Федеральное государственное бюджетное образовательное
учреждение высшего профессионального образования Московский
государственный технический университет имени Н.Э. Баумана

Лабораторная работа №6
«Решение систем нелинейных уравнений методом Ньютона»
по дисциплине
«Численные методы»

Студент группы ИУ9-61 Александрова О.С.

Преподаватель Домрачева А.Б.

Москва, 2024

Исходные данные:

$$\begin{cases} sin(x+1) - y = 1.2 \\ -x + cos(y) = 2 \end{cases}$$

Задание:

1. Решить систему нелинейных уравнений графически и принять полученное решение за начальное приближение .
2. Решить систему методом Ньютона с точностью $\epsilon = 0.01$.

Теоретические сведения:

Выбрав начальное приближение $X^0 = (x_i^0, ..., x_n^0)$ к решению системы , следующи приближения в методе Ньютона строим по рекуррентной зависимости

$$X^{k+1} = X^k - (f'(X^k))^(- 1)f(x^k), k = 0, 1, 2... \quad (1)$$

Здесь $X^{k+1} = (x_1^{k+1}, ..., x_n^{k+1})^T$ и $X^k = (x_1^k, ..., x_n^k)^T$ - столбцы $(k+1)$-го порядка и k-го приближения к решению; $f(X^k)^n = f_1(X^k), ..., f_n(X^k))^T$ - значение столбца левой части системы в точке X^k

$$f'(X^k) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

Приближение метода Ньютона удобно искать в два этапа. Вначал решаем систему линейных уравнений с матриц $f'(X^k)$ - матрицей Якоби вектор-функции f:

$$f'(X^k)y = -f(X^k) \quad (2)$$

Затем (k+1)-е приближение $\mathrm{X}^{k+1}k - \mathrm{y} = (\mathrm{y}_1, ..., y_n)$

$$X^{k+1} = X^k + y \tag{3}$$

Для решения системы нелинейных уравнений с заданной точностью ϵ необходимо сравнить ϵ с погрешностью k-го приближения

$$||X^k - Xk - 1|| = max|x_i^k - x_i^{k-i}| \tag{4}$$

Метод Ньютона сходится, если все функции дважды непрерывно дифференцируемы по всем переменным и начальное приближение находится достаточно близко к точному решению.

Практическая реализация:

Листинг 1: Решение систем нелинейных уравнений методом Ньютона

```
import math


def newton_method(x0, y0, eps):
    x = x0
    y = y0

    def f1(x, y):
        return math.sin(x + 1) - y - 1.2

    def f2(x, y):
        return -x + math.cos(y) - 2


    def df1_dx(x, y):
        return math.cos(x + 1)

    def df1_dy(x, y):
        return -1

    def df2_dx(x, y):
        return -1

    def df2_dy(x, y):
```

```
25          return -math.sin(y)
26      k = 0
27      while True:
28          f1_val = f1(x, y)
29          f2_val = f2(x, y)
30          df1_dx_val = df1_dx(x, y)
31          df1_dy_val = df1_dy(x, y)
32          df2_dx_val = df2_dx(x, y)
33          df2_dy_val = df2_dy(x, y)
34
35          det = df1_dx_val * df2_dy_val - df1_dy_val * df2_dx_val
36          dx = (-f1_val * df2_dy_val + f2_val * df1_dy_val) / det
37          dy = (-f2_val * df1_dx_val + f1_val * df2_dx_val) / det
38          k += 1
39          x += dx
40          y += dy
41          if abs(dx) < eps and abs(dy) < eps:
42              break
43      print(k)
44
45      return x, y
46
47
48  def main():
49      x0 = -3
50      y0 = -2
51      eps = 0.01
52      solution = newton_method(x0, y0, eps)
53      print("x =", solution[0])
54      print("y =", solution[1])
55
56
57  if __name__ == "__main__":
58      main()
```

Результаты:

Ниже приведен вывод программы:

$$x = -2.5883789193952675$$
$$y = -2.1998478971303252$$

Выводы:

В ходе выполнения лабораторной работы был рассмотрен «Метод Ньютона» для поиска решения системы нелинейных уравнений. Была написана реализация данного метода на языке программирования Python, для которой были вручную посчитаны производные для уравнений.