Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
факультет программной инженерии и компьютерной техники

# Лабораторная работа №3

по дисциплине

Вычислительная математика

Выполнил:
Студент группы P3230
Толстых М.А.

Преподаватель:
Перл И. А.
Перл О. В.

Санкт-Петербург, 2024

# Описание метода

Метод простых итераций.

Задается точность ε. Дана система нелинейных уравнений:

$$X = \Phi(X), \text{где } \Phi(X) = \begin{cases} \varphi_1(x_1, x_2, \dots, x_n) \\ \varphi_2(x_1, x_2, \dots, x_n) \\ \dots \dots \dots \dots \dots \dots \dots \\ \varphi_n(x_1, x_2, \dots, x_n) \end{cases}$$

$\varphi_i = x_i + f_i$, $i$=1,2, ...$n$.

Выбираем начальное приближение $X^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})$

Для ускорения сходимости итерационного процесса формула итерационного процесса имеет вид:

$$x_i^{(k+1)} = x_i^{(k)} \pm T_i f_i(X^k),$$

где $T_i$ - параметры, которые принимаются на изначальном этапе.

Полагаем переменную, k нумерующую приближения, равной нулю.

Формула (k +1) - е приближение по приближённой формуле:

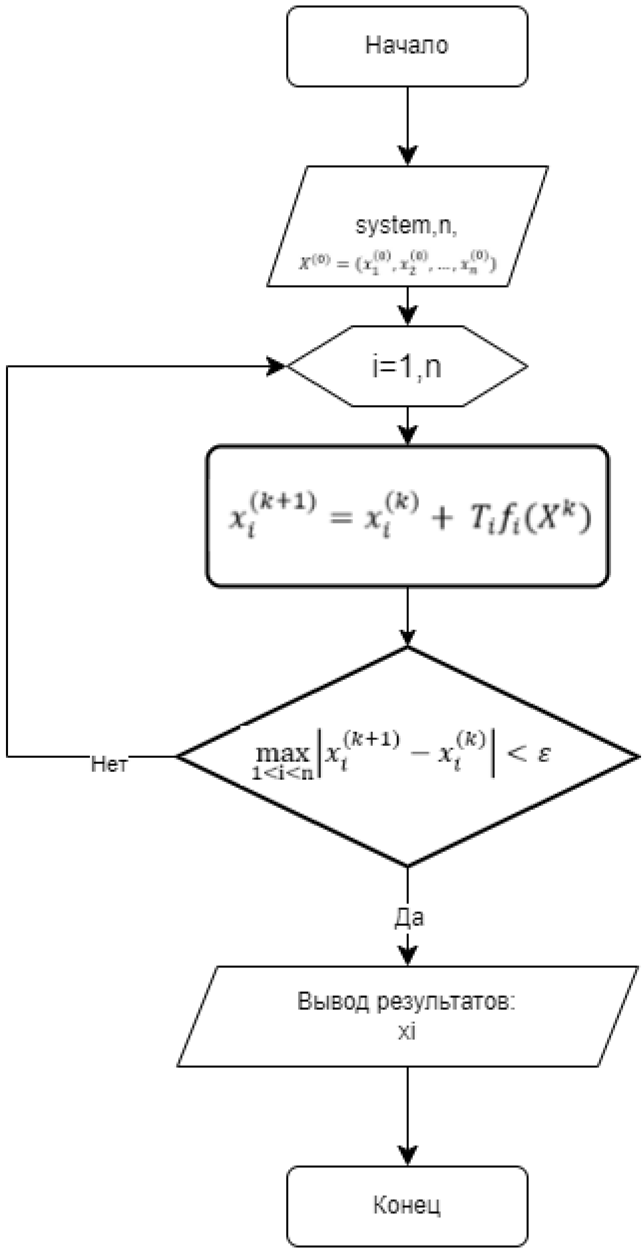
$$x_i^{(k+1)} = x_i^{(k)} + T_i f_i(X^k)$$

Сравниваем полученное приближение с предыдущим, проверяем условие:

$$\max_{1<\mathrm{i}<\mathrm{n}} \left| x_i^{(k+1)} - x_i^{(k)} \right| < \varepsilon$$

Соответственно, если условие выполнено, то решение считается найденным на (k +1) - м шаге и итеративный процесс закончен, в противном случае полагаем k = k +1 и переходим к вычислению следующего приближения.

## Блок-схема численного метод



# Код

```
def solve_by_fixed_point_iterations(system_id, number_of_unknowns,
initial_approximations):
    max_iter = 100
    epsilon = 1e-5
    step = 0.0001

    funcs = get_functions(system_id)
    n = number_of_unknowns
    x = initial_approximations.copy()
    x_prev = [0.0] * n

    for _ in range(max_iter):
        for i in range(n):
            x_prev[i] = x[i]

        for i in range(n):
            try:
                x_prev[i] = x[i] + step * funcs[i](x)
            except OverflowError:
                return initial_approximations

        if all(abs(x_prev[i] - x[i]) < epsilon >= max_iter for i in
range(n)):
            break
        x = x_prev
    return x
```

Метод простых итераций. Здесь параметр T = step = 0.0001

## Примеры работы программы

1. Тест на простой системе из двух неизвестных, у которой id = 1.

```
>? 1
>? 2
>? 0.5
>? 0.5
0.5048151233371783
0.5012576323188581
```

2. Тест на простой системе из двух неизвестных, у которой id = 2.

```
>? 2
>? 2
>? 1.0
>? 1.0
1.0043238235387957
0.942654639918027
```

3. Тест на простой системе из трех неизвестных, у которой id = 3.

```
>? 3
>? 3
>? 0.5
>? 1.0
>? -0.5
0.5166910916934232
1.0146267102434607
-0.4952567510153745
```

4. Тест с системой, для которой не определены функции (default).

```
>? 2
>? 2
>? -2
>? 2
-2.0
2.0
```

5. Тестовый случай с учетом возникновения OverflowError в одной из функций при итерациях, программа завершила выполнение и вернула начальные приближения, а не решение системы уравнений.

```
>? 2
>? 2
>? -1.9
>? 2.1
-1.9795434302941284
2.0754504260511415
```

## Вывод

Этот метод простых итераций для решения СНУ может быть удобен, поскольку это одношаговый итерационный процесс. Но метод может не решить систему уравнений в случаях: если программа не смогла найти корни после максимального количества итераций или если разница между текущими и предыдущими значениями неизвестных меньше заданной погрешности epsilon, то программа завершит выполнение. Если при обновлении значений произошла ошибка OverflowError, программа вернет начальные приближения. Примеры таких случаев есть выше.
При тестировании программы следует учитывать, что результаты могут зависеть от выбора начальных приближений, шага итерации и самой функции системы уравнений. Для некоторых СНУ метод может не сойтись к решению из-за выбора неподходящих начальных приближений или параметров метода.
Используя данный метод можно получить последовательность приближённых значений, сходящуюся к точному решению системы (на примерах 1 и 2 можно заметить множество цифр, которое является нецелой частью значения). С помощью итерационных методов

возможно решать эффективнее задачи большой размерности за приемлемое время с минимумом операций.
Метод простых итераций схож с методом Зейделя. Метод Зейделя тоже является итерационным методом. Этот метод является модификацией метода простых итераций, где после задания начального приближения $x^{(0)}$ вместо параллельного итерирования производится последовательное итерирование, причем на каждой итерации в каждое последующее уравнение подставляются значения неизвестных, полученных из предыдущих уравнений.
Метод Зейделя при вычислении (k+1)-ого приближения неизвестного $x_i (i > 1)$ используется уже вычисленные ранее (k+1)-ое приближение неизвестных.

Вычисления по методу Зейделя:

$$\begin{cases} \boldsymbol{x_1^{(k+1)}} = \varphi_1(x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)}) \\ x_2^{(k+1)} = \varphi_2(\boldsymbol{x_1^{(k+1)}}, x_2^{(k)}, \dots, x_n^{(k)}) \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ x_n^{(k+1)} = \varphi_n(\boldsymbol{x_1^{(k+1)}}, \boldsymbol{x_2^{(k+1)}}, \dots, x_{n-1}^{(k+1)}, x_n^{(k)}) \end{cases}$$

где жирным отмечены значения, которые берутся из предшествующих уравнений на текущей итерации.
Однако, области сходимости методов совпадают.

Алгоритмическая сложность метода составляет для каждой итерации - $O(n\ )$, где n – количество уравнений в системе, оно в свою очередь будет зависеть от заданной точности, шага и начального приближения.

Анализ численной ошибки в самом численном методе:

Анализ численной ошибки в численном методе простых итераций включает в себя такие факторы, как ошибки округления, потому что при каждой итерации происходит округление чисел, что может привести к накоплению ошибок округления. Это особенно заметно при большом количестве итераций.  потерю значимости, начальное приближение. Неудачный выбор начального приближения или точности также может привести к увеличению численной ошибки.

Потеря значимости, так как при вычислении следующего приближения может происходить потеря значимости, особенно если приближения близки к 1 или к 0.