Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Методы численного анализа»

ОТЧЕТ

к лабораторной работе №4

на тему:

«РЕШЕНИЕ СИСТЕМ НЕЛИНЕЙНЫХ УРАВНЕНИЙ»

БГУИР 1-40-04-01

Выполнил студент группы 253504
Дмитрук Богдан Ярославович
31.10.2023

(дата, подпись студента)

Проверил
Анисимов Владимир Яковлевич

(дата, подпись преподавателя)

Минск 2023

СОДЕРЖАНИЕ

1. ЦЕЛИ ВЫПОЛНЕНИЯ ЗАДАНИЯ

1. Изучить методы численного решения систем нелинейных уравнений (метод простых итераций, метод Ньютона)
2. Составить программу численного решения нелинейных уравнений методами простых итераций и Ньютона
3. Составить алгоритм решения нелинейных уравнений методами простых итераций и Ньютона
4. Проверить правильность работы программы на тестовых примерах
5. Численно решить нелинейное уравнение заданного варианта
6. Сравнить число итераций, необходимого для достижения заданной точности вычисления разными методами

2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Краткие теоретические сведения. Пусть дана система нелинейных уравнений (система не линейна, если хотя бы одно из входящих в нее уравнений не линейно):

$$\begin{cases} f_1(x_1,\dots,x_n)=0 \\ f_2(x_1,\dots,x_n)=0 \\ \dots\dots\dots\dots \\ f_n(x_1,\dots,x_n)=0 \end{cases}$$

Мы можем записать систему в более компактной векторной форме

$$f(x)=0, \qquad (4.1)$$

где $f=(f,\dots,f)$, $x=(x,\dots,x)^T$.

Для решения системы (4.1) иногда можно применить метод последовательного исключения неизвестных, который приводит решение системы к решению одного уравнения с одним неизвестным. Однако в подавляющем большинстве случаев систему уравнений (4.1) решают итерационными методами. Для решения системы (4.1) существует набор методов, из которых рассмотрим простейшие методы: метод простых итераций, базирующийся на принципе сжимающих отображений и метод Ньютона (многомерный аналог метода касательных).

Метод простых итераций. Чтобы воспользоваться методом простых итераций, необходимо предварительно привести систему к следующему виду:

$$\begin{cases} x_1=\varphi_1(x_1,\dots,x_n) \\ x_2=\varphi_2(x_1,\dots,x_n) \\ \dots\dots\dots\dots \\ x_n=\varphi_{n1}(x_1,\dots,x_n) \end{cases}$$

Или в векторной форме

$$\bar{x}=\varphi(\bar{x}) \qquad (4.2)$$

где $\vec{\varphi}=(\varphi_1,\varphi_2,\dots\dots,\varphi_n)$.

Исходя из некоторого начального приближения $\bar{x}^0$, построим итерационную последовательность точек

$\bar{x}^k = \varphi(\bar{x}^{k-1})$, $k=1,2,....$

Пусть точка $\bar{x}^0$ есть некоторое начальное приближение к решению. Рассмотрим δ-окрестность $U_\delta(\bar{x})$ этой точки. В силу принципа сжимающих отображений итерационная последовательность

$\bar{x}^k = \varphi(\bar{x}^{k-1})$, $k=1,2,....$

будет сходиться, если существует число q<1 такое, что

выполнено условие:

$$\left\|\varphi(\bar{x}^1) - \varphi(\bar{x}^2)\right\| \le q\left\|\bar{x}^1 - \bar{x}^2\right\|, \quad \forall \bar{x}^1, \bar{x}^2 \in U_\delta(\bar{x}^0), \qquad (4.3)$$

называемое условием сжатия для отображения φ. В частности, это условие всегда выполняется, если векторная функция φ непрерывно дифференцируема и норма матрицы производных функции φ удовлетворяет неравенству:

$$\left\|\frac{\partial\varphi}{\partial x}(\bar{x})\right\| \le q$$

во всех точках *x* из δ-окрестности $U_\delta(\bar{x})$ точки $\bar{x}^0$.

Следующая теорема дает условие сходимости и оценку скорости сходимости метода простых итераций.

Теорема. Пусть отображение φ является сжатием в $U_\delta(\overline{x^0})$ *и пусть*

$$\left\|\varphi(\bar{x}^0)-\bar{x}^0\right\|<\delta(1-q).$$

Тогда итерационная последовательность:

$$\bar{x}^k=\varphi(\bar{x}^{k-1}),$$

с начальной точкой $\bar{x}^0$*, сходится к решению* $\bar{x}^*$ *системы (1). При этом справедлива следующая оценка погрешности:*

$$\left\|\bar{x}^k-\bar{x}^*\right\|\le\frac{q^k}{1-q}\left\|\varphi(\bar{x}^0)-\bar{x}^0\right\|.$$

Отметим, что начальное приближение $\bar{x}^0$ выбирают экспериментально. (Например, на основе грубого графического решения системы, если порядок системы не высок. По точкам строят график первого уравнения, потом второго и ищут приблизительно точку их пересечения).

Метод Ньютона.

Пусть дана система нелинейных уравнений :

$$\begin{cases} f_1(x_1,\ldots,x_n)=0 \\ f_2(x_1,\ldots,x_n)=0 \\ \ldots\ldots\ldots\ldots\ldots \\ f_n(x_1,\ldots,x_n)=0 \end{cases}$$

Запишем ее в векторной форме:

$$f(\bar{x})=0 \qquad (4.1)$$

Найдем начальное приближение $\bar{x}^0$.

Будем предполагать, что векторная функция f непрерывна дифференцируема в некоторой окрестности начального приближения. Вместо системы (4.1) будем искать решение соответствующей ей линеаризованной системы

$$f(\bar{x}^0)+\frac{\partial f}{\partial x}(\bar{x}^0)(\bar{x}-\bar{x}^0)=0 \Rightarrow \quad f(\bar{x}^0)+J(\bar{x}^0)(\bar{x}-\bar{x}^0)=0,$$

где через $J(\bar{x}^0)$ обозначена для удобства записи матрица производных векторной функции f в точке $\bar{x}^0$ (матрица Якоби системы (4.1) в этой точке).

При этом при применении метода Ньютона предполагается, что $\det J(\bar{x}^0)\neq 0$ в окрестности точки $\bar{x}^0$.

Тогда из линеаризованной системы, которая линейна относительно переменных x, можно найти первое приближение

$$\bar{x}^{k} = \bar{x}^{k-1} - J^{-1}(\bar{x}^{k-1}) f(\bar{x}^{k-1}), k = 1,...$$

Рассматривая линеаризованную систему в точках при $k=1,2,....$, найдем к-ое приближение

$$\bar{x}^{k} = \bar{x}^{k-1} - J^{-1}(\bar{x}^{k-1}) f(\bar{x}^{k-1}), k = 1,...$$

Построенная таким способом рекуррентная последовательность Ньютона сходится при определенных дополнительных условиях к решению системы (4.1). Легко видеть, что рассматриваемый метод совпадает с методом касательных в случае $n=1$, т.е. является многомерным вариантом метода касательных.

На практике обратную матрицу не считают, а на каждом шаге решают линеаризованную систему:

$$f(\bar{x}^{k-1}) + J(\bar{x}^{k-1})(\bar{x} - \bar{x}^{k-1}) = 0 \Rightarrow \bar{x} = \bar{x}^{k}$$

Теорема. При сделанных выше предположениях, последовательность Ньютона сходится к решению системы (4.1), если начальное приближение выбрано достаточно близко к решению.

Отметим в заключение, что метод Ньютона сходится достаточно быстро (скорость сходимости квадратичная), если начальное приближение выбрано удачно. На практике итерационный процесс заканчивают, когда норма разности двух последовательных приближений меньше заданной точности вычисления решения.

Теорема 3.13 (о достаточном условии сходимости метода простых итераций). *Пусть функции* $\varphi_i(x)$ *и* $\varphi_i'(x)$, $i = 1, \ldots, n$, *непрерывны в области* G, *причем выполнено неравенство (где* q *— некоторая постоянная)*

$$\max_{x \in G} \max_{i} \sum_{j=1}^{n} \left| \frac{\partial \varphi_i(x)}{\partial x_j} \right| \leqslant q < 1, \tag{3.28}$$

Если последовательные приближения $x^{(k+1)} = \Phi(x^{(k)})$, $k = 0, 1, 2, \cdots$ *не выходят из области* G, *то процесс последовательных приближений сходится:* $x_* = \lim_{k \to \infty} x^{(k)}$ *и вектор* x_* *является в области* G *единственным решением системы*

функции $\varphi(x)$.

Теорема1. Пусть φ – непрерывно дифференцируемая векторная функция, для которой выполняются следующие условия:

1) $\left\|\varphi(x^0) - x^0\right\| \le \delta(1-q)$;

2) $\left\|\frac{\partial \varphi}{\partial x}(x)\right\| \le q, \qquad (q<1), \qquad \forall x \in U_\delta(x^0)$.

Тогда система (6.7) имеет решение $x^* \in U_\delta(x^0)$, причем единственное, и итерационная последовательность $x^k = \varphi(x^{k-1}), \quad k = 1,2,\ldots,$ с начальным приближением x^0 сходится к решению x^* системы (6.7) . При этом справедлива следующая оценка скорости сходимости:

$$\left\|x^k - x^*\right\| \le \frac{q^n}{1-q}\left\|x^1 - x^0\right\|.$$

Доказательство. Из условия (6.7) вытекает сжимаемость этого отображения. Проверим, что

$\varphi : U_\delta(x^0) \to U_\delta(x^0)$.

Разложив $\varphi(x)$ по формуле Тейлора, получаем для $\forall x \in U_\delta(x^0)$

$$\varphi(x) = \varphi(x^0) + \frac{\partial \varphi}{\partial x}(\xi)(x - x^0),$$

где $\xi \in U_\delta(x^0)$. Тогда

$$\left\|\varphi(x) - x^0\right\| = \left\|\varphi(x^0) + \frac{\partial \varphi}{\partial x}(\xi)(x - x^0) - x^0\right\| \le$$

$$\le \left\|\varphi(x^0) - x^0\right\| + \left\|\frac{\partial \varphi}{\partial x}(\xi)\right\| \cdot \left\|x - x^0\right\| < \delta(1-q) + q\delta = \delta,$$

т. е. $\varphi(x) \in U_\delta(x^0)$.

Следовательно, мы можем применить принцип сжимающих отображений, в силу которого получаем результат теоремы. При этом из принципа сжимающих отображений следует, что

$$\rho(x^*, x^k) \le \frac{\alpha^k}{1-\alpha}\rho(x^0, x^1),$$

что равносильно оценке скорости сходимости

$$\left\|x^k - x^*\right\| \le \frac{q^n}{1-q}\left\|x' - x^0\right\|$$

в нашем случае.

$$f(x^*) + \partial(x^*)(x^* - x^*) = 0.$$

Оценим сходимость метода. Очевидно,

$$f(x^*) = f(x^k) + \frac{\partial f(x^k)}{\partial x}(x^* - x^k) + O(\left\|x^* - x^k\right\|^2),$$

где $O(\left\|x^k\right\|^m)$ означает, что $O(\left\|x^k\right\|^m) \le M\left\|x^k\right\|^m$, $M = const > 0$.

Поскольку $f(x^*) = 0$, получим:

$-J^{-1}(x^k)f(x^k) = x^* - x^k + J^{-1}(x^k)O(\left\|x^* - x^k\right\|^2)$, т. е.

$$\left\|x^{k+1} - x^k\right\| = \left\|J^{-1}(x^k)\cdot O(\left\|x^* - x^k\right\|^2)\right\| \le M\left\|x^* - x^k\right\|^2.$$

Таким образом, метод Ньютона имеет квадратичную сходимость.

Недостатки метода Ньютона: начальное приближение должно быть близким к решению, а матрица Якоби должна быть невырожденной.

Достоинство: быстрая сходимость.

Отметим, что выбор начального приближения является слабым местом итерационных методов.

3. ЗАДАНИЕ

ЗАДАНИЕ. Решить систему нелинейных уравнений:

$$\begin{cases} tg(xy+m) = x \\ ax^2 + 2y^2 = 1, \end{cases}$$ где x>0, y>0,

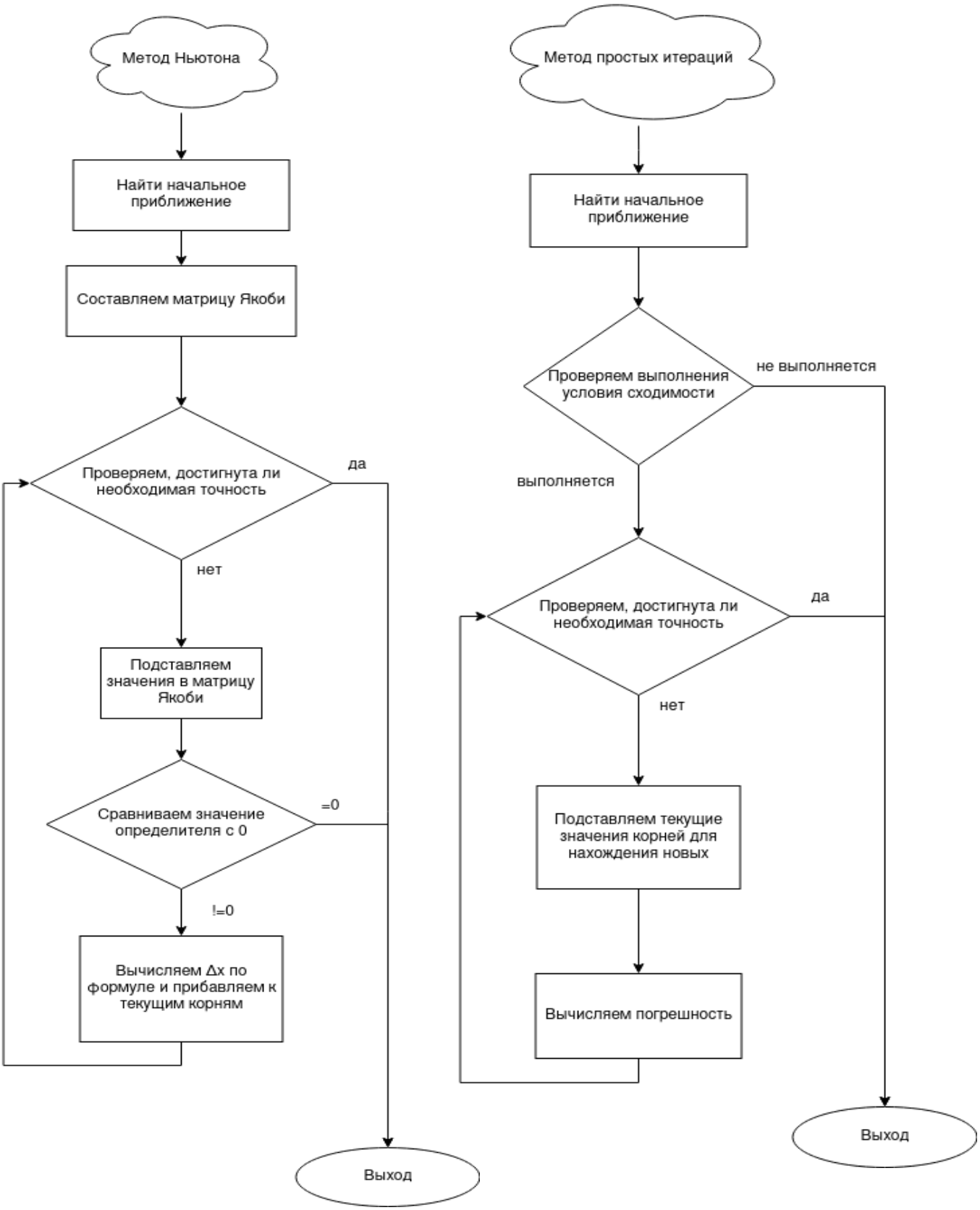
с точностью до 0,0001 методами простых итераций и Ньютона,

принимая для номера варианта k значения параметров a и m из таблицы:

Вариант 5 (k = 5)
m = 0.2
a = 0.9

4. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

Алгоритм программных методов



Метод простых итераций

```
  def iteration_solve(system_equations: np.array, approx,
tol=0.00001, verbose=0):
      n = system_equations.shape[0]
      x = symbols(f'x:{n}')

      fi_equations = system_equations[1]

      prev_roots = np.zeros(shape=(n, ))
      curr_roots = list(approx)

      inaccuracys = np.zeros(shape=(n, ))
      inaccuracy = tol * 10000

      J = get_jacobi(system_equations[0])
      jacobi_values = np.zeros(shape=(n, n))

      roots_d = roots_to_dict(curr_roots, x)

      for i in range(n):
          for j in range(n):
              jacobi_values[i, j] = J[i, j].subs(roots_d)

      # compute q
      if verbose == 1:
          q_1 = float(get_q(fi_equations[0], (approx[0],
approx[1]), 0.1))
          q_2 = float(get_q(fi_equations[1], (approx[1],
approx[0]), 0.1))
          print(f'q_1 = {q_1}')
          print(f'q_2 = {q_2}')
          if q_1>=1 or q_2>=1:
              print("q is greater than 1")
              return None, None

      iteration = 0
      while inaccuracy > tol:
          prev_roots = curr_roots.copy()
          roots_d = roots_to_dict(curr_roots, x)
          for i in range(n):
              try:
                  curr_roots[i] =
float(fi_equations[i].subs(roots_d))
```

```
                except TypeError:
                    print("Complex numbers!")

                inaccuracys[i] = abs(prev_roots[i] -
curr_roots[i])
                roots_d = roots_to_dict(curr_roots, x)

            inaccuracy = np.amax(inaccuracys)
            iteration += 1

        if verbose == 1:
            print('stopped iteration method\n')

        return curr_roots, iteration
```

Метод Ньютона

```
   def newton_solve(system_equations: np.array, approx,
tol=0.00001):
        n = system_equations.shape[0]
        x = symbols(f'x:{n}')

        J = get_jacobi(system_equations)

        error = tol * 10000
        iteration = 0
        roots = approx
        while error > tol:
            iteration += 1

            roots_d = roots_to_dict(roots, x)
            jacobi_values = np.zeros(shape=(n, n))
            for i in range(n):
                for j in range(n):
                    jacobi_values[i, j] = J[i, j].subs(roots_d)

            jacobi_det = np.linalg.det(jacobi_values)
            print(f"Jacobi det = {jacobi_det}")
            if not jacobi_det:
                print("det equal 0. Can't solve system")
                exit(0)

            F = np.zeros(shape=(n, ))
            for i in range(0, n):
```

```
            F[i] = system_equations[i].subs(roots_d)

        delta_x = np.zeros(shape=(n, ), dtype=float)
        delta_x = np.linalg.solve(jacobi_values, -1 * F)

        roots = delta_x + roots
        error = np.amax(abs(delta_x))

    return roots, iteration
```

5. ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

Первый тестовый пример:

$$\begin{cases} x^2 + y^2 = 0 \\ 2xy = 0 \end{cases}$$

Вывод:

```
Initial approximation: (0.6, 0.6)
q_1 = 1.0220940897677138
q_2 = 1.199999999999998

   ValueError: Compression condition is not satisfied!
q > 1
```

Второй тестовый пример:

Я пытался сообразить но не получилось такой пример, чтобы критерий сходимости удовлетворялся и при этом были комплексные корни, но проверка на них присутствует в обоих итеративных методах при подстановке и получении новых приближений вот в таком фрагменте кода:

```
    try:
                    curr_roots[i] =
float(fi_equations[i].subs(roots_d))
                except TypeError:
                    print("Complex numbers!")
```

Исходное условие:

Отсюда за начально приближение берем точку (0,6; 0,6)
Получим:

```
Initial approximation: (0.6, 0.6)
q_1 = 0.8765894087167037
q_2 = 0.504928048014517
stopped iteration method

Simple iteration:
Roots: [0.6190202637188493, 0.57233404659993]
Number of iterations: 9

Initial approximation: (0.6, 0.6)
Jacobi det = -1.2966919501537868
Jacobi det = -1.438284442572687
Jacobi det = -1.4310232132267242
Newton method
Roots: [0.6190172831902083, 0.5723354972447913]
Number of iterations: 3
```

6. ВЫВОДЫ

Таким образом, в ходе выполнения лабораторной работы были изучены методы численного решения систем нелинейных уравнений (метод простой итерации, метод Ньютона), составлена программа численного решения нелинейных уравнений методами простой итерации и Ньютона, проверена правильность работы программы на тестовых примерах, численно решено нелинейное уравнение заданного варианта, сравнено число итераций, необходимого для достижения заданной точности вычисления разными методами.

Как можно заметить, метод Ньютона более практичен, так как в решенных системах уравнений быстрее сходился к корню с заданной точностью, однако он очень сильно зависим от выбора начального приближения. В общем случае, результат всех итерационных методов зависит от выбора начального приближения.