Лабораторная работа № 3. «Решние линейных и нелинейных систем разными методами»

1.Решение системы линейных уравнений с помощью QR разложения.

Невырожденная матрица A размера nxn с комплексными элементами может быть представлена в виде: A=QR, где Q — унитарная матрица размера nxn, а R — верхнетреугольная матрица размера nxn.

В случае, когда матрица A состоит из вещественных чисел, её можно представить в виде A=QR, где Q является ортогональной матрицей размера nxn, а R — верхнетреугольная матрица размера nxn.

Напомню, что матрица A является ортогональной матрицей, тогда и только тогда, когда выполняется одно из следующих эквивалентных условий: 1) строки матрицы A образуют ортонормированный базис, 2) столбцы матрицы A образуют ортонормированный базис, 3) $A^{-1} = A^T$, 4) $A \cdot A^T = E$.

Есть несколько способов нахождения QR разложения матрицы: метод Грама-Шмидта, метод Гивенса (метод вращений) и метод Хаусхолдера (метод отражений).

1 а. Поиск QR разложения методом Грама-Шмидта.

```
A = np.array([[6, 5, 0],[5, -1, 4],[5, 1, -14],[0, 4, 3]],dtype=float)
Q=np.zeros_like(A)
cnt = 0
for a in A.T:
    u = np.copy(a)
    for i in range(0, cnt):
        u -= np.dot(np.dot(Q[:, i].T, a), Q[:, i]) #метод Грама-Шмидта
    e = u / np.linalg.norm(u)  # Нормализован
    Q[:, cnt] = e
    cnt += 1
R = np.dot(Q.T, A)
```

```
np.set_printoptions(precision=4, suppress=True) #оставляем 4 знака после запятой
print(Q, R)
```

Рисунок 1. Программная реализация метода Грама-Шмидта

1 б. Поиск QR разложения методом Хаусхолдера.

Метод Хаусхолдера один из самых распространенных методов нахождения *QR* разложения матрицы *A*. Пусть *v* – *n*-мерный ненулевой вектор-столбец единичной длины (т.е. $\|v\|=1$), *H* – матрица $n\times n$ $H=E-2v\cdot v^T$ называется отражением Хаусхолдера или просто отражением. Можно проверить, что матрица *H* симметрична и ортогональна. Умножению матрицы *H* на произвольный вектор *x* можно дать следующую геометрическую интерпретацию: вектор *Hx* получается отражением вектора *x* относительно гиперплоскости, ортогональной вектору *v*.

Приведем пример. Рассмотрим вектор $x=(1,3,4)^T$, в котором требуется обнулить x_3 . Поскольку x_2 первый элемент отличный от нуля, то матрица Хаусхолдера принимает вид: $H=\begin{pmatrix}1 & 0\\ 0 & H'_{2x2}\end{pmatrix}$. Находим коэффициент: $\beta=sign(-x_2)\|y\|=-5$, где $sign(-x_2)$ знак, противоположный знаку второй координаты вектора x,

$\|y\|$ – вторая норма «усеченного» вектора x $y=(3,4)$, $\|y\|=\sqrt{9+16}$.

Приведем пример. Рассмотрим вектор $x=(1,3,4)^T$, в котором требуется обнулить x_3 . Поскольку x_2 первый элемент отличный от нуля, то матрица Хаусхолдера принимает вид: $H=\begin{pmatrix}1 & 0\\ 0 & H'_{2x2}\end{pmatrix}$. Находим коэффициент: $\beta=sign(-x_2)\|y\|=-5$, где $sign(-x_2)$ знак, противоположный знаку второй координаты вектора x,

$\|y\|$ – вторая норма «усеченного» вектора x $y=(3,4)$, $\|y\|=\sqrt{9+16}$.

Тогда $u=(y_1-\beta, y_2)=(8,4)$ и $v=\frac{u}{\|u\|}=\frac{(8,4)^T}{\sqrt{64+16}}=\frac{(2,1)^T}{\sqrt{5}}$.

Находим матрицу

$$H'=E-2vv^T=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}-\begin{pmatrix}2/\sqrt{5}\\ 1/\sqrt{5}\end{pmatrix}\cdot\begin{pmatrix}2/\sqrt{5} & 1/\sqrt{5}\end{pmatrix}=\begin{pmatrix}-0.6 & -0.8\\ -0.8 & 0.6\end{pmatrix}.$$

Результат: $x'=Hx=\begin{pmatrix}1 & 0 & 0\\ 0 & -0.6 & -0.8\\ 0 & -0.8 & 0.6\end{pmatrix}\cdot\begin{pmatrix}1\\ 3\\ 4\end{pmatrix}=\begin{pmatrix}1\\ -5\\ 0\end{pmatrix}$.

Проверка: $\|x'\|=\|x\|=\sqrt{26}$ (везде имеем в виду вторую норму).

Приведу реализацию метода Хаусхолдера на языке Python.

```
A = np.array([[6, 5, 0],[5, -1, 4],[5, 1, -14],[0, 4, 3]],dtype=float)
(r, c) = np.shape(A)
Q = np.identity(r)
R = np.copy(A)
for cnt in range(r - 1):
    x = R[cnt:, cnt]
    e = np.zeros_like(x)
    e[0] = np.linalg.norm(x)
    u = x - e
    v = u / np.linalg.norm(u)
    Q_cnt = np.identity(r)
    Q_cnt[cnt:, cnt:] -= 2.0 * np.outer(v, v)
    R = np.dot(Q_cnt, R)  # R=H(n-1)*...*H(2)*H(1)*A
    Q = np.dot(Q, Q_cnt)  # Q = H (N-1) * ... * H (2) * H (1) H
```

```
np.set_printoptions(precision=4, suppress=True) #оставляем 4 знака после запятой
print(Q)
print(R)
```

Рисунок 2. Программная реализация метода Хаусхолдера.

1.в Поиск QR разложения методом Гивенса.

Матрица Гивенса G_{kl} имеет следующий вид:

$$G_{kl} = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & \cos\phi & \cdots & -\sin\phi & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & \sin\phi & \cdots & \cos\phi & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}$$

Данная матрица отличается от единичной матрицы только подматрицей:

$$M(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$$

расположенной на строках и столбцах с номерами k и l. Является ортогональной.

Если дан вектор $a = [\,a_1 \quad \dots \quad a_n\,]^T \in \mathbb{R}^n$, $s = \sqrt{a_k^2 + a_l^2} \neq 0$, то выбрав:

$$\cos\phi = \frac{a_k}{\sqrt{a_k^2 + a_l^2}} \quad \sin\phi = \frac{-a_l}{\sqrt{a_k^2 + a_l^2}}$$

можно обнулить l-ю компоненту вектора a:

$$\begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} a_k \\ a_l \end{bmatrix} = \begin{bmatrix} \cos\phi \cdot a_k - \sin\phi \cdot a_l \\ \sin\phi \cdot a_k + \cos\phi \cdot a_l \end{bmatrix} = \begin{bmatrix} \frac{a_k^2 + a_l^2}{\sqrt{a_k^2 + a_l^2}} \\ \frac{-a_l \cdot a_k + a_k \cdot a_l}{\sqrt{a_k^2 + a_l^2}} \end{bmatrix} = \begin{bmatrix} \sqrt{a_k^2 + a_l^2} \\ 0 \end{bmatrix}$$

```
(r, c) = np.shape(A)
Q = np.identity(r)
R = np.copy(A)
(rows, cols) = np.tril_indices(r, -1, c)
for (row, col) in zip(rows, cols):
    if R[row, col] != 0:  # Q = 1, S = 0, R, Q без изменений
        r_ = np.hypot(R[col, col], R[row, col])  # d
        c = R[col, col]/r_
        s = -R[row, col]/r_
        G = np.identity(r)
        G[[col, row], [col, row]] = c
        G[row, col] = s
        G[col, row] = -s
        R = np.dot(G, R)  # R=G(n-1,n)*...*G(2n)*...*G(23,1n)*...*G(12)*A
        Q = np.dot(Q, G.T)  # Q=G(n-1,n).T*...*G(2n).T*...*G(23,1n).T*...*G(12).T
```

```
np.set_printoptions(precision=4, suppress=True) #оставляем 4 знака после запятой
print(Q)
print(R)
```

Рисунок 3. Программная реализация метода Гивенса

Решим исходную линейную систему с помощью QR разложения матрицы. Обратим внимание на то, что если QR разложение уже найдено, то найти обратную матрицу к Q не составит труда – это просто транспонированная матрица (или сопряженная, в комплексном случае). Итак, $AX = B$, используем разложение $QRX = B$, умножаем на транспонированную $Q^T QRX = Q^T B$, получаем верхнетреугольную систему $RX = Q^T B$, которую легко решить обратной подстановкой.

Решение системы линейных уравнений итерационным методом Якоби

Напомним, что исходная система имеет вид: $\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases}$. Допустим, что все коэффициенты на главной диагонали не равны нулю (в противном случае можно переобозначить переменные). Выразим из первого уравнения x_1, из второго уравнения x_2 и так далее. Получим систему $\begin{cases} x_1 = \dfrac{1}{a_{11}}(-a_{12}x_2 - \ldots - a_{1n}x_n + b_1) \\ x_2 = \dfrac{1}{a_{22}}(-a_{12}x_1 - \ldots - a_{2n}x_n + b_2) \\ \ldots \\ x_n = \dfrac{1}{a_{nn}}(-a_{n1}x_1 - \ldots - a_{nn-1}x_{n-1} + b_n) \end{cases}$

Выберем начальное приближение $x^0 = (x_1^0, x_2^0, \ldots x_n^0)$, подставим в правую часть системы, получим новое решение $x^1 = (x_1^1, x_2^1, \ldots x_n^1)$. Таким образом, имеем итерационный процесс: $\begin{cases} {x_1}^{k+1} = \dfrac{1}{a_{11}}(-a_{12}{x_2}^k - \ldots - a_{1n}{x_n}^k + b_1) \\ {x_2}^{k+1} = \dfrac{1}{a_{22}}(-a_{12}{x_1}^k - \ldots - a_{2n}{x_n}^k + b_2) \\ \ldots \\ {x_n}^{k+1} = \dfrac{1}{a_{nn}}(-a_{n1}{x_1}^k - \ldots - a_{nn-1}{x_{n-1}}^k + b_n) \end{cases}$.

Обычно записывают в матричном виде: $X^{k+1} = BX^k + C$, где B - квадратная матрица с нулями по главной диагонали, на остальных местах элементы вида $b_{ij} = \dfrac{-a_{ij}}{a_{ii}}$, C - вектор-столбец с элементами $c_i = \dfrac{b_i}{a_{ii}}$. Достаточным условием сходимости итерационного процесса является условие диагонального преобладания матрицы A: $\sum_{j, i \neq j} |a_{ij}| < |a_{ii}|$ (для любой строки i) или вытекающее отсюда условие для матрицы B: $\|B\|_1 = \max_i \sum_j |b_{ij}| < 1$, также можно показать, что для сходимости метода достаточно, чтобы любая норма матрицы B была меньше 1.

Пример. Решим систему $\begin{cases}100x_1+30x_2-70x_3=60\\15x_1-50x_2-5x_3=-40\\6x_1+2x_2+20x_3=28\end{cases}$ с точностью до $\epsilon=0.01$. Заметим, что матрица $A=\begin{pmatrix}100 & 30 & -70\\15 & -50 & -5\\6 & 2 & 20\end{pmatrix}$ не удовлетворяет условию диагонального преобладания, но в данной системе достаточно прибавить к первому уравнению второе:

$$\begin{cases}115x_1-20x_2-75x_3=20\\15x_1-50x_2-5x_3=-40\\6x_1+2x_2+20x_3=28\end{cases}\Rightarrow\begin{cases}x_1=20/115+20/115x_2+75/115x_3\\x_2=40/50+15/50x_1-5/50x_3\\x_3=28/20-6/20x_1-2/20x_2.\end{cases}$$

Получили систему, удовлетворяющую достаточному условию сходимости метода Якоби. Результат выполнения итераций представлен на рисунке:

k	x_1	x_2	x_3	norma(X_k- X_k-1)
0	0.0	0.0	0.0	
1	0.17391304347826086	0.8	1.4	1.4
2	1.2260869565217392	0.7121739130434783	1.2678260869565217	1.0521739130434784
3	1.1246124763705103	1.0410434782608697	0.9609565217391303	0.3288695652173914
4	0.9816748582230623	1.0412880907372402	0.9585119092627599	0.1429376181474448
5	0.9801230870387113	0.9986512665406427	1.0013687334593573	0.04285682419659742
6	1.0006580899153448	0.9993000527656776	1.0060979472343223	0.02853500287663345 7
7	1.0029160617287629	0.9995876322511712	1.0004125677488287	0.005687579485493588
8	1.0001973497929182	1.0000833561741346	0.9991664182586539	0.0027187119279446747

Рисунок 8. Пример работы методы Якоби

В левом столбце указан номер итерации. В последней строке набор координат (x_1, x_2, x_3) является приближенным решением исходной системы, найденным с указанной точностью. В последнем столбце выводится норма разности текущей итерации и предыдущей (максимум из модулей координат). Вывод таблицы оформлен в Python с помощью модуля PrettyTable. В лабораторной

Итерационный метод Зейделя.

Метод Зейделя является модификацией метода Якоби. В отличии от метода Якоби, в методе Зейделя при вычислении x_i^{k+1} используются уже найденные на $k+1$ итерации значения x_1^{k+1},…, x_{i-1}^{k+1}, то есть итерационный процесс выглядит следующим образом:

$$\begin{cases}x_1^{k+1}=\frac{1}{a_{11}}(-a_{12}x_2^{k}-\ldots-a_{1n}x_n^{k}+b_1)\\x_2^{k+1}=\frac{1}{a_{22}}(-a_{12}x_1^{k+1}-\ldots-a_{2n}x_n^{k}+b_2)\\\ldots\\x_n^{k+1}=\frac{1}{a_{nn}}(-a_{n1}x_1^{k+1}-\ldots-a_{nn-1}x_{n-1}^{k+1}+b_n)\end{cases}$$

, или в матричном виде: $X^{k+1}=B_1X^{k+1}+B_2X^{k}+C$,

где $B = B_1 + B_2$, B_1 нижнетреугольная матрица, B_2 верхнетреугольная матрица. Для сходимости метода Зейделя достаточно, чтобы $\max|b_{ij}| < 1$. Условием остановки итерационного процесса может служить условие $\|X^{k+1} - X^k\|_\infty < \varepsilon$, где ε заданная точность. Метод Зейделя сходится, как правило, быстрее метода Якоби.

Пример.

Решим пример из предыдущего пункта методом Зейделя

$$\begin{cases} x_1^{k+1} = 20/115 + 20/115\, x_2^k + 75/115\, x_3^k \\ x_2^{k+1} = 40/50 + 15/50\, x_1^{k+1} - 5/50\, x_3^k \\ x_3^{k+1} = 28/20 - 6/20\, x_1^{k+1} - 2/20\, x_2^{k+1} \end{cases}$$

$$X^{k+1} = \begin{pmatrix} 20/115 \\ 40/50 \\ 28/20 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 15/50 & 0 & 0 \\ -6/20 & -2/20 & 0 \end{pmatrix} \cdot X^{k+1} + \begin{pmatrix} 0 & 20/115 & 75/115 \\ 0 & 0 & -5/50 \\ 0 & 0 & 0 \end{pmatrix} \cdot X^k.$$

Результат выполнения итераций представлен на рисунке:

k	x_1	x_2	x_3	norma(X_k- X_k-1)
0	0.0	0.0	0.0	
1	0.9996013259932693	1.00014256311201	0.9998574388879898	1.00014256311201
2	0.9999318189464299	0.9998946542891818	1.0001053455598182	0.0003304919531605943

Рисунок 9. Метод Зейделя

Решение нелинейных уравнений.

Метод хорд

$f'f'' > 0$

$$x_{n+1} = x_n - \frac{(b - x_n) f(x_n)}{f(b) - f(x_n)} \qquad \{x_n\}\uparrow,\ x_n \le b \ \Rightarrow\ \exists\, x = \lim_{n\to\infty} x_n$$

$$x = x - \frac{(b - x) f(x)}{f(b) - f(x)} \ \Rightarrow\ f(x) = 0$$

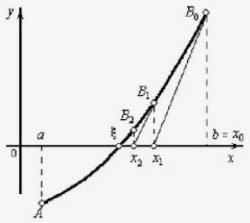
$f'f'' < 0$

$$x_{n+1} = x_n - \frac{(x_n - a) f(x_n)}{f(x_n) - f(a)} \qquad \{x_n\}\downarrow,\ x_n \ge a \ \Rightarrow\ \exists\, x = \lim_{n\to\infty} x_n$$

$$x = x - \frac{(x - a) f(x)}{f(x) - f(a)} \ \Rightarrow\ f(x) = 0$$

Метод Ньютона (касательной)

В качестве исходной точки x_0 выбирается тот конец интервала $[a, b]$, которому отвечает ордината того же знака, что и знак $f''(x)$.



$$y - f(x_0) = f'(x_0)(x - x_0). \qquad x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Комбинированный метод хорд и касательных.

- Если $f'(x) \cdot f''(x) > 0$:

$$a_{n+1} = a_n - \frac{f(a_n)(b_n - a_n)}{f(b_n) - f(a_n)}; \qquad b_{n+1} = b_n - \frac{f(b_n)}{f'(b_n)};$$

- Если $f'(x) \cdot f''(x) < 0$:

$$a_{n+1} = a_n - \frac{f(a_n)}{f'(a_n)}, \qquad b_{n+1} = b_n - \frac{f(b_n)(b_n - a_n)}{f(b_n) - f(a_n)};$$

Метод простых итераций

$1-й$ *шаг*: $f(x)=0 \Leftrightarrow x=\varphi(x),\ |\varphi'(x)|<1$ *на* $[a,b]$.

$2-й$ *шаг*: *задаём* $x^{(0)} \in [a,b],\ k=0.$

$3-й$ *шаг*: $x^{(k+1)}=\varphi(x^{(k)}).$

$4-й$ *шаг*: *если* $\left|x^{(k+1)}-x^{(k)}\right|<\varepsilon \Rightarrow x^{*}=x^{(k+1)},$

иначе $k=k+1$ *и переходим к шагу* 3.

Пример.

$$x^2-e^{-x}=0 \Leftrightarrow x=e^{-\frac{x}{2}},\ \varepsilon=0.001,\ [0.5,1],\ x^{(0)}=0.75.$$

$$x^{(1)}=0.6873,\ x^{(2)}=0.7091,\ x^{(3)}=0.7015,\ x^{(4)}=0.7042,\ x^{(5)}=0.7032.$$

$$\left|x^{(1)}-x^{(0)}\right|=0.0627,\ \left|x^{(2)}-x^{(1)}\right|=0.0218,\ \left|x^{(3)}-x^{(2)}\right|=0.0076,$$

$$\left|x^{(4)}-x^{(3)}\right|=0.0027,\ \left|x^{(5)}-x^{(4)}\right|=0.0010.$$

Метод Ньютона решения систем нелинейных уравнений

Рассмотрим систему нелинейных уравнений второго порядка

$$\begin{cases} f(x,y)=0 \\ g(x,y)=0 \end{cases}$$

Требуется построить последовательность $\{x_i,\ y_i\}$, которая при определенных условиях сходится к решению системы. Пусть задано начальное приближение $\{x_0,\ y_0\}$ (его можно определить графическим методом). Тогда очередное приближение:

$$\begin{cases} x_1=x_0+\Delta x \\ y_1=y_0+\Delta y \end{cases} \text{ и } \begin{cases} f(x_0+\Delta x, y_0+\Delta y)=0 \\ g(x_0+\Delta x, y_0+\Delta y)=0 \end{cases}$$

Разложим функцию в окрестности некоторой фиксированной точки по формуле Тейлора:

$$\begin{cases} f(x_0+\Delta x, y_0+\Delta y) = f(x_0,y_0) + \dfrac{\partial f(x_0,y_0)}{\partial x}\Delta x + \dfrac{\partial f(x_0,y_0)}{\partial y}\Delta y + R = 0 \\ g(x_0+\Delta x, y_0+\Delta y) = g(x_0,y_0) + \dfrac{\partial g(x_0,y_0)}{\partial x}\Delta x + \dfrac{\partial g(x_0,y_0)}{\partial y}\Delta y + R = 0 \end{cases}$$

Пренебрегая остаточным членом, получаем:

$$\begin{cases} \left(\dfrac{\partial f(x_0,y_0)}{\partial x}\Delta x + \dfrac{\partial f(x_0,y_0)}{\partial y}\Delta y = -f(x_0,y_0) \right) \\ \left(\dfrac{\partial g(x_0,y_0)}{\partial x}\Delta x + \dfrac{\partial g(x_0,y_0)}{\partial y}\Delta y = -g(x_0,y_0) \right) \end{cases}$$

Введем матрицу Якоби:

$$J(x,y) = \begin{vmatrix} \dfrac{\partial f(x,y)}{\partial x} & \dfrac{\partial f(x,y)}{\partial y} \\ \dfrac{\partial g(x,y)}{\partial x} & \dfrac{\partial g(x,y)}{\partial y} \end{vmatrix}$$

Тогда вместо системы нелинейных уравнений будем решать систему линейных уравнений относительно $\Delta x, \Delta y$:

$$\begin{vmatrix} \dfrac{\partial f(x,y)}{\partial x} & \dfrac{\partial f(x,y)}{\partial y} \\ \dfrac{\partial g(x,y)}{\partial x} & \dfrac{\partial g(x,y)}{\partial y} \end{vmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = -\begin{pmatrix} f(x,y) \\ g(x,y) \end{pmatrix}$$

А далее вычислять на каждой итерации:

$$x_{i+1} = x_i + \Delta x_i \text{ и } y_{i+1} = y_i + \Delta y_i,$$

где x_i – текущее приближение к корню, x_{i+1} – последующее приближение, $\Delta x_i, \Delta y_i$ – приращения к очередным приближениям.

Процесс заканчивается, когда $\|x_{i+1} - x_i\| < \varepsilon$

Рассмотрим пример: Найти решение системы нелинейных уравнений:

$$\begin{cases} x^2 + y^2 = 4 \\ y = 3x^2 \end{cases} \rightarrow \begin{cases} f(x,y) = 0 \\ g(x,y) = 0 \end{cases} \rightarrow \begin{cases} x^2 + y^2 - 4 = 0 \\ -3x^2 + y = 0 \end{cases}$$

Как отмечено ранее, система имеет не более двух различных решений. Построим матрицу Якоби:

$$\frac{\partial f}{\partial x} = 2x \quad \frac{\partial f}{\partial y} = 2y \quad \frac{\partial g}{\partial x} = -6x \quad \frac{\partial g}{\partial y} = 1$$

Тогда будем решать следующую систему линейных уравнений:

$$\begin{cases} 2x\Delta x + 2y\Delta y = 4 - x^2 - y^2 \\ -6x\Delta x + \Delta y = 3x^2 - y \end{cases} \quad (8)$$

Определим начальные приближения (см. рис. 14): $x_0 = 1$, $y_0 = 2$. Подставим эти значения в систему уравнений (8). Тогда система (8) на первой итерации будет иметь вид:

$$\begin{cases} 2\Delta x + 4\Delta y = -1 \\ -6\Delta x + \Delta y = 1 \end{cases} \quad (9)$$

Решать эту систему относительно неизвестных Δx, Δy можно любым известным способом. Получаем $\Delta x = -0{,}192$ и $\Delta y = -0{,}154$. Тогда $x_1 = x_0 + \Delta x = 1 - 0{,}192 = 0{,}808$ и $y_1 = y_0 + \Delta y = 2 - 0{,}154 = 1{,}846$. Новые приближения x_1, y_1 подставляем в систему (8) и, решая ее заново, найдем сначала Δx и Δy, а затем значения $x_2 = x_1 + \Delta x$, $y_2 = y_1 + \Delta y$ и т.д.

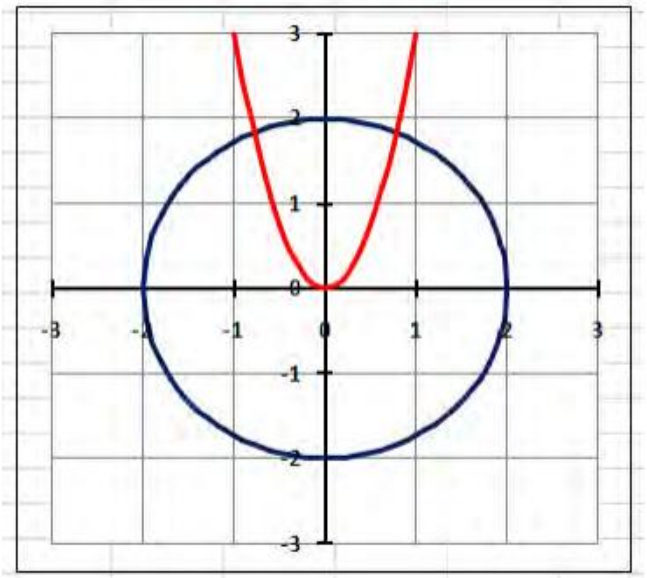


Рис. 14. Графики функций $x^2 + y^2 = 4$ и $y = 3x^2$

Методические указания к выполнению лабораторной работы.

В работе 5 заданий. Первое задание на нахождение и понимание QR разложения матрицы, второе на решение лин системы с помощью QR разложения, третье решение лин системы итерационными методами, четвертое – на решение нелинейного уравнения разными методами, пятое задание – на решение системы нелинейных уравнений.

Некоторую помощь можно найти в файле: Лаб3

https://colab.research.google.com/drive/158CUasycXn7YUZFY3QipGCrgDNvJCHpe?usp=sharing

1. QR разложение должно быть найдено приведенными в тексте алгоритмами, не нужно изобретать велосипед. Создайте для поиска QR разложения отдельную функцию и используйте эту функцию для решения 2 задания. Внимательно прочитайте текст выше и решите доп задание - найти матрицы Гивенса, Хаусхолдера. С методом Грама-Шмидта вы познакомились сна алгебре.
2. Придерживайтесь плана решения

План решения системы AX = b с помощью LU разложения.

   a. Записать систему в виде AX = b
   b. Найти QR разложение матрицы A самостоятельно написанной функцией.
   c. Записать систему в виде QRX = b.
   d. Записать систему в виде $RX = Q^Tb$.
   e. Решить систему $RX = Q^Tb$ самостоятельно написанным методом обратной подстановки.
   f. Вывести решение системы X.
   g. Проверить найденное решение с помощью функции np.linalg.solve().
3. Метод простых итераций также называют методом Якоби. И для метода Якоби и для метода Зейделя (для сходимости методов) важно диагональное преобладание. Я изменила все системы так, что диагональное преобладание везде есть .В паре вариантов нужно только переставить местами уравнения, в остальных вариантах дополнительные преобразования не требуются. Оформляйте вывод решения (все итерации) таблицами Pandas или PrettyTable
4. Постройте график функции. Графическим методом отделите корни. Если в указанном отрезке корней несколько, то постарайтесь найти все корни. Проверьте выполнение условий (знаки на концах интервала, знаки первых и вторых производных). Вывод решения оформляйте таблицами Pandas или PrettyTable. Примерный план решения:.
   a. Построить график функции.
   b. Выбрать интервал, на котором функция имеет единственный корень
   c. Проверить условия сходимости для каждого метода.
   d. Подсказака. Для метода половинного деления использовать формулу
   e. $x_n = \frac{a+b}{2}$
   f. Для метода хорд использовать формулу $x_n = x_{n-1} - \frac{f(x_{n-1})(b - x_{n-1})}{f(b) - f(x_{n-1})}$ или

$$x_n = x_{n-1} - \frac{f(x_{n-1})(x_{n-1} - a)}{f(x_{n-1}) - f(a)}$$

g. Для метода Ньютона (касательных) $x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$

h. Стоп по условию $\left|x_n - x_{n-1}\right| < \varepsilon$

i. В оформлении задания должны быть введены используемые формулы с помощью Latex

j. Вывести полученные таблицы итераций

k. Проверить полученное решение методами NumPy

5. Постройте график функции. Для построения графика можно использовать метод построения для функции, заданной неявно. По графику предположите начальное приближение. Вывод решения оформляйте таблицами Pandas или PrettyTable. Построить графики функций. (вспомните первую лаб работу)
   a. Построить графики функций (как в 1 лаб работе)
   b. Определите начальное приближение исходя из графиков.
   c. Вычислить производные, составить матрицу Якоби. (Вывести на экран)
   d. Составить систему линейных уравнений относительно приращений x и y. (вывести на экран)
   e. Решить линейную систему методом Крамера. (вывести на экран решение)
   f. Составить итерационную систему. (вывести на экран)
   g. Стоп по условию $\left\|x_n - x_{n-1}\right\| < \varepsilon$.
   h. Вывести количество итераций.
   i. Проверить полученные решения подстановкой и сравнить с решениями функцией Питон
   j. Plt.countor - для построения функции, заданной неяно, в работе Лазутова (есть на почте)

Контрольные вопросы к лабораторной работе.

1. Определение QR разложения.
2. Условие существования QR разложения.
3. Определение ортогональной матрицы, определение унитарной матрицы. Основное свойство.
4. План решения системы лин уравнений с помощью QR разложения
5. Определение линейной системы уравнений.
6. Методы решения систем: точные, численные, итерационные.
7. Основная идея метода Якоби (простых итераций). Условия сходимости метода
8. Основная идея метода Зейделя. Условия сходимости метода
9. Методы решений нелинейных уравнений. Знать формулы и условия сходимости
10. План решения нелинейной системы методом Ньютона
11. План решения линейной системы уравнений с помощью LU разложения

Задание к лабораторной работе

Вариант 1

1. Создать матрицу 3x3 из случайных целых чисел из отрезка [-10, 10]. Найти преобразование Хаусхолдера (матрицу H), которое при умножении на матрицу преобразовывает элемент a_{11} и зануляет элементы a_{21} и a_{31} (вторая норма первого столбца не должна измениться). Найти QR разложение матрицы методом Хаусхолдера. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Хаусхолдера. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 4.4x_1 - 2.5x_2 + 19.2x_3 - 10.8x_4 = 4.3 \\ 5.5x_1 - 9.3x_2 - 14.2x_3 + 13.2x_4 = 6.8 \\ 7.1x_1 - 11.5x_2 + 5.3x_3 - 6.7x_4 = -1.8 \\ 14.2x_1 + 23.4x_2 - 8.8x_3 + 5.3x_4 = 7.2 \end{cases}.$$

3. Решить систему методом простых итераций (методом Якоби) с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение.

$$\begin{cases} 5.7x_1 + 3.3x_2 + 1.3x_3 = 2.1 \\ 3.5x_1 - 6.7x_2 + 2.8x_3 = 1.7 \\ 3.1x_1 + 2.8x_2 - 8.7x_3 = 0.8 \end{cases}$$

4. Решить нелинейное уравнение методом половинного деления с точностью 10^{-3} и комбинированным методом с точностью 10^{-5}. До решения отделить корни графическим методом, проверить выполнение условий метода.

№ Варианта	Функция	a	b
1	$2{,}74x^3 - 1{,}93x^2 - 15{,}28x - 3{,}72$	-3	4

5. Решить систему нелинейных уравнений методом Ньютона с точностью до 10^{-4}

№ Варианта	Система уравнений
1	$\begin{cases} \sin(x+1) - y = 1.2 \\ 2x + \cos y = 2 \end{cases}$

Вариант 2

1. Создать матрицу 3x3 из случайных целых чисел из отрезка [-8, 8]. Методом Грама-Шмидта ортогонализовать первые два столбца матрицы. Найти QR разложение матрицы методом Грама-Шмидта. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Грама-Шмидта. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 8.2x_1 - 3.2x_2 + 14.2x_3 + 14.8x_4 = -8.4 \\ 5.6x_1 - 12x_2 + 15x_3 - 6.4x_4 = 4.5 \\ 5.7x_1 + 3.6x_2 - 12.4x_3 - 2.3x_4 = 3.3 \\ 6.8x_1 + 13.2x_2 - 6.3x_3 - 8.7x_4 = 14.3 \end{cases}.$$

3. Решить систему методом Зейделя с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение. $\begin{cases} 3.1x_1 + 2.8x_2 + 4.9x_3 = 0.2 \\ 1.9x_1 + 4.1x_2 + 2.1x_3 = 2.1 \\ 7.5x_1 + 3.8x_2 + 4.8x_3 = 5.6 \end{cases}$
4. Решить нелинейное уравнение методом половинного деления с точностью 10^{-3} и комбинированным методом с точностью 10^{-5} . До решения отделить корни графическим методом, проверить выполнение условий метода.

2	$-1{,}38x^3-5{,}42x^2+2{,}57x+10{,}95$	-5	3

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

2	$\begin{cases} \sin(x)+2y=2 \\ 2x+\cos(y-1)=0.7 \end{cases}$

Вариант 3

1. Создать матрицу 3x3 из случайных целых чисел из отрезка [-8, 10]. Найти матрицу Гивенса, которая зануляет элемент a_{31}. Найти QR разложение матрицы методом Гивенса. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Гивенса. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 5.7x_1 - 7.8x_2 - 5.6x_3 - 8.3x_4 = 2.7 \\ 6.6x_1 + 13.1x_2 - 6.3x_3 + 4.3x_4 = -5.5 \\ 14.7x_1 - 2.8x_2 + 5.6x_3 - 12.1x_4 = 8.6 \\ 8.5x_1 + 12.7x_2 - 23.7x_3 + 5.7x_4 = 14.7 \end{cases}.$$

3. Решить систему методом Зейделя с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение. $\begin{cases} 1.21x_1 - 0.18x_2 + 0.75x_3 = 0.11 \\ 0.13x_1 + 1.75x_2 - 0.11x_3 = 2 \\ 3.01x_1 - 0.33x_2 + 1.11x_3 = 0.13 \end{cases}$
4. Решить нелинейное уравнение методом хорд с точностью 10^{-4} и методом Ньютона (касательных) с точностью 10^{-5}. До решения отделить корни графическим методом, проверить выполнение условий метода.

3	$x^3+2{,}84x^2-5{,}606x-14{,}766$	-4	3

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

3	$\begin{cases} \sin(x+0.5)-y=1 \\ x+\cos(y-2)=0 \end{cases}$

Вариант 4.

1. Создать матрицу 3x3 из случайных целых чисел из отрезка [-10, 10]. Найти преобразование Хаусхолдера (матрицу H), которое при умножении на матрицу преобразовывает элемент a_{22} и зануляет элемент a_{32} (вторая норма второго столбца не должна измениться). Найти QR разложение матрицы методом Хаусхолдера. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Хаусхолдера. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 3.8x_1 + 14.2x_2 + 6.3x_3 - 15.5x_4 = 2.8 \\ 8.3x_1 - 6.6x_2 + 5.8x_3 + 12.2x_4 = -4.7 \\ 6.4x_1 - 8.5x_2 - 4.3x_3 + 8.8x_4 = 7.7 \\ 17.1x_1 - 8.3x_2 + 14.4x_3 - 7.2x_4 = 13.5 \end{cases}.$$

3. Решить систему методом простых итераций(Якоби) с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение. $\begin{cases} 5.3x_1 + 2.1x_2 + 2.8x_3 = 0.8 \\ 4.1x_1 + 6.7x_2 + 4.8x_3 = 5.7 \\ 2.7x_1 + 1.8x_2 + 8.1x_3 = 3.2 \end{cases}$
4. Решить нелинейное уравнение методом половинного деления с точностью 10^{-3} и методом хорд с точностью 10^{-4}

4	$x^3 - 1{,}89x^2 - 2x + 1{,}76$	-3	4

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

4	$\begin{cases} \cos(x+0.5) - y = 2 \\ \sin y - 2x = 1 \end{cases}$

Вариант 5

1. Создать матрицу 4x4 из случайных целых чисел из отрезка [-8, 8]. Методом Грама-Шмидта ортогонализовать первые два столбца матрицы. Найти QR разложение матрицы методом Грама-Шмидта. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Грама-Шмидта. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 15.7x_1 + 6.6x_2 - 5.7x_3 + 11.5x_4 = -2.4 \\ 8.8x_1 - 6.7x_2 + 5.5x_3 - 4.5x_4 = 5.6 \\ 6.3x_1 - 5.7x_2 - 23.4x_3 + 6.6x_4 = 7.7 \\ 14.3x_1 + 8.7x_2 - 15.7x_3 - 5.8x_4 = 23.4 \end{cases}.$$

3. Решить систему методом Зейделя с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение. $\begin{cases} 6.15x_1 - 1.72x_2 - 1.23x_3 = 2.15 \\ 0.72x_1 + 5.67x_2 + 1.18x_3 = 1.43 \\ 2.57x_1 - 1.34x_2 - 3.68x_3 = 1.03 \end{cases}$
4. Решить нелинейное уравнение методом хорд с точностью 10^{-4} и комбинированным методом с точностью 10^{-5}. До решения отделить корни графическим методом, проверить выполнение условий метода.

5	$-2{,}7x^3 - 1{,}48x^2 + 19{,}23x + 6{,}35$	-4	4

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

5	$\begin{cases} \sin(x+1.5) - y + 2.9 = 0 \\ \cos(y-2) + x = 0 \end{cases}$

Вариант 6.

1. Создать матрицу 3x3 из случайных целых чисел из отрезка [-8, 10]. Найти матрицу Гивенса, которая зануляет элемент a_{32}. Найти QR разложение матрицы методом Гивенса. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Гивенса. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 4.3x_1 - 12.1x_2 + 23.2x_3 - 14.1x_4 = 15.5 \\ 2.4x_1 - 4.4x_2 + 3.5x_3 + 5.5x_4 = 2.5 \\ 5.4x_1 + 8.3x_2 - 7.4x_3 - 12.7x_4 = 8.6 \\ 6.3x_1 - 7.6x_2 + 1.34x_3 + 3.7x_4 = 12.1 \end{cases}.$$

3. Решить систему методом простых итераций (методом Якоби) с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение.

$$\begin{cases} 3.2x_1 - 2.5x_2 + 3.7x_3 = 6.5 \\ 0.5x_1 + 0.34x_2 + 1.7x_3 = -0.24 \\ 1.6x_1 + 2.3x_2 - 1.5x_3 = 4.3 \end{cases}$$

4. Решить нелинейное уравнение методом касательных (Ньютона) с точностью 10^{-4} и комбинированным методом точностью 10^{-5}. До решения отделить корни графическим методом, проверить выполнение условий метода.

6	$2x^3+3{,}41x^2-23{,}74x+2{,}95$	-5	4

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

№ Варианта	Система уравнений
6	$\begin{cases} \sin y - 2x = 1.6 \\ \cos(x+0.5) + y = 0.8 \end{cases}$

Вариант 7

1. Создать матрицу 4x4 из случайных целых чисел из отрезка [-10, 10]. Найти преобразование Хаусхолдера (матрицу H), которое при умножении на матрицу преобразовывает элемент a_{11} и зануляет элементы a_{21}, a_{31} , a_{41} (вторая норма первого столбца не должна измениться). Найти QR разложение матрицы методом Хаусхолдера. Проверить методом np.linalg

2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Хаусхолдера. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 4.4x_1 - 2.5x_2 + 19.2x_3 - 10.8x_4 = 4.3 \\ 5.5x_1 - 9.3x_2 - 14.2x_3 + 13.2x_4 = 6.8 \\ 7.1x_1 - 11.5x_2 + 5.3x_3 - 6.7x_4 = -1.8 \\ 14.2x_1 + 23.4x_2 - 8.8x_3 + 5.3x_4 = 7.2 \end{cases}.$$

3. Решить систему методом простых итераций (методом Якоби) с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение.

$$\begin{cases} 4.7x_1 + 3.3x_2 + 1.3x_3 = 2.1 \\ 3.5x_1 - 6.7x_2 + 2.8x_3 = 1.7 \\ 4.1x_1 + 5.8x_2 - 4.7x_3 = 0.8 \end{cases}$$

4. Решить нелинейное уравнение методом хорд с точностью 10^{-4} и комбинированным методом с точностью 10^{-5}. До решения отделить корни графическим методом, проверить выполнение условий метода.

7	$x^3+2{,}28x^2-1{,}934x-3{,}907$	-4	2

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

7	$\begin{cases} \sin(x-1)+y=0.1 \\ x-\sin(y+1)=0.8 \end{cases}$

Вариант 8

1. Создать матрицу 3x3 из случайных целых чисел из отрезка [-10, 8]. Методом Грама-Шмидта ортогонализовать первые два столбца матрицы. Найти QR разложение матрицы методом Грама-Шмидта. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Грама-Шмидта. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 8.2x_1 - 3.2x_2 + 14.2x_3 + 14.8x_4 = -8.4 \\ 5.6x_1 - 12x_2 + 15x_3 - 6.4x_4 = 4.5 \\ 5.7x_1 + 3.6x_2 - 12.4x_3 - 2.3x_4 = 3.3 \\ 6.8x_1 + 13.2x_2 - 6.3x_3 - 8.7x_4 = 14.3 \end{cases}.$$

3. Решить систему методом Зейделя с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение. $\begin{cases} 3.1x_1 + 2.8x_2 + 5.9x_3 = 0.2 \\ 1.9x_1 + 4.1x_2 + 2.1x_3 = 2.1 \\ 7.5x_1 + 3.8x_2 + 4.8x_3 = 5.6 \end{cases}$
4. Решить нелинейное уравнение методом хорд с точностью 10^{-5} и половинным делением с точностью 10^{-3}. До решения отделить корни графическим методом, проверить выполнение условий метода.

№ Варианта	Функция	a	b
8	$3x^3+1{,}7x^2-15{,}42x+6{,}89$	-4	3

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

8	$\begin{cases} \cos(x+y)+2y=0 \\ x+\sin y=0.6 \end{cases}$

Вариант 9

1. Создать матрицу 4x4 из случайных целых чисел из отрезка [-8, 10]. Найти матрицу Гивенса, которая зануляет элемент a_{42}. Найти QR разложение матрицы методом Гивенса. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Гивенса. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 5.7x_1 - 7.8x_2 - 5.6x_3 - 8.3x_4 = 2.7 \\ 6.6x_1 + 13.1x_2 - 6.3x_3 + 4.3x_4 = -5.5 \\ 14.7x_1 - 2.8x_2 + 5.6x_3 - 12.1x_4 = 8.6 \\ 8.5x_1 + 12.7x_2 - 23.7x_3 + 5.7x_4 = 14.7 \end{cases}.$$

3. Решить систему методом Зейделя с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение. $\begin{cases} 2.21x_1 - 0.18x_2 + 0.75x_3 = 0.11 \\ 0.13x_1 + 3.75x_2 - 0.11x_3 = 2 \\ 3.01x_1 - 0.33x_2 + 1.11x_3 = 0.13 \end{cases}$
4. Решить нелинейное уравнение методом Ньютона(касательных) с точностью 10^{-4} и комбинированным методом с точностью 10^{-5} . До решения отделить корни графическим методом, проверить выполнение условий метода.

9	$-1{,}8x^3 - 2{,}94x^2 + 10{,}37x + 5{,}38$	-5	4

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

9	$\begin{cases} \cos(x+0.5) - y = 2 \\ \sin y - 2x = 1 \end{cases}$

Вариант 10.

1. Создать матрицу 4x4 из случайных целых чисел из отрезка [-10, 10]. Найти преобразование Хаусхолдера (матрицу H), которое при умножении на матрицу зануляет элементы a_{32}, a_{42} (вторая норма второго столбца не должна измениться). Найти QR разложение матрицы методом Хаусхолдера. Проверить методом np.linalg
2. Решить систему с помощью QR разложения матрицы A. QR разложение найти методом Хаусхолдера. Проверить полученное решение непосредственной подстановкой в исходную систему, а также методом np.solve.

$$\begin{cases} 3.8x_1 + 14.2x_2 + 6.3x_3 - 15.5x_4 = 2.8 \\ 8.3x_1 - 6.6x_2 + 5.8x_3 + 12.2x_4 = -4.7 \\ 6.4x_1 - 8.5x_2 - 4.3x_3 + 8.8x_4 = 7.7 \\ 17.1x_1 - 8.3x_2 + 14.4x_3 - 7.2x_4 = 13.5 \end{cases}.$$

3. Решить систему методом простых итераций (методом Якоби) с точностью до 10^{-3}. Проверить выполнение достаточного условия сходимости. Если условие не выполняется, в программе выполнить эквивалентные преобразования системы, после этого привести к удобному для итераций виду. Оформить итерации в виде таблицы (можно пользоваться модулями PrettyTable или Pandas). Проверить полученное решение.

$$\begin{cases} 5.3x_1 + 2.1x_2 + 2.8x_3 = 0.8 \\ 4.1x_1 + 6.7x_2 + 1.8x_3 = 2.7 \\ 2.7x_1 + 1.8x_2 + 4.1x_3 = 3.2 \end{cases}$$

4. Решить нелинейное уравнение методом половинного деления с точностью 10^{-3} и комбинированным методом точностью 10^{-5}. До решения отделить корни графическим методом, проверить выполнение условий метода.

10	$x^3 - 3{,}12x^2 - 3{,}5x + 2{,}458$	-3	5

5. Решить систему нелинейных уравнений методом Ньютона с точностью 10^{-4}

10	$\begin{cases} \sin(0.5x + y) - 1.2y = 1 \\ x^2 + y^2 = 1 \end{cases}$