МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего

образования

Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

Лабораторная работа 3

По дисциплине «Прикладная математика»

Выполнили студенты группы
№M32101
Подколзин Олег Иванович
Якимов Даниил Павлович
Юрченко Владислав Витальевич

Проверила:
Москаленко Мария
Александровна

САНКТ-ПЕТЕРБУРГ

LU-разложение

Представление матрицы A в виде произведения двух матриц L и U.

$$A = LU$$

Где L — это нижняя треугольная матрица, а U верхняя треугольная матрица.

$$L = \begin{pmatrix} 1 & 0 & & \cdots & & 0 \\ \ell_{2,1} & \ddots & & & & \\ & \ddots & 1 & & & \vdots \\ \vdots & \cdots & \ell_{n+1,n} & 1 & & \\ & & \vdots & \ddots & \ddots & 0 \\ \ell_{N,1} & \cdots & \ell_{N,n} & \cdots & \ell_{N,N-1} & 1 \end{pmatrix} U = \begin{pmatrix} u_{1,1} & u_{1,2} & & \cdots & & u_{1,N} \\ 0 & \ddots & & \cdots & & u_{2,N} \\ & & u_{n-1,n-1} & & & \\ \vdots & & & u_{n,n} & & \vdots \\ & & & & \ddots & \\ 0 & & \cdots & & 0 & u_{N,N} \end{pmatrix}$$

Данный метод работает только при условии, что матрица A невырожденная.

PLU/LUP-разложение

Данный метод похож на предыдущий, но в то же время позволяет находить разложение для всех квадратных матриц. Достигается это с помощью дополнительной матрицы перестановки P.

$$P_\sigma = \begin{pmatrix} \mathbf{e}_{\sigma(1)} \\ \mathbf{e}_{\sigma(2)} \\ \vdots \\ \mathbf{e}_{\sigma(n)} \end{pmatrix},$$

где e_i — вектор размерности n, i-й элемент которого равен единице, а все остальные элементы являются нулями.

Тогда чтобы получить исходную матрицу необходимо перемножить P, L и U.

$$A = PLU$$

```
A
[[ 1.  2.  3.  4.]
 [ 5.  0.  7.  8.]
 [ 9. 10.  0. 12.]
 [13. 14. 15.  0.]]
P
[[0. 0. 0. 1.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [1. 0. 0. 0.]]
L
[[ 1.          0.          0.          0.        ]
 [ 0.38461538  1.          0.          0.        ]
 [ 0.69230769 -0.05714286  1.          0.        ]
 [ 0.07692308 -0.17142857 -0.19944598  1.        ]]
U
[[ 13.          14.          15.           0.        ]
 [  0.          -5.38461538   1.23076923   8.        ]
 [  0.           0.         -10.31428571  12.45714286]
 [  0.           0.           0.           7.85595568]]
```

Пример PLU разложения

```
A
[[1. 2. 3. 4.]
 [5. 6. 7. 8.]
 [9. 3. 7. 6.]
 [5. 4. 3. 2.]]
P
[[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
L
[[1.   0.   0.   0.  ]
 [5.   1.   0.   0.  ]
 [9.   3.75 1.   0.  ]
 [5.   1.5  0.   1.  ]]
U
[[  1.   2.   3.   4.]
 [  0.  -4.  -8. -12.]
 [  0.   0.  10.  15.]
 [  0.   0.   0.   0.]]
```

Если матрица невырожденная, то P является единичной матрицей.

Решение уравнений с помощью разложения

С помощью разложения можно также решать уравнения вида

$$Ax = b$$

$$A = LU \Rightarrow LUx = b$$

Произведем замену

$$Ux = y$$

и получим

$$Ly = b$$

Далее решим это уравнение относительно y с помощью метода прямой подстановки.
Затем, зная y решим изначальную подстановку относительно x с помощью обратной подстановки.

Для PLU-разложения преобразования выглядят так:

$$PLUx = b$$
$$LUx = P^T b$$
$$Ux = y$$
$$Ly = P^T b$$

Метод Зейделя

Смысл этого метода состоит в расчётах **i**-ой координаты новой точки x по известным **(i-1)** координатам новой точки и по **(n-i+1)** координатам старой точки.

Итерации продолжаются до достижения необходимой точности **ε**.

```
A
[[ -6.7  -2.9   0.   -0.9  -2.9]
 [ -0.9  -4.7  -1.9  -1.9   0. ]
 [ -3.9  -3.9 -10.6  -1.9  -0.9]
 [ -1.9  -0.9  -2.9  -5.7   0. ]
 [  0.   -0.9  -2.9  -2.9  -6.7]]
b
[31 37 27 91 42]
```

0	0.0	0.0	0.0	0.0	0.0
1	-4.6268656716417915	-6.986344871387743	1.725615199793885	-14.197461213865598	0.06806922762043728
2	0.2747335014036157	-2.883138877558937	0.9515749970275467	-16.085392307762056	0.66908883446763
3	-1.5078211638879644	-1.4656825664053745	1.3732764133840623	-15.92956615397576	0.22870963544488931
4	-1.9516670091994641	-1.6141594009779268	1.600666039655223	-15.873863287130007	0.12612171321746302
5	-1.8504796637429328	-1.747977347325377	1.6113974572469532	-15.891923096896802	0.14726924889214899
...	...	...	...	...	...
17	-1.8042206987525324	-1.7441233144703385	1.5949301218611072	-15.899573165393702	0.15719041929370559
18	-1.8042207097214082	-1.7441233145827253	1.5949301256292279	-15.899573163636779	0.1571904169173654
19	-1.8042207088802025	-1.7441233169773351	1.5949301260876085	-15.899573163772295	0.1571904170992824

Нахождение обратной матрицы

Для нахождения обратной матрицы мы также решаем наше уравнение, только на месте вектора b у нас будет единичная матрица (на главной диагонали все нули)

$$A * A^{-1} = E$$

```
A
[[1.         0.5        0.33333333 0.25      ]
 [0.5        0.33333333 0.25       0.2       ]
 [0.33333333 0.25       0.2        0.16666667]
 [0.25       0.2        0.16666667 0.14285714]]
Inverse A
[[   16.  -120.   240.  -140.]
 [ -120.  1200. -2700.  1680.]
 [  240. -2700.  6480. -4200.]
 [ -140.  1680. -4200.  2800.]]
A * A-1
[[ 1.  0.  0.  0.]
 [ 0.  1. -0.  0.]
 [ 0. -0.  1. -0.]
 [ 0.  0.  0.  1.]]
```

Разреженная матрица

Матрица с преимущественно нулевыми элементами. Программно она представляется в виде трех массивов. В первом массиве находится ненулевые значения. Во втором массиве находится индекс i-го элемента первой матрицы. Третий массив - массив индексации строк для индекса о хранит кол-во ненулевых элементов в строках до i-1

Матрицы Гильберта

Матрица Гильберта строится следующим образом:

$$a_{ij} = \frac{1}{i+j-1},\ i, j\ =\ 1..\,k$$

```
[[1.         0.5        0.33333333 0.25       0.2       ]
 [0.5        0.33333333 0.25       0.2        0.16666667]
 [0.33333333 0.25       0.2        0.16666667 0.14285714]
 [0.25       0.2        0.16666667 0.14285714 0.125     ]
 [0.2        0.16666667 0.14285714 0.125      0.11111111]]
```

```
(0, 0)    1.0
(0, 1)    0.5
(0, 2)    0.3333333333333333
(0, 3)    0.25
(0, 4)    0.2
(1, 0)    0.5
(1, 1)    0.3333333333333333
(1, 2)    0.25
(1, 3)    0.2
(1, 4)    0.16666666666666666
(2, 0)    0.3333333333333333
(2, 1)    0.25
(2, 2)    0.2
(2, 3)    0.16666666666666666
(2, 4)    0.14285714285714285
(3, 0)    0.25
(3, 1)    0.2
(3, 2)    0.16666666666666666
(3, 3)    0.14285714285714285
(3, 4)    0.125
(4, 0)    0.2
(4, 1)    0.16666666666666666
(4, 2)    0.14285714285714285
(4, 3)    0.125
(4, 4)    0.1111111111111111
```

Итерационный метод для матрицы Гильберта

Размер матрицы	Количество итераций	Время с.
2	66	0.0021982192993164062
3	901	0.050623416900634766
4	16023	1.3679537773132324
5	289345	33.30628752708435

Прямой метод для матрицы Гильберта

Размер матрицы	Время с.	Время для не Гильбертовой матрицы
10	0.02892446517944336	0.0167593956
50	0.15837955474853516	0.1082468033
100	0.45406484603881836	0.2405164242
500	13.677391529083252	11.0410738
1000	100.23388838768005	91.05975771

Исследование методов решения СЛАУ

Прямой метод

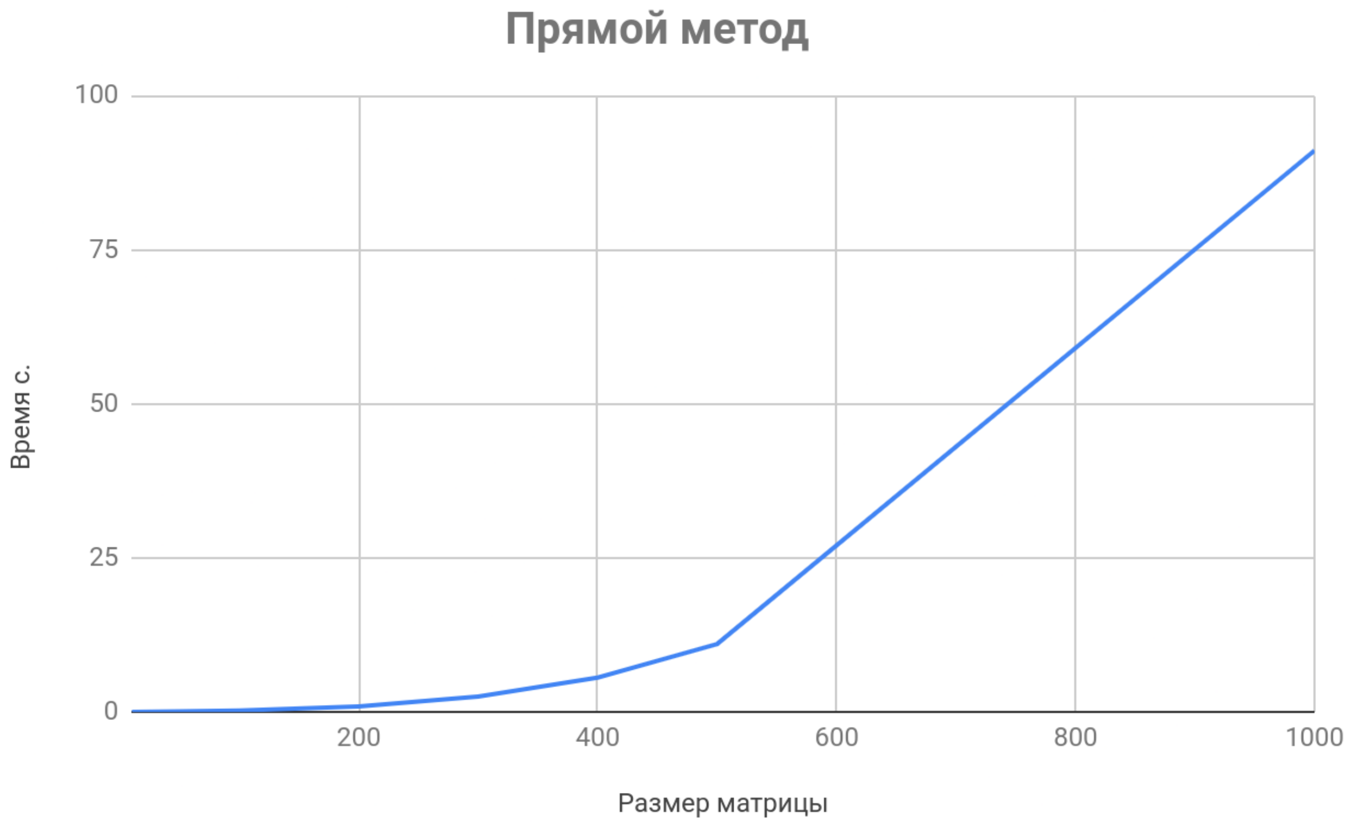
```
Матрица A
[[-3.55396536  0.34603464 -3.9       ]
 [-1.15790269 -3.24508311 -2.08718043]
 [-1.35837994 -3.57120102 -4.92958097]]
Наш x
[53. 98. 98.]
Полученное значение b при умножении A на наш x
[-536.64876871 -583.93066959 -905.07077193]
x by numpy
[53. 98. 98.]
Изначальное значение b
[-536.64876871 -583.93066959 -905.07077193]
```

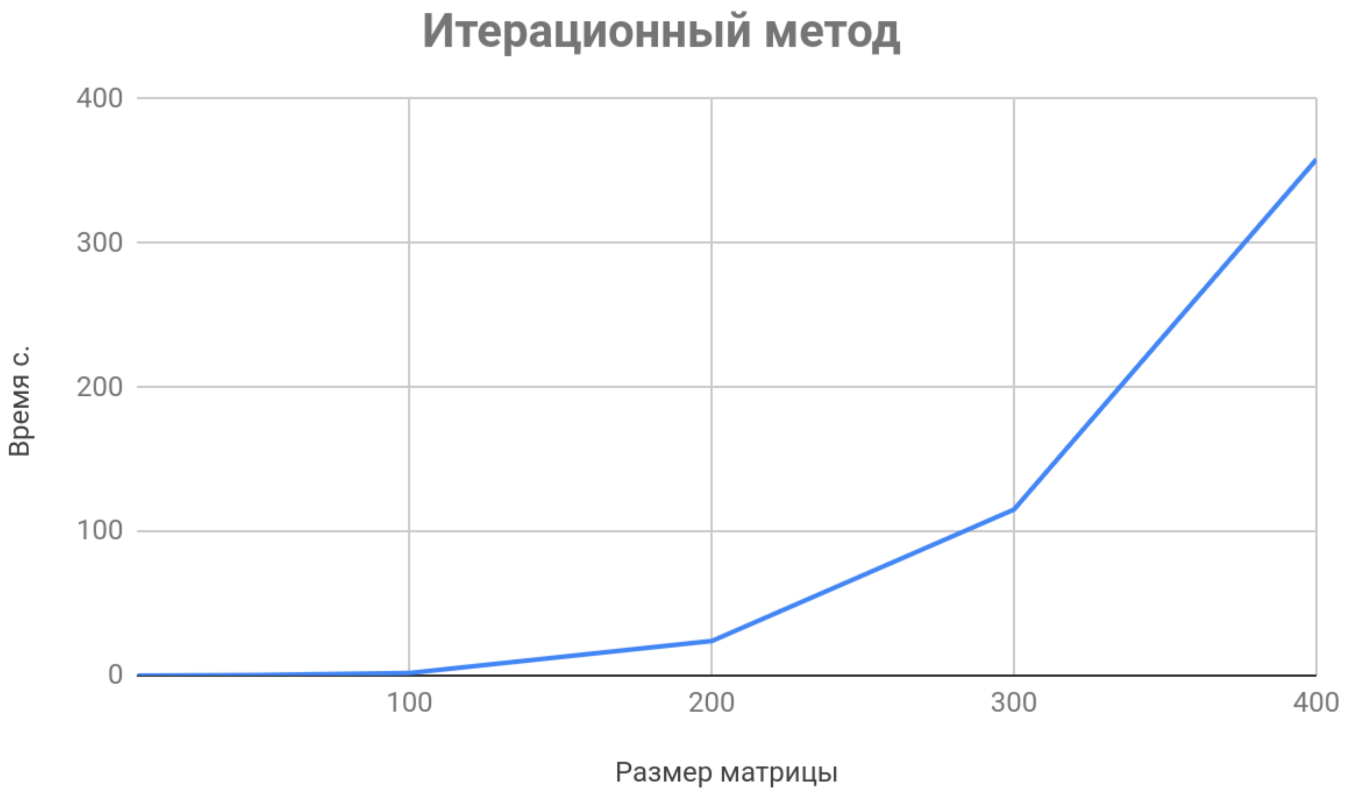
Итерационный метод

```
Матрица A
[[-3.55396536  0.34603464 -3.9       ]
 [-1.15790269 -3.24508311 -2.08718043]
 [-1.35837994 -3.57120102 -4.92958097]]
Наш x
[53. 98. 98.]
Полученное значение b при умножении A на наш x
[-536.64876872 -583.9306696  -905.07077193]
x by numpy
[53. 98. 98.]
Изначальное значение b
[-536.64876871 -583.93066959 -905.07077193]
```

```
Матрица A
[[-5.91285277 -3.9        -0.09931186  0.02938316 -1.94292407]
 [-0.74405702 -6.88027298 -1.04277451 -3.80655843 -1.28688303]
 [ 0.33439796 -1.59055968 -2.21085099 -0.95468928  0.        ]
 [-3.65278497 -0.97941911 -1.01605246 -4.97941167  0.66884487]
 [-1.9        -3.56238184  0.66505783 -2.75245789 -7.5497819 ]]
Наш x
[48. 74. 61. 40. 86.]
Полученное значение b при умножении A на наш x
[ -744.3910999   -871.3984593   -274.69979514  -451.44570047
 -1073.62728738]
x by numpy
[48. 74. 61. 40. 86.]
Изначальное значение b
[ -744.3910999   -871.3984593   -274.69979514  -451.44570047
 -1073.62728738]
```

```
Матрица A
[[-5.91285277 -3.9        -0.09931186  0.02938316 -1.94292407]
 [-0.74405702 -6.88027298 -1.04277451 -3.80655843 -1.28688303]
 [ 0.33439796 -1.59055968 -2.21085099 -0.95468928  0.        ]
 [-3.65278497 -0.97941911 -1.01605246 -4.97941167  0.66884487]
 [-1.9        -3.56238184  0.66505783 -2.75245789 -7.5497819 ]]
Наш x
[48. 74. 61. 40. 86.]
Полученное значение b при умножении A на наш x
[ -744.39109992  -871.39845929  -274.69979514  -451.44570048
 -1073.62728738]
x by numpy
[48. 74. 61. 40. 86.]
Изначальное значение b
[ -744.3910999   -871.3984593   -274.69979514  -451.44570047
 -1073.62728738]
```


Прямой метод
100
75
50
25
0
Время с.
200
400
600
800
1000
Размер матрицы


Итерационный метод
400
300
200
100
0
Время с.
100
200
300
400
Размер матрицы

Вывод

В ходе лабораторной работы мы реализовали LU разложение, с помощью которого написали алгоритм решения СЛАУ. Также реализовали итерационный метод решения СЛАУ - метод Зейделя. Протестировали и установили, что итерационный метод гораздо быстрее работает для матриц большого размера - это его преимущество по сравнению с прямым методом. Однако для матрицы Гильберта итерационный метод работает плохо и начиная с размерности 5 он очень плохо сходится, так как требуется большое кол-во итераций. Это объясняется плохой обусловленностью таких матриц.