МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по практической работе №4

по дисциплине «Вычислительная математика»

Тема: Решение нелинейных уравнений и систем

Студент гр. 9304 Афанасьев А.

Преподаватель Попова Е. В.

Санкт-Петербург
2020

Цель работы.

Формирование практических навыков нахождения корней алгебраических и трансцендентных уравнений, а также системы нелинейных уравнений методами Ньютона и простых итераций.

Основные теоретические положения.

Решение нелинейных уравнений

Численное решение нелинейных (алгебраических или трансцендентных) уравнений вида f(x)=0 заключается в нахождении значений x, удовлетворяющих (с заданной точностью) данному уравнению и состоит из следующих основных этапов:

1. *Отделение* (изоляция, локализация) корней уравнения.
2. *Уточнение* с помощью некоторого вычислительного алгоритма конкретного выделенного корня с заданной точностью.

Целью первого этапа является нахождение отрезков из области определения функции f(x), внутри которых содержится только один корень решаемого уравнения. Иногда ограничиваются рассмотрением лишь какой-нибудь части области определения, вызывающей по тем или иным соображениям интерес. Для реализации данного этапа используются *графические* или *аналитические* способы.

При аналитическом способе отделения корней полезна следующая теорема.

Теорема. *Непрерывная строго монотонная функция* f(x) *имеет и притом единственный нуль на отрезке* [a,b] *тогда и только тогда, когда на его концах она принимает значения разных знаков.*

Достаточным признаком монотонности функции f(x) на отрезке [a,b] является сохранение знака производной функции.

Графический способ отделения корней целесообразно использовать в том

случае, когда имеется возможность построения графика функции y=f(x). Наличие графика исходной функции дает непосредственное представление о количестве и расположении нулей функции, что позволяет определить промежутки, внутри которых содержится только один корень. Если построение графика функции y=f(x) вызывает затруднение, часто оказывается удобным преобразовать уравнение f(x)=0 к эквивалентному виду $f_1(x)=f_2(x)$ и построить графики функций $y=f_1(x)$ и $y=f_2(x)$. Абсциссы точек пересечения этих графиков будут соответствовать значениям корней решаемого уравнения.

Так или иначе, при завершении первого этапа, должны быть определены промежутки, на каждом из которых содержится только один корень уравнения.

Для уточнения корня с требуемой точностью применяются различные итерационные методы, заключающиеся в построении числовой последовательности $x^{(k)}$ (k=0, 1, 2, ...), сходящейся к искомому корню $x^{(*)}$ уравнения f(x)=0.

Решение систем нелинейных уравнений

Систему нелинейных уравнений с n неизвестными можно записать в виде

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0 \\ f_2(x_1, x_2, \ldots, x_n) = 0 \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ f_n(x_1, x_2, \ldots, x_n) = 0 \end{cases}$$

или, более коротко, в векторной форме

$$F(X) = 0,$$

где $X=(x_1, x_2, ..., x_n)^T$ – вектор неизвестных, $F = (f_1, f_2, ..., f_n)^T$ – вектор-функция.

В редких случаях для решения такой системы удается применить метод последовательного исключения неизвестных и свести решение исходной задачи к решению одного нелинейного уравнения с одним неизвестным. Значения других неизвестных величин находятся соответствующей подстановкой в

конкретные выражения. Однако в подавляющем большинстве случаев для решения систем нелинейных уравнений используются итерационные методы. В дальнейшем предполагается, что ищется изолированное решение нелинейной системы.

Как и в случае одного нелинейного уравнения, локализация решения может осуществляться на основе специфической информации по конкретной решаемой задаче (например, по физическим соображениям), и – с помощью методов математического анализа. При решении системы двух уравнений, достаточно часто удобным является графический способ, когда месторасположение корней определяется как точки пересечения кривых $f_1(x_1, x_2) = 0$, $f_2(x_1, x_2) = 0$ на плоскости (x_1,x_2).

Постановка задачи.

Численно решить уравнение и систему уравнений методами Ньютона и простых итераций с заданной точностью ε (ε = 0.01, 0.0001, 0.000001).

Уравнение: $7.3\cos(2.9x + 6.7) = 6.2x^2 - 5.8x - 13.1$

Система уравнений: $\begin{cases} 3x^6 + 5y^6 = 677 + 509x - 745y \\ 7x - 6e^{-0.6y} = 3y + 9e^{-0.5x} - 15 \end{cases}$

Выполнение работы.

Решение нелинейного уравнения методом Ньютона.

Для нахождения корней уравнения $7.3\cos(2.9x + 6.7) - 6.2x^2+5.8x+13.1 = 0$ была реализована функция newton(eps, a, b) с аргументами eps (задаваемая точность), a (левый конец отрезка с корнем), b (правый конец отрезка с корнем) и x0 (начальное приближение). Пример вызова функции представлен на рисунке 1, а ее исходный код представлен в приложении А.

```
newton(0.01, 0, 1, 0.5)
```

Рисунок 1 - Пример вызова newton()

Рассчитаем первые две производные:

df(x) = 5.8 - 12.4x - 21.17sin(6.7 + 2.9x),

ddf(x) = -12.4 - 61.393cos(6.7 + 2.9x).

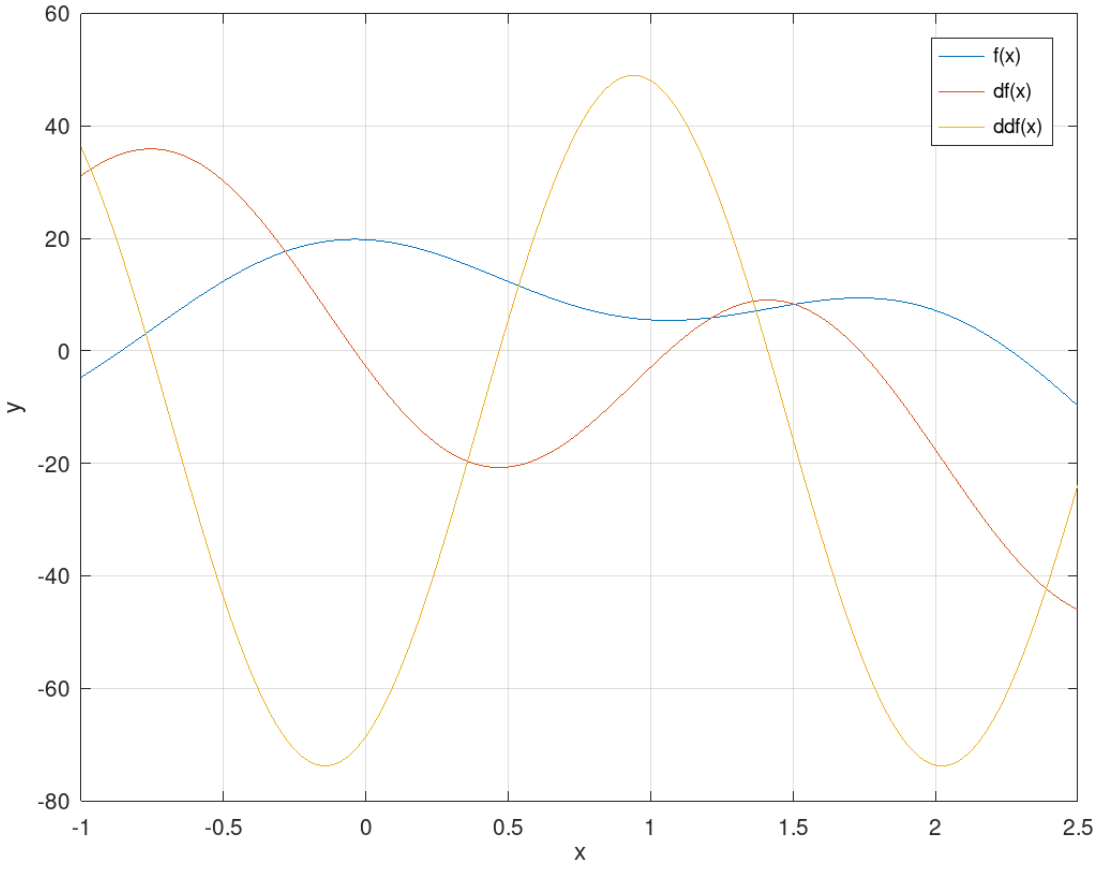

Рисунок 2 - Графики f(x), df(x) и ddf(x)

Графическим методом были найдены отрезки, на которых находятся по одному корню,: [-1; -0.5], [2; 2.5], на рисунке 2 представлены графики f(x), df(x) и ddf(x). Результаты решения нелинейного уравнения методом Ньютона представлены в таблице 1. А графики зависимости k от eps представлены на рисунке 3.

Выберем начальное приближение для обоих корней, опираясь на условие $f(x^0) * ddf(x^0) > 0$.

Для первого корня: $x^{(0)}_1 = -0.8$, так как $f(x^{(0)}_1) * ddf(x^{(0)}_1) = 310.95$.

Для второго корня: $x^{(0)}_2 = 2.5$, так как $f(x^{(0)}_2) * ddf(x^{(0)}_2) = 233.27$.

Таблица 1. Результаты метода Ньютона для первого корня при eps = 0.01

k	$x^{(k)}$	$f(x^{(k)})$	$df(x^{(k)})$	$-f(x^{(k)}) / df(x^{(k)})$
0	-0.8	2.109994	35.731271	-0.059052
1	-0.859052	0.018990	34.993607	-0.000543

Таблица 2. Результаты метода Ньютона для первого корня при eps = 0.0001

k	$x^{(k)}$	$f(x^{(k)})$	$df(x^{(k)})$	$-f(x^{(k)}) / df(x^{(k)})$
0	-0.8	2.109994	35.731271	-0.059052
1	-0.859052	0.018990	34.993607	-0.000543
2	-0.859594	0.000003	34.984234	-0.00000009

Таблица 3. Результаты метода Ньютона для первого корня при eps = 0.000001

k	$x^{(k)}$	$f(x^{(k)})$	$df(x^{(k)})$	$-f(x^{(k)}) / df(x^{(k)})$
0	-0.8	2.109994	35.731271	-0.059052
1	-0.859052	0.018990	34.993607	-0.000543
2	-0.859594	0.000003	34.984234	-0.00000009

Таблица 4. Результаты метода Ньютона для второго корня при eps = 0.01

k	$x^{(k)}$	$f(x^{(k)})$	$df(x^{(k)})$	$-f(x^{(k)}) / df(x^{(k)})$
0	2.500000	-9.791645	-46.000273	-0.212861
1	2.287139	-0.807489	-37.241892	-0.021682
2	2.265457	-0.013518	-35.985619	-0.000376

Таблица 5. Результаты метода Ньютона для второго корня при eps = 0.0001

k	$x^{(k)}$	$f(x^{(k)})$	$df(x^{(k)})$	$-f(x^{(k)}) / df(x^{(k)})$
0	2.500000	-9.791645	-46.000273	-0.212861
1	2.287139	-0.807489	-37.241892	-0.021682
2	2.265457	-0.013518	-35.985619	-0.000376
3	2.265081	-0.000004	-35.963366	-0.00000011

Таблица 6. Результаты метода Ньютона для второго корня при eps = 0.000001

k	$x^{(k)}$	$f(x^{(k)})$	$df(x^{(k)})$	$-f(x^{(k)}) / df(x^{(k)})$
0	2.500000	-9.791645	-46.000273	-0.212861
1	2.287139	-0.807489	-37.241892	-0.021682
2	2.265457	-0.013518	-35.985619	-0.000376
3	2.265081	-0.000004	-35.963366	-0.00000011

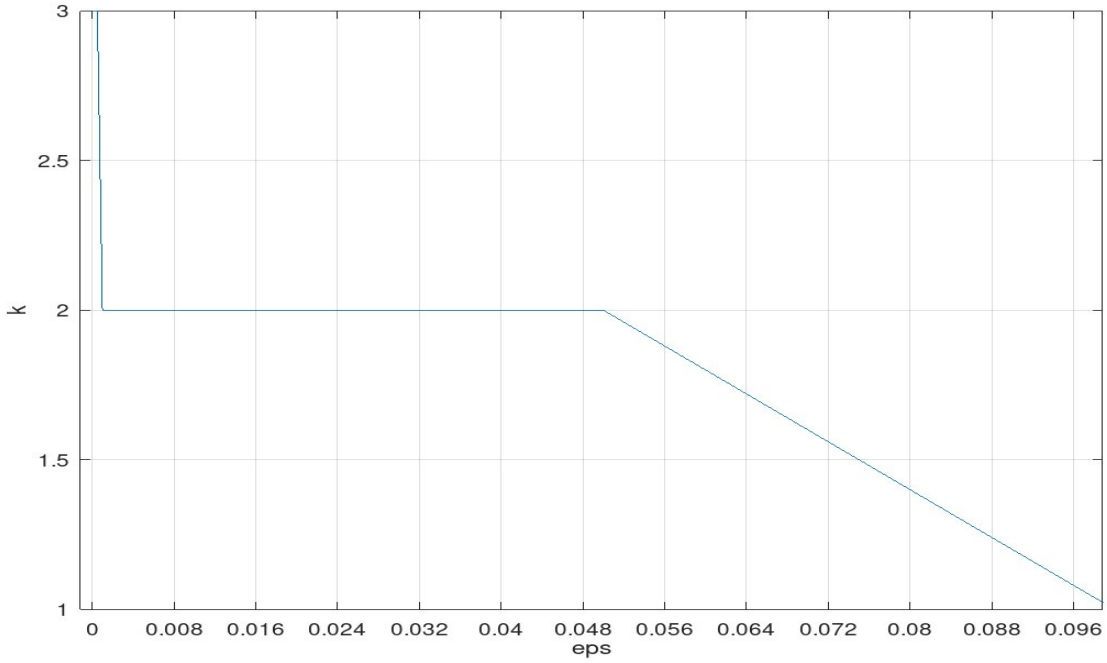

Рисунок 3 - Зависимость k от eps при поиске второго корня

Для построения графиков на рисунках 3 и 4 были использованы дополнительные значения eps, например, при eps = 0.1, поиск второго корня закончился за 1 шаг. Анализируя рисунки 3 и 4 и таблицы 1 - 6, можно прийти к выводам, что точность значения корня и количество шагов зависят от значения точности eps: чем он меньше, тем точнее результат и тем больше шагов.

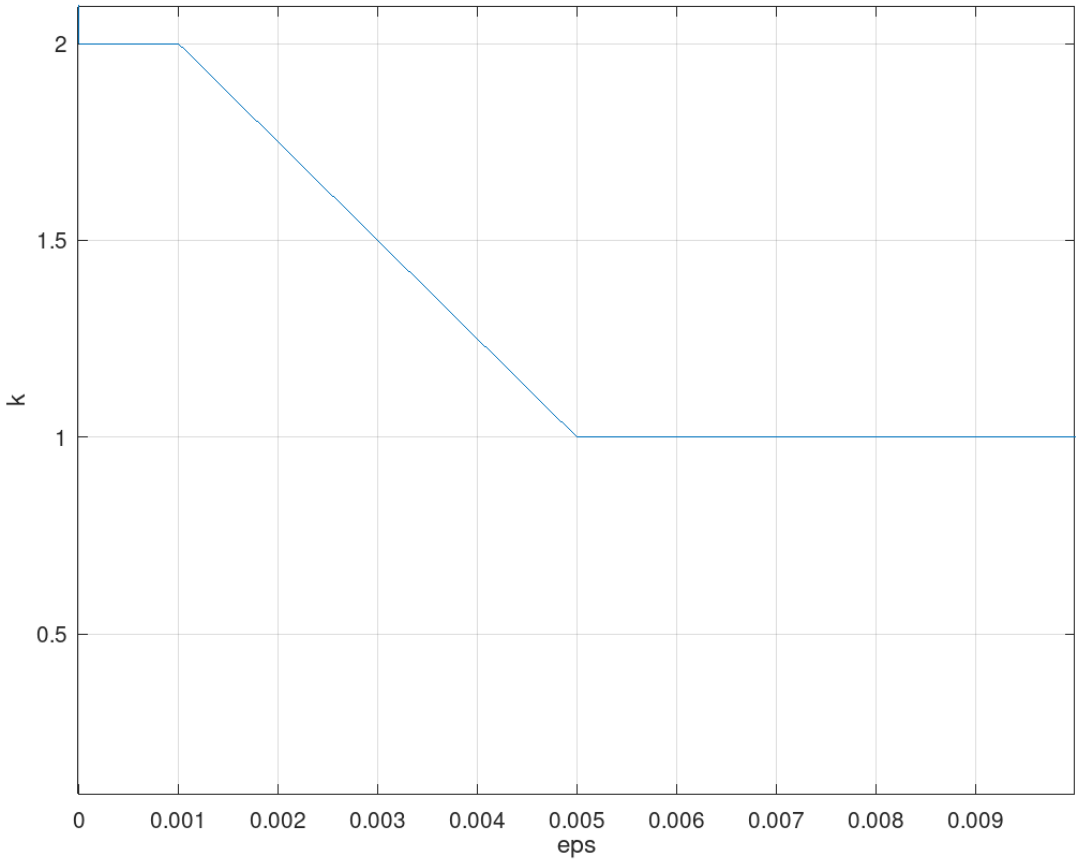

Рисунок 4 - Зависимость k от eps при поиске первого корня

Решение нелинейного уравнения методом простых итераций.

Для вычисления корня уравнения была реализована функция siter(), аргументами которой являются a и b (концы отрезка, на котором лежит 1 корень), eps (задаваемая точность). Исходный код представлен в приложении Б, а пример вызова представлен на рисунке 5.

```
siter(0.01, -1, -0.5)
```

Рисунок 5 - Пример вызова siter()

Начальным приближением $x^{(0)}$ является $\frac{a+b}{2}$. Для критерия выхода на [a, b] будут вычислены максимум M минимум m df(x), с помощью которых будет вычислено q = $\frac{M-m}{M+m}$. Также будет вычислено lambda = $\frac{2}{m+M}$
Выберем a = -1, b = -0.5 для первого корня и a = 2, b = 2.7 для второго.

Таблица 7. Результаты метода простых итераций для 1 корня при eps = 0.01

k	$x^{(k)}$	$\varphi(x^{(k)})$
0	-0.75	-0.868106
1	-0.868106	-0.859114
2	-0.859114	-0.859623

Таблица 8. Результаты метода простых итераций для 1 корня при eps = 0.0001

k	$x^{(k)}$	$\varphi(x^{(k)})$
0	-0.75	-0.868106
1	-0.868106	-0.859114
2	-0.859114	-0.859623
3	-0.859623	-0.859593

Таблица 9. Результаты метода простых итераций для 1 корня при eps = 0.000001

k	$x^{(k)}$	$\varphi(x^{(k)})$
0	-0.75	-0.868106
1	-0.868106	-0.859114
2	-0.859114	-0.859623
3	-0.859623	-0.859593
4	-0.859593	-0.859595
5	-0.859595	-0.859594

Таблица 10. Результаты метода простых итераций для 2 корня, eps = 0.01

k	$x^{(k)}$	$\varphi(x^{(k)})$
0	2.35	2.250150
1	2.250150	2.266417
2	2.266417	2.264942

Таблица 11. Результаты метода простых итераций для 2 корня, eps = 0.0001

k	$x^{(k)}$	$\varphi(x^{(k)})$
0	2.35	2.250150
1	2.250150	2.266417
2	2.266417	2.264942
3	2.264942	2.265096
4	2.265096	2.265080

Таблица 12. Результаты метода простых итераций для 2 корня, eps = 0.000001

k	$x^{(k)}$	$\varphi(x^{(k)})$
0	2.35	2.250150
1	2.250150	2.266417
2	2.266417	2.264942
3	2.264942	2.265096
4	2.265096	2.265080
5	2.265080	2.265082
6	2.265082	2.265081

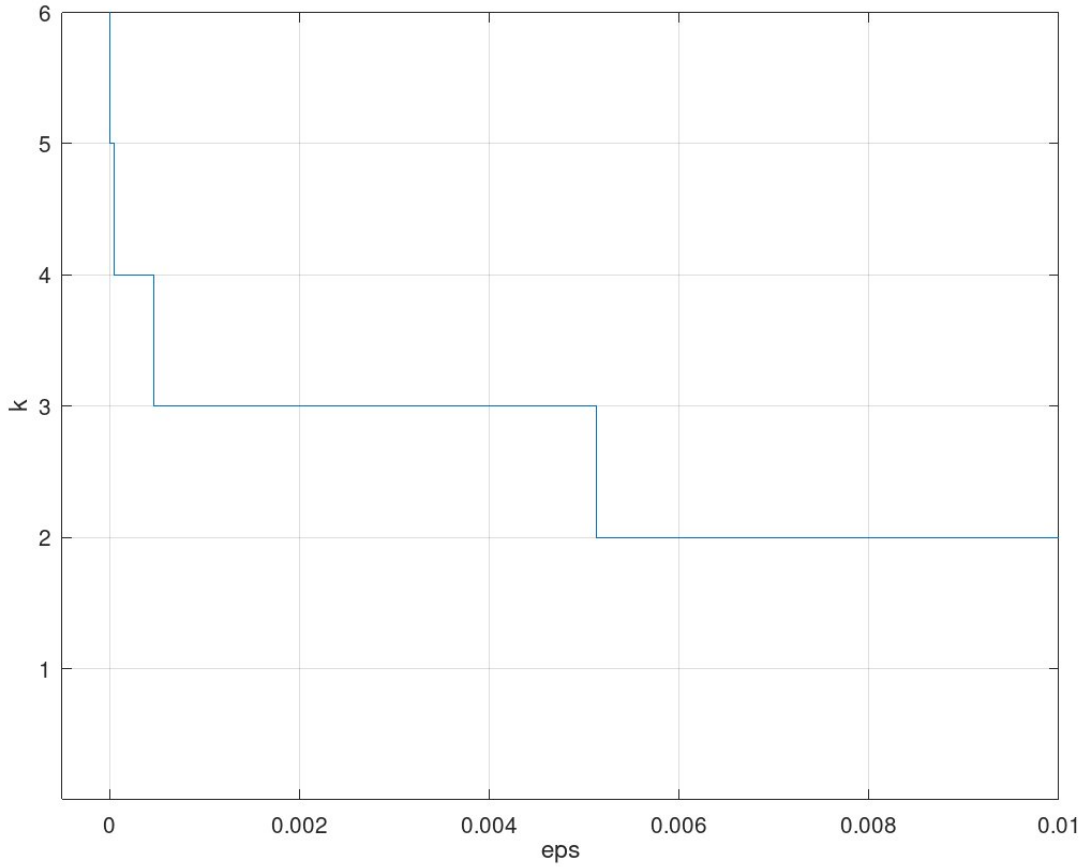

Рисунок 6 - Зависимость k от eps при поиске второго корня

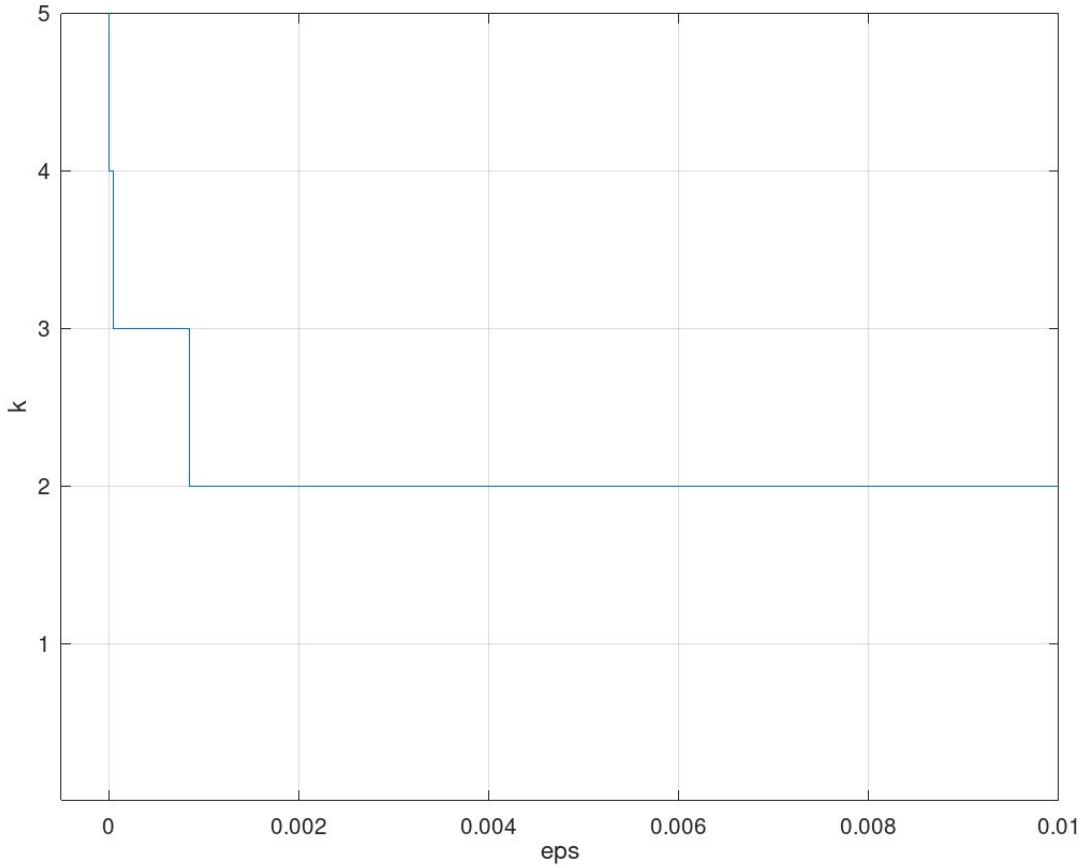

Рисунок 7 - Зависимость k от eps при поиске первого корня

Проанализировав данные таблиц 7 - 12 и графики на рисунках 6 и 7, можно сделать вывод, что чем меньше значение eps, тем больше число итераций k и тем точнее вычисляется корень.

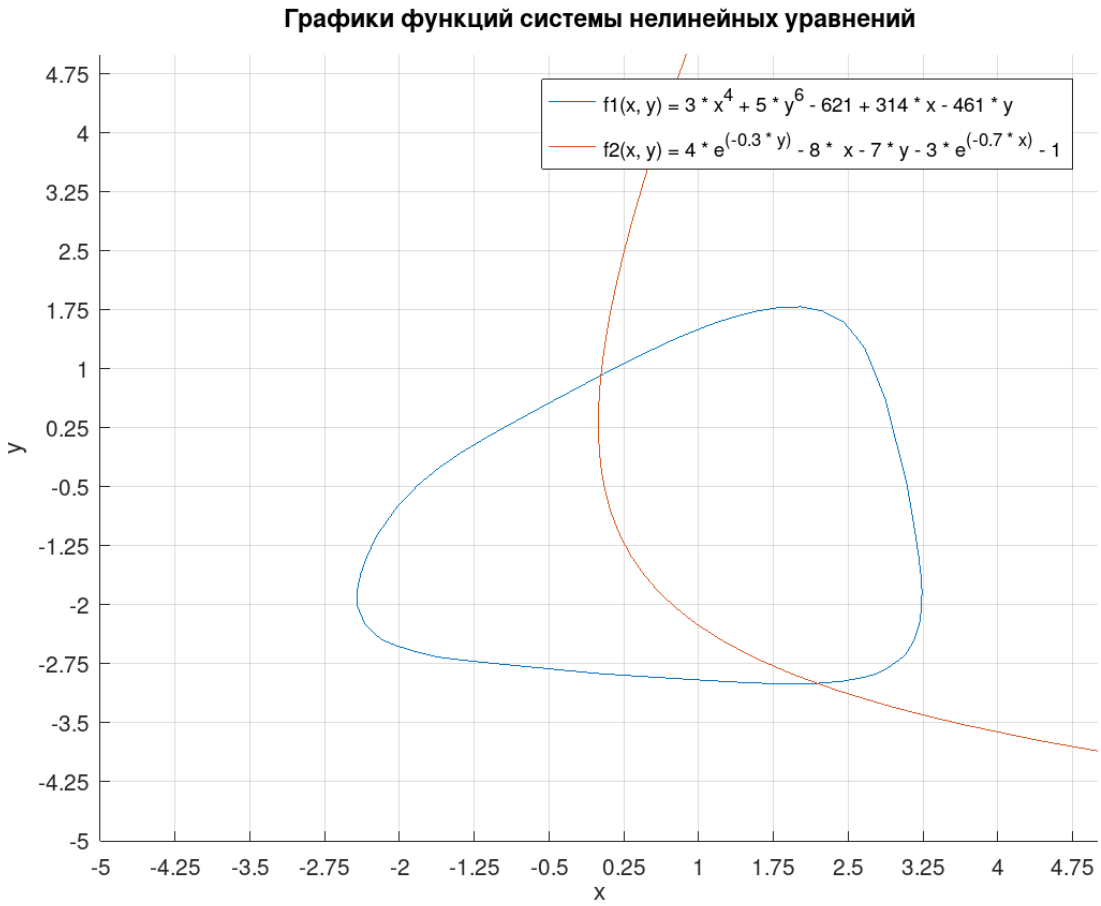

Рисунок 8 — Графическое изображение системы уравнений

Графическим методом были найдены приблизительные корни системы: [2.203; -3.007] и [0.019; 0.917]. Графическое изображение системы представлено на рисунке 8.

Система нелинейных уравнений:

$$\begin{cases} 3x^6 + 5y^6 = 677 + 509x - 745y \\ 7x - 6e^{-0.6y} = 3y + 9e^{-0.5x} - 15 \end{cases}$$

Решение системы нелинейных уравнений методом Ньютона.

Была реализована функция newtons(), исходный код которой представлен в приложении В. Она с помощью метода Ньютона вычисляет корень заданной системы. Пример вызова функции представлен на рисунке 9. Она принимает вектор-столбец z0 (начальное приближение) и eps (задаваемая точность).

Матрица Якоби для заданной системы:

$$J=\begin{pmatrix} \frac{\partial f_1(x,y)}{\partial x} & \frac{\partial f_1(x,y)}{\partial y} \\ \frac{\partial f_2(x,y)}{\partial x} & \frac{\partial f_2(x,y)}{\partial y} \end{pmatrix}=\begin{pmatrix} -509+18x^5 & 745+30y^5 \\ 7+4.5e^{(-0.5x)} & -3+3.6e^{(-0.6y)} \end{pmatrix}$$

Для первого корня начальное приближение $x^{(0)}_1$ = [2.1; -3], для второго $x^{(0)}_2$ = [0.1; 1.1]. Результаты метода Ньютона представлены в таблицах 13 - 18.

```
newtons([2.1; -3], 0.01)
```

Рисунок 9 - Пример вызова newtons()

Таблица 13. Результаты метода Ньютона для 1 корня системы, eps = 0.01

k	$\vec{x}^{(k)}$	$f_1(\vec{x}^{(k)})$	$f_2(\vec{x}^{(k)})$	$-J^{-1}(\vec{x}^{(k)}) * F(\vec{x}^{(k)})$
0	[2.1; -3]	-78.601637	-0.747325	[0.105474; -0.008365]
1	[2.205474;-3.008365]	10.839773	-0.004762	[-0.002680; 0.001457]
2	[2.202794; -3.006908]	0.020671616	-0.00001662	[-0.0000043; 0.0000028]

Таблица 14. Результаты метода Ньютона для 1 корня системы, eps = 0.0001

k	$\vec{x}^{(k)}$	$f_1(\vec{x}^{(k)})$	$f_2(\vec{x}^{(k)})$	$-J^{-1}(\vec{x}^{(k)}) * F(\vec{x}^{(k)})$
0	[2.1; -3]	-78.601637	-0.747325	[0.105474; -0.008365]
1	[2.205474;-3.008365]	10.839773	-0.004762	[-0.002680; 0.001457]
2	[2.202794; -3.006908]	0.020671616	-0.00001662	[-0.000004; 0.000003]
3	[2.202790; 3.006905]	$6.9485 * 10^{-8}$	$-5.997 * 10^{-11}$	$[-1.42 * 10^{-11}; 9.57 * 10^{-12}]$

Таблица 15. Результаты метода Ньютона для 1 корня системы, eps = 0.00001

k	$\vec{x}^{(k)}$	$f_1(\vec{x}^{(k)})$	$f_2(\vec{x}^{(k)})$	$-J^{-1}(\vec{x}^{(k)}) * F(\vec{x}^{(k)})$
0	[2.1; -3]	-78.601637	-0.747325	[0.105474; -0.008365]
1	[2.205474;-3.008365]	10.839773	-0.004762	[-0.002680; 0.001457]
2	[2.202794; -3.006908]	0.020671616	-0.00001662	[-0.000004; 0.000003]
3	[2.202790; 3.006905]	$6.9485 * 10^{-8}$	$-5.997 * 10^{-11}$	$[-1.42 * 10^{-11}; 9.57 * 10^{-12}]$
4	[2.2028; -3.0069]	$-9.0949 * 10^{-13}$	$5.3291 * 10^{-15}$	$[-2.82 * 10^{-16}; -1.55 * 10^{-16}]$

Таблица 16. Результаты метода Ньютона для 2 корня системы, eps = 0.01

k	$\vec{x}^{(k)}$	$f_1(\vec{x}^{(k)})$	$f_2(\vec{x}^{(k)})$	$-J^{-1}(\vec{x}^{(k)}) * F(\vec{x}^{(k)})$
0	[0.1; 1.2]	181.029923	0.618422	[-0.085096; -0.273707]
1	[0.014904; 0.926293]	8.660542	-0.049490	[0.0003588; -0.008928]
2	[0.018492; 0.917365]	0.004345256	-0.00006384	$[5.403 * 10^{-6}; -2.086 * 10^{-6}]$

Таблица 17. Результаты метода Ньютона для 2 корня системы, eps = 0.0001

k	$\vec{x}^{(k)}$	$f_1(\vec{x}^{(k)})$	$f_2(\vec{x}^{(k)})$	$-J^{-1}(\vec{x}^{(k)}) * F(\vec{x}^{(k)})$
0	[0.1; 1.2]	181.029923	0.618422	[-0.085096; -0.273707]
1	[0.014904; 0.926293]	8.660542	-0.049490	[0.0003588; -0.008928]
2	[0.018492; 0.917365]	0.004345256	-0.00006384	$[5.403 * 10^{-6}; -2.086 * 10^{-6}]$
3	[0.018498; 0.917363]	$2.3135 * 10^{-10}$	$-3.525 * 10^{-11}$	$[3.225 * 10^{-12}; 1.845 * 10^{-12}]$

Таблица 18. Результаты метода Ньютона для 2 корня системы, eps = 0.000001

k	$\vec{x}^{(k)}$	$f_1(\vec{x}^{(k)})$	$f_2(\vec{x}^{(k)})$	$-J^{-1}(\vec{x}^{(k)}) * F(\vec{x}^{(k)})$
0	[0.1; 1.2]	181.029923	0.618422	[-0.085096; -0.273707]
1	[0.014904; 0.926293]	8.660542	-0.049490	[0.0003588; -0.008928]
2	[0.018492; 0.917365]	0.004345256	-0.00006384	$[5.403*10^{-6}; -2.086*10^{-6}]$
3	[0.018498; 0.917363]	$2.3135 *10^{-10}$	$-3.525 * 10^{-11}$	$[3.225*10^{-12}; 1.845*10^{-12}]$
4	[0.018498; 0.917363]	$1.1021*10^{-12}$	-1.7764e-15	$[1.638*10^{-16}; 1.091*10^{-16}]$

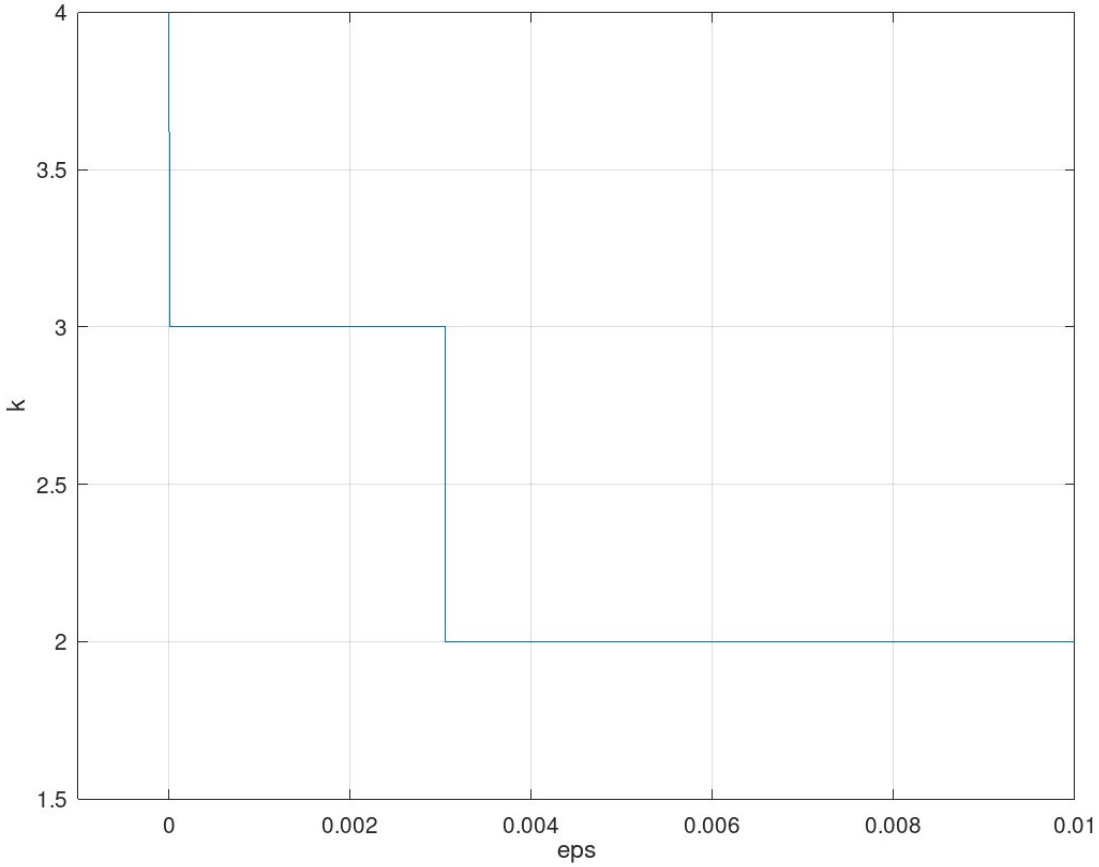

Рисунок 10 - Зависимость k от eps при поиске 1 корня

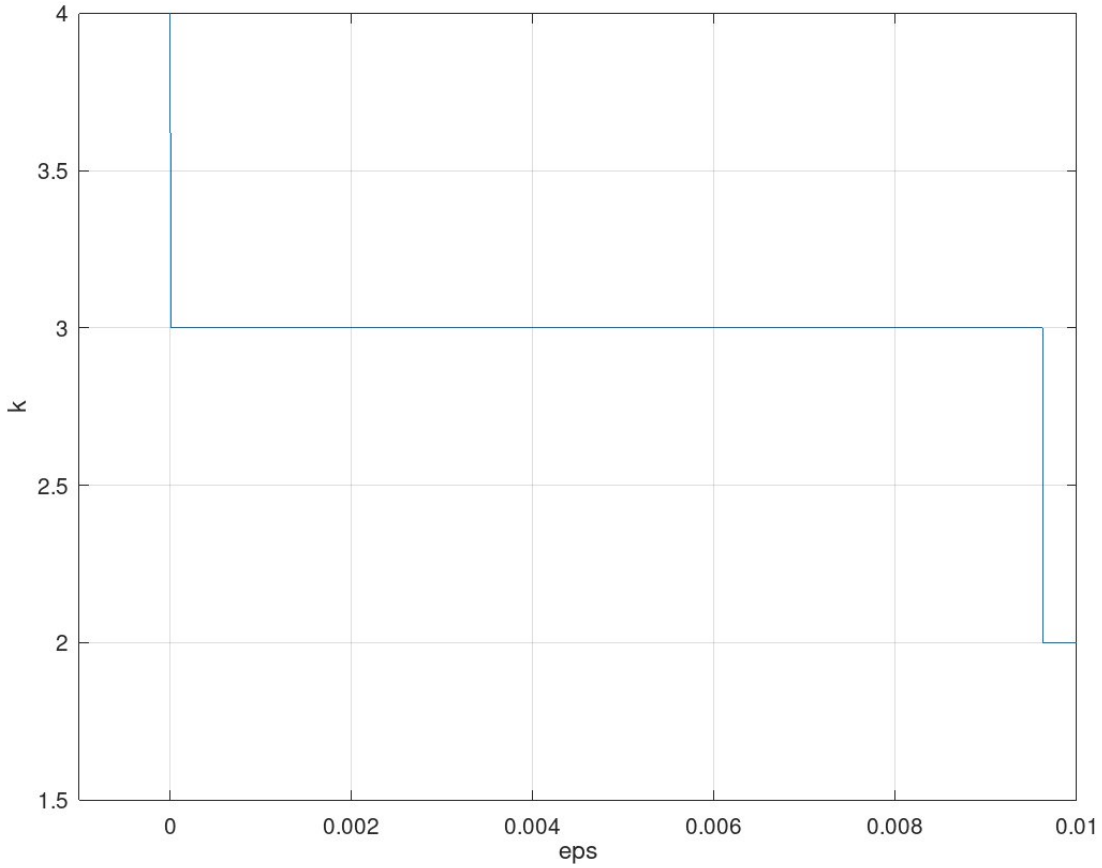

Рисунок 11 - Зависимость k от eps при поиске 2 корня

Анализируя рисунки 10 и 11 и таблицы 13 - 18, можно прийти к выводам, что точность значения корня и количество шагов зависят от значения точности eps: чем он меньше, тем точнее результат и тем больше шагов.

Решение системы нелинейных уравнений методом простых итераций.

Была реализована функция siters() (исходный код см. в приложение Г), вычисляющая корень заданной системы нелинейных уравнений. Функция принимает x0 (начальное приближение) и eps (задаваемая точность). Пример вызова siters() представлен на рисунке 12.

```
siters([1; 2], 0.01)
```

Рисунок 12 - Пример вызова siters()

Графическим методом установили начальное приближение для первого корня [2.1; -3], для второго [0.1; 1.2]. Функция не выдает значений корней. Проверим условие сходимости итерационной последовательности. Чтобы ряд сходился, должно выполняться следующее:

$$max\|\varphi'(x)\|<1, x\in G,$$

где G — это область: первого корня $|x-2.1|\le 0.25, |y+3|\le 0.25$ или второго $|x-0.1|\le 0.25, |y-1.2|\le 0.25$.

Чтобы проверить условие сходимости, найдем:

$$max\|\varphi'(x)\|=max\left\{\left|\frac{\partial\varphi_1(x,y)}{\partial x}\right|+\left|\frac{\partial\varphi_1(x,y)}{\partial y}\right|, \left|\frac{\partial\varphi_2(x,y)}{\partial x}\right|+\left|\frac{\partial\varphi_2(x,y)}{\partial y}\right|\right\}, x\in G$$

Нашли φ_1 и φ_2:

$$\varphi_1(x)=(3x^6+5y^6-677+745y)/509$$

$$\varphi_2(x)=(7x-6e^{(-0.6y)}-9e^{(-0.5x)}+15)/3$$

Для первого корня:

$$\left|\frac{\partial\varphi_1(x,y)}{\partial x}\right|+\left|\frac{\partial\varphi_1(x,y)}{\partial y}\right|=\left|\frac{18x^5}{509}\right|+\left|\frac{5(6y^5+149)}{509}\right|\ge 14>1$$

Для второго корня:

$$\left|\frac{\partial\varphi_1(x,y)}{\partial x}\right|+\left|\frac{\partial\varphi_1(x,y)}{\partial y}\right|=\left|\frac{18x^5}{509}\right|+\left|\frac{5(6y^5+149)}{509}\right|\ge 1.6>1$$

Полученных только что значений достаточно, чтобы сделать вывод, что условие сходимости нарушено. Это значит, что использовать метод простых итераций на этой системе нельзя.

Выводы.

Были сформированы практические навыки нахождения корней алгебраических и трансцендентных уравнений, а также системы нелинейных уравнений методами Ньютона и простых итераций. Были реализованы функции на языке программирования GNU Octave для решения нелинейных уравнений и

системы нелинейных уравнений с помощью методов Ньютона и простых итераций.

Были построены зависимости числа итераций k от точности eps, по которым можно сделать вывод, что чем меньше eps, тем больше k, что увеличивает точность результата.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ФУНКЦИИ NEWTON

```
function r = newton(eps, a, b, x0)
    dx = a : 0.001 : b;
    m = min(abs(df(dx)));
    M = max(abs(ddf(dx)));
    eps = sqrt(2 * eps * m / M);
    x = next(x0);
    k = 1;
    while (abs(x - x0) >= eps)
        x0 = x;
        x = next(x0);
        k++;
    end
    r = [x, k];
end

function y = f(x)
    y = 7.3*cos(2.9*x + 6.7) - 6.2 * x .^ 2 + 5.8 * x + 13.1;
end

function y = df(x)
    y = 5.8 - 12.4 * x - 21.17 * sin(6.7 + 2.9 * x);
end

function y = ddf(x)
    y = -12.4 - 61.393 * cos(6.7 + 2.9 * x);
end

function x = next(x0)
    x = x0 - f(x0) / df(x0);
end
```

ПРИЛОЖЕНИЕ Б

ИСХОДНЫЙ КОД ФУНКЦИИ SITER

```
function r = siter(eps, a, b)
    dx = a : 0.0001 : b;
    m = min(df(dx));
    M = max(df(dx));
    lambda = 2 / (m + M);
    q = (M - m) / (M + m);
    x0 = (a + b) / 2;
    x1 = phi(x0, lambda);
    k = 1;
    while (abs(x1 - x0) >= eps * abs((-1 + q**(-1))))
        x0 = x1;
        x1 = phi(x0, lambda);
        ++k;
    end
    r = [x1, k];
end

function y = f(x)
    y = 7.3*cos(2.9*x + 6.7) - 6.2 * x .^ 2 + 5.8 * x + 13.1;
end

function y = df(x)
    y = 5.8 - 12.4 * x - 21.17 * sin(6.7 + 2.9 * x);
end

function y = phi(x, lambda)
    y = x - lambda * f(x);
end
```

ПРИЛОЖЕНИЕ В

ИСХОДНЫЙ КОД ФУНКЦИИ NEWTONS

```
function res = newtons(z0, eps)
    z = z0 - j(z0)^(-1) * fs(z0);
    k = 1;
    while(norm(z - z0) > eps)
        z0 = z;
        z = z0 - j(z0)^(-1) * fs(z0);
        ++k;
    end
    res = z;
end

function J = j(z)
    x = z(1);
    y = z(2);
    J = [0, 0; 0, 0];
    J(1,1) = -509 + 18 * x^5;
    J(1,2) = 745 + 30 * y^5;
    J(2,1) = 7 + 4.5 * e^(-0.5 * x);
    J(2,2) = -3 + 3.6 * e^(-0.6 * y);
end

function z = fs(v)
    x = v(1);
    y = v(2);
    z1 = 3*x.^6 + 5*y.^6 - 677-509*x+745*y;
    z2 = 7*x - 6*e.^(-0.6*y)- 3*y - 9*e.^(-0.5*x)+15;
    z = [z1; z2];
end
```

ПРИЛОЖЕНИЕ Г

ИСХОДНЫЙ КОД ФУНКЦИИ SITERS

```
function res = siters(x0, eps)
    res = [];
    x = phis(x0);
    k = 1;
    while(norm(x - x0) > eps)
        x0 = x;
        x = phis(x0);
        ++k;
    end
    res = [x; k];
end

function f = phis(X)
    f = [];
    x = X(1);
    y = X(2);
    f1 = (3*x.^6 + 5*y.^6 - 677+745*y) / 509;
    f2 = (7*x - 6*e.^(-0.6*y) - 9*e.^(-0.5*x)+15) / 3;
    f = [f1;f2];
end
```