метод деления пополам. Его принцип работы такой же, за исключением того, как мы берем точку на отрезке. Мы будем брать точку не по середине, а ближе к тому концу отрезка, где модуль значения функции больше. Отрезок делится в таком же отношении, как и значения функции в левом и правом концах отрезка.

2) Метод простых итераций необходим для решения нелинейных систем уравнений. Из каждого уравнения выражается i-ая переменная (например, из первого уравнения получается x1 = f(x1, x2, .. ,xn) и т.д. Дальше итеративно считаем значения переменных, пока хотя бы одна переменная меняется больше, чем наперед заданное эпсилон.

Блок-схема


Начало
Нужно решить систему уравнений
да
нет
Задать систему уравнений, завести flg = правда ans - переменные
выбрать уравнение mid - середина отрезка
Пока flg = true
нет
да
flg = false
нет
Пока eq с i = 0 количество уравнений
ans = buf
нет
да
buf[i] = eq(ans[i])
да
buf[i] - ans[i] > eps
да
flg = true
пока |func(mid)| > eps
mid = (left+right)/2
дв
func(mid)*func(left) < 0
да
right = mid
нет
left = mid
нет
выбрать уравнение mid = left - (right - left) * func(left) / (func(right) - func(left))
пока |func(mid)| > eps
mid = left - (right - left) * func(left) / (func(right) - func(left))
дв
func(mid)*func(left) < 0
да
right = mid
нет
left = mid
нет
конец
сравнить результаты

Реализация:

```cpp
Ans BisectionMethodImpl::get_root(const std::function<double(double)> &func, double left, double right) {
    if (func(left) * func(right) >= 0) {
        return not_success;
    }
    double mid = (left + right) / 2;
    while (!DoubleComparator::double_equals(func(mid), 0.0)) {
        if (func(mid) * func(left) < 0) {
            right = mid;
        } else {
            left = mid;
        }
        mid = (left + right) / 2;
    }
    return Ans{true, mid};
}

Ans ChordMethod::get_root(const std::function<double(double)> &func, double left, double right) {
    if (func(left) * func(right) >= 0) {
        return not_success;
    }
    double mid = get_mid(func, left, right);

    while(!DoubleComparator::double_equals(func(mid), 0)) {
        if(func(mid) * func(left) < 0) {
            right = mid;
        } else {
            left = mid;
        }
        mid = get_mid(func, left, right);
    }
    return Ans{true, mid};
}

vector<double> IterationMethod::solve_system(vector<function<double(vector<double>)>> system, double
precision) {
    vector<double> ans = std::vector<double>(system.size());
    fill(ans.begin(), ans.end(),1);
    vector<double> buf = ans;
    bool need_iter = true;
    while (need_iter) {
        need_iter = false;
        for (int i = 0; i < system.size(); i++) {
            buf[i] = system[i](ans);
            if(abs(buf[i] - ans[i]) > precision) need_iter = true;
        }
        ans = buf;
    }
    return ans;
}
}
            //"There are only two hard things in Computer Science: cache invalidation and naming things."
```

```
Do you want to solve equation or system? Type 1 if system, else 2
2
what equition you want to solve: polynomial, logarithmic, or trigonometrical? type number
1
Bisectional: 1.3652300089597702026
Chord: 1.3652299546434190081
Difference: 5.431635119457212113e-08
Process finished with exit code 0
```

Вывод:

Выбор метода численного решения зависит от конкретной задачи и ее условий. Каждый из перечисленных методов имеет свои преимущества и ограничения, и выбор метода зависит от требуемой точности, формы функции, ее производной и промежутка, на котором нужно решить уравнение.

Метод деления пополам обеспечивает сходимость к решению с гарантированной точностью, однако его сходимость медленнее, чем у других методов. Он хорошо подходит для функций с одним корнем на заданном интервале и малым количеством локальных экстремумов.

Метод хорд является более быстрым, но он может не сойтись к решению, если функция имеет локальные минимумы и максимумы вблизи корня, а также может сходиться к другому корню в случае, если функция имеет более одного корня на заданном интервале.

Метод касательных обычно сходится быстрее, чем метод хорд, но он может не сходиться к решению, если функция имеет локальные минимумы и максимумы вблизи корня, а также может сходиться к другому корню в случае, если функция имеет более одного корня на заданном интервале.

Метод простой итерации является более общим методом и может использоваться для решения систем нелинейных уравнений, а не только для одного уравнения. Он может сходиться быстрее, чем метод деления пополам, но может не сходиться, если функция имеет локальные минимумы и максимумы вблизи корня или если матрица Якоби не удовлетворяет условию Липшица.

Таким образом, для выбора оптимального метода следует учитывать все эти факторы и проводить тестирование разных методов для конкретной задачи.

Метод Ньютона и метод простой итерации являются двумя распространенными методами численного решения систем нелинейных уравнений. Оба метода могут использоваться для поиска решения системы уравнений, но они имеют различные преимущества и ограничения.

Метод Ньютона:

Метод Ньютона является одним из самых эффективных методов численного решения систем нелинейных уравнений. Он основан на разложении функции в ряд Тейлора до первого порядка и применяет формулу Ньютона для нахождения корня. Этот метод может сходиться к решению очень быстро, если начальное приближение достаточно близко к решению, и функция является гладкой. Однако он может не сходиться, если начальное приближение находится вблизи локального экстремума или если матрица Якоби функции является вырожденной.

Метод простой итерации:

Метод простой итерации является более общим методом и может использоваться для решения систем нелинейных уравнений, а не только для одного уравнения. Он основан на переписывании системы уравнений в эквивалентную форму итерационного процесса, в котором каждый элемент нового вектора

определяется как функция предыдущего вектора. Этот метод может сходиться к решению, если заданный итерационный процесс является сжимающим. Однако этот метод может сходиться медленнее, чем метод Ньютона, и может требовать больше итераций для достижения точности.

Если матрица Якоби функции является несингулярной и начальное приближение достаточно близко к решению, метод Ньютона может быть более эффективным. В других случаях, когда матрица Якоби функции является сингулярной или начальное приближение находится далеко от решения, метод простой итерации может быть более подходящим выбором.