|  |  |
| --- | --- |
| МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ПЕРМСКОГО КРАЯ | |
| Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования  «Пермский государственный национальный исследовательский университет» | |
| **ЧИСЛЕННЫЕ МЕТОДЫ  «ИТЕРАЦИОННЫЕ МЕТОДЫ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАЧЕСКИХ УРАВНЕНИЙ»** | |
| *Лабораторная работа №3* | |
|  | |
|  | Работу выполнили студенты группы ПМИ-1-2015 3 курса механико-математического факультета  Толов Я.Ф., Заманов М.Р. |
|  | Проверил:  Профессор, доктор физико-математических наук  С. В. Русаков  “\_\_\_\_” 20\_\_ г. |
|  |  |
| Пермь 2017 | |

Оглавление

[*Задание* 3](#_Toc496731031)

[*Исходные данные* 5](#_Toc496731032)

[Решение 6](#_Toc496731033)

[Краткие выводы 8](#_Toc496731034)

[Анализ эффективности выбранных методов 8](#_Toc496731035)

[Сравнение фактического числа итерации с теоретическим 9](#_Toc496731036)

[Сравнение с прямыми методами 10](#_Toc496731037)

[Результаты тестирования 11](#_Toc496731038)

[Код на C++ 24](#_Toc496731039)

# *Задание*

1. Решить систему линейных алгебраических уравнений  итерационными методами с критерием остановки  (по невязке):
   * методом простой итерации;
   * градиентным методом наискорейшего спуска;
   * методом ПВР;
   * методом сопряженных градиентов.

В качества начального приближения выбирать вектор , который во всех вариантах принять равным .

1. Для каждого метода получить число итераций, необходимое для достижения требуемой точности (по невязке), выдавая (на печать) на каждом шаге (или через заданное число шагов)

- значение параметров итерационного метода;

- значение нормы невязки;

- оценку нормы матрицы перехода *q*;

- оценку погрешности приближенного решения.

3) Оценку нормы матрицы перехода осуществлять по формуле



4) В методе простой итерации значения итерационного параметра вычислять по формуле



1. В методе ПВР получить решение при оптимальном значении параметра , которое необходимо определить, варьируя параметр в диапазоне (0, 2) с шагом 0.1, и производя вычисления с критерием остановки(или по критерию минимальности нормы вектора невязки при заданном числе итераций).
2. Провести анализ эффективности рассматриваемых методов.
3. Сравнить решение, полученное итерационным методом, с решением полученным прямым методом.
4. Сравнить фактическое число итераций, необходимое для достижения заданной точности, с теоретической оценкой, вычислив число обусловленности.

# *Исходные данные*

Варианты исходных данных: 14 a) и 20 a).

Дана матрица А.

Вариант 14:

a)┌ ┐ b = [1,2,3,4]

│ 8.2 -3.3 1.0 -2.4│

│-3.3 6.4 -4.0 0.1│

│ 1.0 -4.0 6.9 1.4│

│-2.4 0.1 1.4 1.2│

└ ┘

Вариант 20:

a)┌ ┐ b = [1,2,3,4]

│ 4.3 1.9 -2.9 -1.2│

│ 1.9 4.4 2.0 4.3│

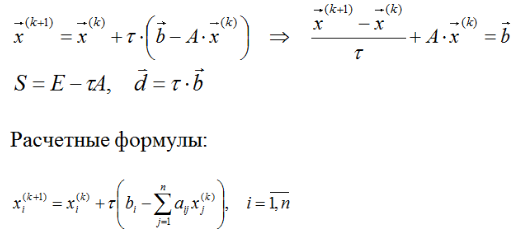
│-2.9 2.0 9.8 2.4│

│-1.2 4.3 2.4 9.3│

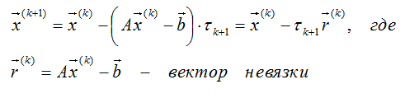
└ ┘

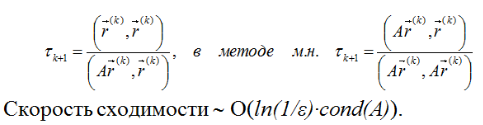
# Решение

1. Решение методом простой итерации:

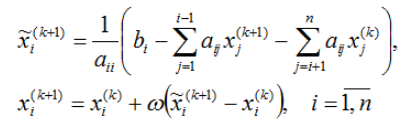


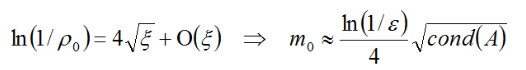
1. Методом наискорейшего градиентного спуска:



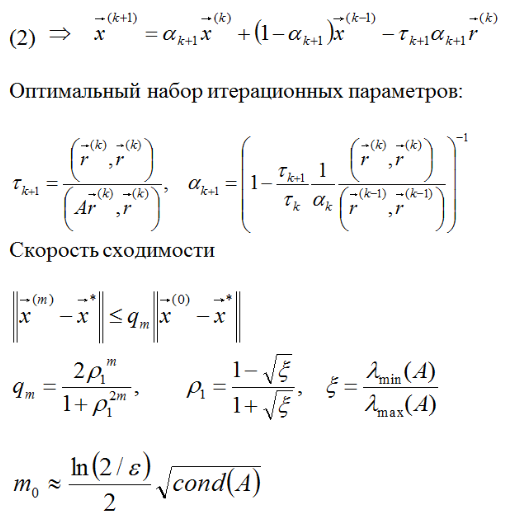


1. Методом ПВР:

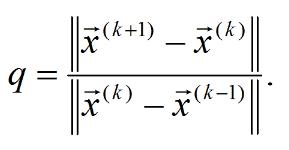




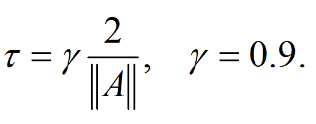
1. Метод сопряженных градиентов:



Оценка нормы матрицы перехода:



Значение итерационного параметра в методе простой итерации:



# Краткие выводы

Решая СЛАУ 4 разными итерационными методами, мы пришли к следующим выводам:

1. Разные методы решения несут за собой разное количество экономических, временных, трудовых затрат.

2. Так как все методы дают ответ с заданной точностью, рядовому пользователю может быть без разницы то, каким методом СЛАУ решать.

3. Для работы с такими массивами данных требуется большой объём памяти.

4. Для решения СЛАУ 4-мя представленными методами требуются знания в области высшей линейной алгебры.

# Анализ эффективности выбранных методов

Метод простых итерации является самым медленным методом среди представленных, также является одним из самых неточных методов.

Градиентный метод наискорейшего спуска имеет более высокую скорость сходимости по сравнению с методом простых итераций.

У ПВР-метода меньше итераций, чем у градиентного метода или метода простых итераций, но для его реализации требуются дополнительных действий для нахождения оптимального омега.

Метод сопряженных градиентов сходится быстрее всех остальных методов и является самым точным, в то же время этот метод является наиболее трудоёмким в плане вычислительных затрат.

# Сравнение фактического числа итерации с теоретическим

Для подсчёта теоретического числа итераций использовались следующие формулы:

Метод простых итераций: m0 = (log(1/ε)\*cond(A))/2

Градиентный метод наискорейшего спуска: m0 = log(1/ε)\*cond(A)

ПВР метод: m0 = (log(1/ε)\*sqrt(cond(A)))/4

Метод сопряженных градиентов: m0 = (log(2/ε)\*sqrt(cond(A)))/2

ЧИ – число итераций.

Фактическое число итераций в методе ПВР может быть больше из-за того, что оптимальное значение параметра определялось экспериментально.

Вариант 20

Cond(A) = 246.9

Метод: Фактическое ЧИ Теоретическое ЧИ Выполняется

Простых ит. 699 ≈1137 +

Градиентный 538 <=2274 +

ПВР 58 ≈36.18 +

Сопряж. Град. 4 ≈77.78 +

Фактическое число итераций в методе ПВР немного больше из-за того, что оптимальное значение параметра w определялось экспериментально.

Вариант 14

Cond(A) = 316.98

Метод: Фактическое ЧИ Теоретическое ЧИ Выполняется

Простых ит. 1028 ≈1459.69 +

Градиентный 795 <=2919.38 +

ПВР 35 ≈41 +

Сопряж. Град. 4 ≈88.13 +

# Сравнение с прямыми методами

K - Количество верных знаков по сравнению с LU разложением

Вариант 20

Метод: x1 x2 x3 x4 K

Через LU 7.9805 -9.3038 3.3683 4.8924

Простых ит. 7.97159 -9.29290 3.36471 4.88705 1-2

Градиентный 7.97433 -9.29625 3.36585 4.88873 1-2

ПВР 7.98056 -9.30403 3.36827 4.89239 2-4

Сопряж. Град. 7.98052 -9.30383 3.36831 4.89238 3-5

Вариант 20

Метод: x1 x2 x3 x4 K

Через LU 15.649 0.75437 -11.018 47.422

Простых ит. 15.64365 0.75454 -11.01359 47.40611 1-3

Градиентный 15.64343 0.75461 -11.01350 47.40561 1-3

ПВР 15.64913 0.75459 -11.01783 47.42313 2-3

Сопряж. Град. 15.64870 0.75437 -11.01764 47.42178 2-5

Решение, полученное итерационным методом, отличается от решения прямым методом тем, что прямое решение имеет определённое количество шагов выполнения программы и вычисляет более точно, в то время как итерационный метод имеет неопределённое количество шагов и получает решение с определённой точностью.

Наиболее точный ответ по сравнению с прямым методом даёт метод сопряженных градиентов, далее – ПВР, после идут методы наискорейшего спуска и простых итераций.

# Результаты тестирования

Входные данные: Вариант 20

a)┌ ┐ b = [1,2,3,4]

│ 4.3 1.9 -2.9 -1.2│

│ 1.9 4.4 2.0 4.3│

│-2.9 2.0 9.8 2.4│

│-1.2 4.3 2.4 9.3│

└ ┘

Метод простой итерации:

0 X = 1.66977 -1.33837 -0.88256 -1.00233 ||r|| = 5.00233 ||A\*x-b|| = 23.19849

1 X = 0.89542 -0.20904 1.37516 1.42542 ||r|| = 2.42775 ||A\*x-b|| = 10.88275

q = 1.01459 2 X = 1.23505 -1.01079 0.23626 0.31785 ||r|| = 1.13889 ||A\*x-b|| = 6.30537

q = 1.05401 3 X = 1.09653 -0.77411 0.81446 0.97772 ||r|| = 0.65986 ||A\*x-b|| = 2.60007

q = 1.06182 4 X = 1.23163 -1.03683 0.54236 0.72625 ||r|| = 0.27210 ||A\*x-b|| = 1.88058

q = 1.06262 5 X = 1.24401 -1.03533 0.70847 0.92305 ||r|| = 0.19681 ||A\*x-b|| = 1.19424

q = 1.06059 6 X = 1.32565 -1.16030 0.65823 0.88747 ||r|| = 0.12498 ||A\*x-b|| = 0.74685

q = 1.05772 7 X = 1.37568 -1.21744 0.71939 0.96563 ||r|| = 0.07816 ||A\*x-b|| = 0.84804

q = 1.05457 8 X = 1.44294 -1.30619 0.72534 0.98436 ||r|| = 0.08875 ||A\*x-b|| = 0.67776

q = 1.05153 9 X = 1.50174 -1.37712 0.75947 1.03174 ||r|| = 0.07093 ||A\*x-b|| = 0.74942

q = 1.04866 10 X = 1.56449 -1.45555 0.77938 1.06374 ||r|| = 0.07843 ||A\*x-b|| = 0.70097

q = 1.04602 11 X = 1.62466 -1.52890 0.80630 1.10277 ||r|| = 0.07336 ||A\*x-b|| = 0.71416

q = 1.04361 12 X = 1.68540 -1.60364 0.82942 1.13762 ||r|| = 0.07474 ||A\*x-b|| = 0.69684

q = 1.04141 13 X = 1.74507 -1.67657 0.85415 1.17400 ||r|| = 0.07293 ||A\*x-b|| = 0.69526

q = 1.03939 14 X = 1.80446 -1.74933 0.87775 1.20907 ||r|| = 0.07276 ||A\*x-b|| = 0.68597

q = 15-688 X = ||r|| = ||A\*x-b|| =

q = 1.00001 689 X = 7.97078 -9.29192 3.36438 4.88657 ||r|| = 0.00011 ||A\*x-b|| = 0.00108

q = 1.00001 690 X = 7.97088 -9.29203 3.36442 4.88663 ||r|| = 0.00011 ||A\*x-b|| = 0.00107

q = 1.00001 691 X = 7.97097 -9.29214 3.36446 4.88668 ||r|| = 0.00011 ||A\*x-b|| = 0.00106

q = 1.00001 692 X = 7.97106 -9.29226 3.36450 4.88674 ||r|| = 0.00011 ||A\*x-b|| = 0.00105

q = 1.00001 693 X = 7.97115 -9.29237 3.36453 4.88679 ||r|| = 0.00011 ||A\*x-b|| = 0.00104

q = 1.00001 694 X = 7.97124 -9.29247 3.36457 4.88685 ||r|| = 0.00011 ||A\*x-b|| = 0.00103

q = 1.00001 695 X = 7.97133 -9.29258 3.36460 4.88690 ||r|| = 0.00011 ||A\*x-b|| = 0.00102

q = 1.00001 696 X = 7.97141 -9.29269 3.36464 4.88695 ||r|| = 0.00011 ||A\*x-b|| = 0.00101

q = 1.00001 697 X = 7.97150 -9.29280 3.36467 4.88700 ||r|| = 0.00011 ||A\*x-b|| = 0.00100

q = 1.00001 698 X = 7.97159 -9.29290 3.36471 4.88705 ||r|| = 0.00010 ||A\*x-b|| = 0.00099

717

Ответ: VIM X = 7.97159 -9.29290 3.36471 4.88705

Градиентный метод наискорейшего спуска:

0 X = 1.45027 -0.24430 0.38986 0.63707 tau = 0.07035 ||r|| = 3.36293 ||A\*x-b|| = 3.19975

1 X = 1.06492 -0.67289 0.70393 0.62774 tau = 0.13395 ||r|| = 0.42859 ||A\*x-b|| = 1.16978

q = 0.92720 2 X = 1.16238 -0.90366 0.51236 0.75477 tau = 0.19728 ||r|| = 0.23077 ||A\*x-b|| = 1.34563

q = 1.07924 3 X = 1.17522 -0.96220 0.66907 0.87491 tau = 0.11646 ||r|| = 0.15671 ||A\*x-b|| = 1.09950

q = 1.09408 4 X = 1.30535 -1.14924 0.61393 0.84179 tau = 0.17012 ||r|| = 0.18704 ||A\*x-b|| = 1.20609

q = 1.06956 5 X = 1.34739 -1.18080 0.73585 0.98221 tau = 0.11643 ||r|| = 0.14042 ||A\*x-b|| = 1.05973

q = 1.08943 6 X = 1.47709 -1.36109 0.68488 0.94711 tau = 0.17013 ||r|| = 0.18029 ||A\*x-b|| = 1.17335

q = 1.05906 7 X = 1.51869 -1.39136 0.80374 1.08372 tau = 0.11643 ||r|| = 0.13661 ||A\*x-b|| = 1.03098

q = 1.07565 8 X = 1.64536 -1.56677 0.75429 1.04931 tau = 0.17013 ||r|| = 0.17540 ||A\*x-b|| = 1.14270

q = 1.05060 9 X = 1.68592 -1.59622 0.87010 1.18235 tau = 0.11643 ||r|| = 0.13304 ||A\*x-b|| = 1.00416

q = 1.06522 10 X = 1.80935 -1.76707 0.82196 1.14880 tau = 0.17013 ||r|| = 0.17084 ||A\*x-b|| = 1.11309

q = 1.04405 11 X = 1.84887 -1.79575 0.93477 1.27839 tau = 0.11643 ||r|| = 0.12959 ||A\*x-b|| = 0.97816

q = 1.05712 12 X = 1.96910 -1.96217 0.88787 1.24571 tau = 0.17013 ||r|| = 0.16642 ||A\*x-b|| = 1.08427

q = 1.03888 13 X = 2.00760 -1.99012 0.99777 1.37195 tau = 0.11643 ||r|| = 0.12624 ||A\*x-b|| = 0.95283

q = 1.05067 14 X = 2.12472 -2.15223 0.95208 1.34011 tau = 0.17013 ||r|| = 0.16211 ||A\*x-b|| = 1.05620

q = 15-527 X = tau = ||r|| = ||A\*x-b|| =

q = 1.00001 528 X = 7.97360 -9.29538 3.36546 4.88818 tau = 0.17013 ||r|| = 0.00019 ||A\*x-b|| = 0.00125

q = 1.00001 529 X = 7.97364 -9.29541 3.36558 4.88833 tau = 0.11643 ||r|| = 0.00015 ||A\*x-b|| = 0.00110

q = 1.00001 530 X = 7.97378 -9.29560 3.36553 4.88829 tau = 0.17013 ||r|| = 0.00019 ||A\*x-b|| = 0.00122

q = 1.00001 531 X = 7.97382 -9.29563 3.36565 4.88843 tau = 0.11643 ||r|| = 0.00014 ||A\*x-b|| = 0.00107

q = 1.00001 532 X = 7.97395 -9.29581 3.36560 4.88840 tau = 0.17013 ||r|| = 0.00018 ||A\*x-b|| = 0.00118

q = 1.00001 533 X = 7.97399 -9.29584 3.36572 4.88853 tau = 0.11643 ||r|| = 0.00014 ||A\*x-b|| = 0.00104

q = 1.00001 534 X = 7.97412 -9.29602 3.36567 4.88850 tau = 0.17013 ||r|| = 0.00018 ||A\*x-b|| = 0.00115

q = 1.00001 535 X = 7.97416 -9.29605 3.36579 4.88863 tau = 0.11643 ||r|| = 0.00013 ||A\*x-b|| = 0.00101

q = 1.00001 536 X = 7.97429 -9.29622 3.36574 4.88860 tau = 0.17013 ||r|| = 0.00017 ||A\*x-b|| = 0.00112

q = 1.00001 537 X = 7.97433 -9.29625 3.36585 4.88873 tau = 0.11643 ||r|| = 0.00013 ||A\*x-b|| = 0.00099

506

Ответ: SDM X = 7.97433 -9.29625 3.36585 4.88873

Метод ПВР: Используется omega = 1.7

Поиск оптимального омега:

w = 0.10000 k = 3475.00000

w = 0.20000 k = 1644.00000

w = 0.30000 k = 1034.00000

w = 0.40000 k = 728.00000

w = 0.50000 k = 544.00000

w = 0.60000 k = 423.00000

w = 0.70000 k = 338.00000

w = 0.80000 k = 273.00000

w = 0.90000 k = 222.00000

w = 1.00000 k = 181.00000

w = 1.10000 k = 146.00000

w = 1.20000 k = 116.00000

w = 1.30000 k = 87.00000

w = 1.40000 k = 73.00000

w = 1.50000 k = 58.00000

w = 1.60000 k = 50.00000

w = 1.70000 k = 30.00000

w = 1.80000 k = 47.00000

w = 1.90000 k = 99.00000

0 X = 3.53023 -12.18242 2.75758 7.07142 ||r|| = 14.18242 ||A\*x-b|| = 25.44929

1 X = 13.59158 -14.55610 7.53355 6.89893 ||r|| = 10.06135 ||A\*x-b|| = 18.85846

q = 1.65807 2 X = 13.72553 -16.39680 4.96818 9.62134 ||r|| = 2.72241 ||A\*x-b|| = 10.42524

q = 1.03739 3 X = 13.36476 -17.38406 5.79156 8.05130 ||r|| = 1.57004 ||A\*x-b|| = 6.89310

q = 0.93095 4 X = 14.55805 -15.59680 5.84907 7.98203 ||r|| = 1.78726 ||A\*x-b|| = 5.42496

q = 1.02760 5 X = 12.41330 -15.20279 4.62202 7.78868 ||r|| = 2.14475 ||A\*x-b|| = 5.41557

q = 0.88732 6 X = 12.12010 -13.99396 4.99456 6.74610 ||r|| = 1.20883 ||A\*x-b|| = 3.99247

q = 0.94484 7 X = 11.34982 -12.83046 4.37667 6.66348 ||r|| = 1.16350 ||A\*x-b|| = 2.73782

q = 0.94529 8 X = 10.26742 -12.23561 4.09270 6.14090 ||r|| = 1.08240 ||A\*x-b|| = 2.40008

q = 0.91747 9 X = 10.00478 -11.37157 4.07715 5.77677 ||r|| = 0.86404 ||A\*x-b|| = 1.65870

q = 0.96505 10 X = 9.34901 -10.87803 3.73852 5.64844 ||r|| = 0.65576 ||A\*x-b|| = 1.67446

q = 0.94972 11 X = 8.98819 -10.48377 3.71069 5.36143 ||r|| = 0.39426 ||A\*x-b|| = 1.09907

q = 0.96276 12 X = 8.77654 -10.10605 3.61214 5.26225 ||r|| = 0.37772 ||A\*x-b|| = 0.74773

q = 0.97878 13 X = 8.48093 -9.91252 3.50656 5.16103 ||r|| = 0.29561 ||A\*x-b|| = 0.66897

q = 0.97385 14 X = 8.37341 -9.71932 3.50149 5.05867 ||r|| = 0.19320 ||A\*x-b|| = 0.39199

q = 0.98695 15 X = 8.24918 -9.58938 3.44008 5.02788 ||r|| = 0.12994 ||A\*x-b|| = 0.32173

q = 0.98875 16 X = 8.15351 -9.51150 3.42074 4.97571 ||r|| = 0.09567 ||A\*x-b|| = 0.20321

q = 0.99037 17 X = 8.11506 -9.43618 3.41052 4.94908 ||r|| = 0.07531 ||A\*x-b|| = 0.13662

q = 0.99559 18 X = 8.06105 -9.39711 3.38804 4.93502 ||r|| = 0.05401 ||A\*x-b|| = 0.12446

q = 0.99515 19 X = 8.03706 -9.36613 3.38681 4.91579 ||r|| = 0.03098 ||A\*x-b|| = 0.07367

q = 0.99726 20 X = 8.02005 -9.34241 3.37889 4.91036 ||r|| = 0.02371 ||A\*x-b|| = 0.04498

q = 0.99836 21 X = 8.00249 -9.33099 3.37390 4.90351 ||r|| = 0.01756 ||A\*x-b|| = 0.03657

q = 0.99835 22 X = 7.99723 -9.31989 3.37374 4.89850 ||r|| = 0.01109 ||A\*x-b|| = 0.01921

q = 0.99935 23 X = 7.99002 -9.31392 3.37024 4.89727 ||r|| = 0.00720 ||A\*x-b|| = 0.01707

q = 0.99938 24 X = 7.98598 -9.31039 3.36995 4.89459 ||r|| = 0.00404 ||A\*x-b|| = 0.01023

q = 0.99958 25 X = 7.98455 -9.30714 3.36942 4.89383 ||r|| = 0.00325 ||A\*x-b|| = 0.00608

q = 0.99985 26 X = 7.98214 -9.30597 3.36849 4.89332 ||r|| = 0.00240 ||A\*x-b|| = 0.00496

q = 0.99980 27 X = 7.98164 -9.30486 3.36872 4.89260 ||r|| = 0.00110 ||A\*x-b|| = 0.00277

q = 0.99993 28 X = 7.98108 -9.30420 3.36835 4.89262 ||r|| = 0.00067 ||A\*x-b|| = 0.00145

q = 0.99996 29 X = 7.98056 -9.30403 3.36827 4.89239 ||r|| = 0.00052 ||A\*x-b|| = 0.00089

93

Ответ: SOR X = 7.98056 -9.30403 3.36827 4.89239

Метод сопряжённых градиентов:+

0 X = 1.45027 -0.24430 0.38986 0.63707 ||r|| = 3.36293 ||A\*x-b|| = 3.19975

1 X = 1.06555 -0.69904 0.68147 0.59475 ||r|| = 0.45474 ||A\*x-b|| = 1.11781

q = 1.03510 2 X = 1.11461 -1.15920 0.53692 1.00043 ||r|| = 0.46016 ||A\*x-b|| = 1.16727

q = 4.68296 3 X = 7.98052 -9.30383 3.36831 4.89238 ||r|| = 8.14463 ||A\*x-b|| = 0.00000

Ответ: CGM X = 7.98052 -9.30383 3.36831 4.89238

Входные данные: Вариант 14

a)┌ ┐ b = [1,2,3,4]

│ 8.2 -3.3 1.0 -2.4│

│-3.3 6.4 -4.0 0.1│

│ 1.0 -4.0 6.9 1.4│

│-2.4 0.1 1.4 1.2│

└ ┘

Метод простой итерации:

0 X = 1.72483 2.49530 1.03087 3.66174 ||r|| = 1.96913 ||A\*x-b|| = 4.52067

1 X = 2.06891 1.94918 0.91211 3.90973 ||r|| = 0.54612 ||A\*x-b|| = 3.03937

q = 0.99744 2 X = 1.94067 1.90208 0.54494 4.24820 ||r|| = 0.36717 ||A\*x-b|| = 2.60664

q = 1.20334 3 X = 2.06318 1.65876 0.41931 4.56310 ||r|| = 0.31490 ||A\*x-b|| = 2.72300

q = 1.07872 4 X = 2.07381 1.58789 0.21277 4.89205 ||r|| = 0.32895 ||A\*x-b|| = 2.65001

q = 1.10941 5 X = 2.16598 1.47227 0.08722 5.21219 ||r|| = 0.32014 ||A\*x-b|| = 2.67438

q = 1.07357 6 X = 2.22874 1.41826 -0.05482 5.53527 ||r|| = 0.32308 ||A\*x-b|| = 2.64157

q = 1.07609 7 X = 2.31862 1.35848 -0.16679 5.85439 ||r|| = 0.31912 ||A\*x-b|| = 2.63709

q = 1.06307 8 X = 2.40169 1.32279 -0.27914 6.17296 ||r|| = 0.31857 ||A\*x-b|| = 2.61501

q = 1.06041 9 X = 2.49418 1.28967 -0.37900 6.48887 ||r|| = 0.31591 ||A\*x-b|| = 2.60102

q = 1.05425 10 X = 2.58550 1.26696 -0.47622 6.80308 ||r|| = 0.31422 ||A\*x-b|| = 2.58151

q = 1.05118 11 X = 2.68015 1.24744 -0.56755 7.11494 ||r|| = 0.31186 ||A\*x-b|| = 2.56426

q = 1.04749 12 X = 2.77471 1.23284 -0.65637 7.42472 ||r|| = 0.30978 ||A\*x-b|| = 2.54527

q = 1.04488 13 X = 2.87032 1.22057 -0.74202 7.73220 ||r|| = 0.30748 ||A\*x-b|| = 2.52690

q = 1.04229 14 X = 2.96582 1.21080 -0.82576 8.03747 ||r|| = 0.30526 ||A\*x-b|| = 2.50800

q = 15-1017 X = ||r|| = ||A\*x-b|| =

q = 1.00000 1018 X = 15.64329 0.75456 -11.01330 47.40499 ||r|| = 0.00013 ||A\*x-b|| = 0.00107

q = 1.00000 1019 X = 15.64333 0.75456 -11.01333 47.40512 ||r|| = 0.00013 ||A\*x-b|| = 0.00106

q = 1.00000 1020 X = 15.64337 0.75455 -11.01337 47.40524 ||r|| = 0.00013 ||A\*x-b|| = 0.00105

q = 1.00000 1021 X = 15.64341 0.75455 -11.01340 47.40537 ||r|| = 0.00013 ||A\*x-b|| = 0.00105

q = 1.00000 1022 X = 15.64345 0.75455 -11.01343 47.40550 ||r|| = 0.00013 ||A\*x-b|| = 0.00104

q = 1.00000 1023 X = 15.64349 0.75455 -11.01346 47.40562 ||r|| = 0.00013 ||A\*x-b|| = 0.00103

q = 1.00000 1024 X = 15.64353 0.75455 -11.01350 47.40575 ||r|| = 0.00012 ||A\*x-b|| = 0.00102

q = 1.00000 1025 X = 15.64357 0.75455 -11.01353 47.40587 ||r|| = 0.00012 ||A\*x-b|| = 0.00101

q = 1.00000 1026 X = 15.64361 0.75455 -11.01356 47.40599 ||r|| = 0.00012 ||A\*x-b|| = 0.00101

q = 1.00000 1027 X = 15.64365 0.75454 -11.01359 47.40611 ||r|| = 0.00012 ||A\*x-b|| = 0.00100

837

Ответ: VIM X = 15.64365 0.75454 -11.01359 47.40611

Градиентный метод наискорейшего спуска:

0 X = 1.74499 2.50908 0.97610 3.65234 tau = 0.12417 ||r|| = 2.02390 ||A\*x-b|| = 4.76044

1 X = 2.14630 1.81704 0.89512 3.97037 tau = 0.14537 ||r|| = 0.69204 ||A\*x-b|| = 3.61299

q = 1.01794 2 X = 1.85188 1.91228 0.35506 4.41160 tau = 0.14948 ||r|| = 0.54006 ||A\*x-b|| = 3.14830

q = 1.29460 3 X = 2.20900 1.43544 0.38097 4.78453 tau = 0.15146 ||r|| = 0.47684 ||A\*x-b|| = 2.88326

q = 1.02960 4 X = 2.01270 1.61225 -0.04926 5.22847 tau = 0.15397 ||r|| = 0.44393 ||A\*x-b|| = 2.46406

q = 1.19080 5 X = 2.38476 1.24288 0.02108 5.60826 tau = 0.15414 ||r|| = 0.37980 ||A\*x-b|| = 2.83971

q = 1.01927 6 X = 2.22704 1.46649 -0.35352 6.04961 tau = 0.15542 ||r|| = 0.44135 ||A\*x-b|| = 2.45025

q = 1.14587 7 X = 2.60648 1.14821 -0.25925 6.42648 tau = 0.15486 ||r|| = 0.37944 ||A\*x-b|| = 2.79191

q = 1.01321 8 X = 2.46609 1.39328 -0.60537 6.86139 tau = 0.15577 ||r|| = 0.43491 ||A\*x-b|| = 2.44859

q = 1.12019 9 X = 2.84568 1.10082 -0.50024 7.23238 tau = 0.15502 ||r|| = 0.37959 ||A\*x-b|| = 2.74103

q = 1.00999 10 X = 2.71416 1.35350 -0.83004 7.65958 tau = 0.15585 ||r|| = 0.42720 ||A\*x-b|| = 2.42352

q = 1.10299 11 X = 3.08995 1.07566 -0.72092 8.02385 tau = 0.15506 ||r|| = 0.37580 ||A\*x-b|| = 2.68899

q = 1.00816 12 X = 2.96373 1.32919 -1.03989 8.44299 tau = 0.15587 ||r|| = 0.41914 ||A\*x-b|| = 2.38679

q = 1.09033 13 X = 3.33385 1.06086 -0.93009 8.80032 tau = 0.15507 ||r|| = 0.37012 ||A\*x-b|| = 2.63690

q = 1.00701 14 X = 3.21135 1.31211 -1.24079 9.21135 tau = 0.15588 ||r|| = 0.41103 ||A\*x-b|| = 2.34490

q = 15-784 X = tau = ||r|| = ||A\*x-b|| =

q = 1.00001 785 X = 15.64305 0.75451 -11.01302 47.40408 tau = 0.15507 ||r|| = 0.00017 ||A\*x-b|| = 0.00121

q = 1.00000 786 X = 15.64300 0.75462 -11.01316 47.40427 tau = 0.15588 ||r|| = 0.00019 ||A\*x-b|| = 0.00108

q = 1.00001 787 X = 15.64317 0.75451 -11.01311 47.40443 tau = 0.15507 ||r|| = 0.00017 ||A\*x-b|| = 0.00118

q = 1.00000 788 X = 15.64311 0.75462 -11.01325 47.40461 tau = 0.15588 ||r|| = 0.00018 ||A\*x-b|| = 0.00106

q = 1.00001 789 X = 15.64327 0.75450 -11.01320 47.40477 tau = 0.15507 ||r|| = 0.00016 ||A\*x-b|| = 0.00116

q = 1.00000 790 X = 15.64322 0.75461 -11.01333 47.40495 tau = 0.15588 ||r|| = 0.00018 ||A\*x-b|| = 0.00103

q = 1.00001 791 X = 15.64338 0.75450 -11.01329 47.40511 tau = 0.15507 ||r|| = 0.00016 ||A\*x-b|| = 0.00114

q = 1.00000 792 X = 15.64333 0.75461 -11.01342 47.40528 tau = 0.15588 ||r|| = 0.00018 ||A\*x-b|| = 0.00101

q = 1.00001 793 X = 15.64349 0.75450 -11.01337 47.40543 tau = 0.15507 ||r|| = 0.00016 ||A\*x-b|| = 0.00112

q = 1.00000 794 X = 15.64343 0.75461 -11.01350 47.40561 tau = 0.15588 ||r|| = 0.00017 ||A\*x-b|| = 0.00099 607

Ответ:SDM X = 15.64343 0.75461 -11.01350 47.40561

Метод ПВР: Используется omega = 1.5

w = 0.10000 k = 2663.00000

w = 0.20000 k = 1269.00000

w = 0.30000 k = 804.00000

w = 0.40000 k = 571.00000

w = 0.50000 k = 431.00000

w = 0.60000 k = 337.00000

w = 0.70000 k = 270.00000

w = 0.80000 k = 219.00000

w = 0.90000 k = 179.00000

w = 1.00000 k = 147.00000

w = 1.10000 k = 120.00000

w = 1.20000 k = 97.00000

w = 1.30000 k = 77.00000

w = 1.40000 k = 58.00000

w = 1.50000 k = 35.00000

w = 1.60000 k = 38.00000

w = 1.70000 k = 50.00000

w = 1.80000 k = 74.00000

w = 1.90000 k = 141.00000

0 X = 2.24390 4.17942 0.82542 8.26676 ||r|| = 4.26676 ||A\*x-b|| = 15.40689

1 X = 5.43801 3.02984 -1.04397 20.01049 ||r|| = 11.74373 ||A\*x-b|| = 15.47592

q = 1.21026 2 X = 8.64643 4.34876 -3.27681 26.94014 ||r|| = 6.92965 ||A\*x-b|| = 13.10006

q = 1.28563 3 X = 11.21370 3.11942 -5.94808 36.29025 ||r|| = 9.35012 ||A\*x-b|| = 12.38645

q = 1.12694 4 X = 13.78162 3.14431 -7.91145 42.36660 ||r|| = 6.07634 ||A\*x-b|| = 8.52671

q = 1.08801 5 X = 15.43150 2.32562 -9.84630 47.67617 ||r|| = 5.30958 ||A\*x-b|| = 6.40542

q = 1.05037 6 X = 16.75945 1.86592 -11.10354 51.03316 ||r|| = 3.35698 ||A\*x-b|| = 4.15320

q = 1.01670 7 X = 17.44633 1.36482 -12.04446 52.95580 ||r|| = 1.92265 ||A\*x-b|| = 2.63539

q = 1.00322 8 X = 17.77439 0.95236 -12.53630 53.75927 ||r|| = 0.80347 ||A\*x-b|| = 0.96075

q = 0.98597 9 X = 17.76430 0.68832 -12.72688 53.57794 ||r|| = 0.26404 ||A\*x-b|| = 1.08189

q = 0.98087 10 X = 17.54001 0.47887 -12.68209 52.88310 ||r|| = 0.69484 ||A\*x-b|| = 1.64628

q = 0.97584 11 X = 17.19871 0.39236 -12.47493 51.81046 ||r|| = 1.07263 ||A\*x-b|| = 1.91454

q = 0.97618 12 X = 16.80179 0.35358 -12.19039 50.65293 ||r|| = 1.15754 ||A\*x-b|| = 1.85043

q = 0.97813 13 X = 16.41816 0.37754 -11.87218 49.52437 ||r|| = 1.12855 ||A\*x-b|| = 1.65238

q = 0.98179 14 X = 16.07559 0.42856 -11.57097 48.54501 ||r|| = 0.97937 ||A\*x-b|| = 1.32664

q = 0.98636 15 X = 15.80056 0.49780 -11.30800 47.76407 ||r|| = 0.78093 ||A\*x-b|| = 0.98005

q = 0.99079 16 X = 15.59738 0.57138 -11.10015 47.19724 ||r|| = 0.56683 ||A\*x-b|| = 0.69421

q = 0.99498 17 X = 15.46482 0.63958 -10.95026 46.83637 ||r|| = 0.36087 ||A\*x-b|| = 0.45591

q = 0.99829 18 X = 15.39364 0.69829 -10.85531 46.65035 ||r|| = 0.18602 ||A\*x-b|| = 0.24367

q = 1.00079 19 X = 15.37139 0.74351 -10.80756 46.60380 ||r|| = 0.04775 ||A\*x-b|| = 0.07648

q = 1.00238 20 X = 15.38484 0.77562 -10.79660 46.65585 ||r|| = 0.05204 ||A\*x-b|| = 0.17448

q = 1.00322 21 X = 15.42101 0.79511 -10.81193 46.77003 ||r|| = 0.11419 ||A\*x-b|| = 0.23156

q = 1.00345 22 X = 15.46902 0.80423 -10.84342 46.91451 ||r|| = 0.14447 ||A\*x-b|| = 0.24624

q = 1.00325 23 X = 15.52007 0.80529 -10.88274 47.06477 ||r|| = 0.15027 ||A\*x-b|| = 0.23108

q = 1.00278 24 X = 15.56798 0.80078 -10.92330 47.20355 ||r|| = 0.13878 ||A\*x-b|| = 0.19699

q = 1.00217 25 X = 15.60882 0.79296 -10.96055 47.32025 ||r|| = 0.11669 ||A\*x-b|| = 0.15360

q = 1.00154 26 X = 15.64066 0.78367 -10.99173 47.40998 ||r|| = 0.08973 ||A\*x-b|| = 0.10919

q = 1.00095 27 X = 15.66312 0.77435 -11.01557 47.47215 ||r|| = 0.06217 ||A\*x-b|| = 0.07704

q = 1.00046 28 X = 15.67690 0.76597 -11.03198 47.50922 ||r|| = 0.03707 ||A\*x-b|| = 0.04727

q = 1.00008 29 X = 15.68337 0.75908 -11.04166 47.52543 ||r|| = 0.01621 ||A\*x-b|| = 0.02219

q = 0.99981 30 X = 15.68420 0.75392 -11.04577 47.52580 ||r|| = 0.00516 ||A\*x-b|| = 0.01486

q = 0.99965 31 X = 15.68112 0.75046 -11.04567 47.51536 ||r|| = 0.01043 ||A\*x-b|| = 0.02616

q = 0.99957 32 X = 15.67570 0.74851 -11.04273 47.49866 ||r|| = 0.01670 ||A\*x-b|| = 0.03115

q = 0.99957 33 X = 15.66924 0.74778 -11.03815 47.47941 ||r|| = 0.01924 ||A\*x-b|| = 0.03134

q = 0.99961 34 X = 15.66274 0.74797 -11.03293 47.46042 ||r|| = 0.01900 ||A\*x-b|| = 0.02826

q = 0.99967 35 X = 15.65688 0.74875 -11.02782 47.44354 ||r|| = 0.01688 ||A\*x-b|| = 0.02325

q = 0.99975 36 X = 15.65206 0.74986 -11.02329 47.42984 ||r|| = 0.01370 ||A\*x-b|| = 0.01747

q = 0.99983 37 X = 15.64844 0.75108 -11.01964 47.41970 ||r|| = 0.01014 ||A\*x-b|| = 0.01242

q = 0.99990 38 X = 15.64601 0.75224 -11.01695 47.41303 ||r|| = 0.00668 ||A\*x-b|| = 0.00835

q = 0.99996 39 X = 15.64463 0.75325 -11.01520 47.40939 ||r|| = 0.00364 ||A\*x-b|| = 0.00473

q = 1.00001 40 X = 15.64411 0.75406 -11.01425 47.40819 ||r|| = 0.00120 ||A\*x-b|| = 0.00178

q = 1.00003 41 X = 15.64423 0.75463 -11.01396 47.40877 ||r|| = 0.00058 ||A\*x-b|| = 0.00260

q = 1.00005 42 X = 15.64477 0.75500 -11.01414 47.41050 ||r|| = 0.00173 ||A\*x-b|| = 0.00371

q = 1.00006 43 X = 15.64554 0.75518 -11.01462 47.41283 ||r|| = 0.00233 ||A\*x-b|| = 0.00409

q = 1.00005 44 X = 15.64639 0.75522 -11.01526 47.41534 ||r|| = 0.00250 ||A\*x-b|| = 0.00392

q = 1.00005 45 X = 15.64720 0.75516 -11.01593 47.41770 ||r|| = 0.00236 ||A\*x-b|| = 0.00341

q = 1.00004 46 X = 15.64791 0.75504 -11.01657 47.41972 ||r|| = 0.00202 ||A\*x-b|| = 0.00271

q = 1.00003 47 X = 15.64847 0.75489 -11.01711 47.42131 ||r|| = 0.00159 ||A\*x-b|| = 0.00195

q = 1.00002 48 X = 15.64887 0.75473 -11.01753 47.42244 ||r|| = 0.00113 ||A\*x-b|| = 0.00139

q = 1.00001 49 X = 15.64913 0.75459 -11.01783 47.42313 ||r|| = 0.00070 ||A\*x-b|| = 0.00088

33

Ответ: SOR X = 15.64913 0.75459 -11.01783 47.42313

Метод сопряжённых градиентов:+

0 X = 1.74499 2.50908 0.97610 3.65234 ||r|| = 2.02390 ||A\*x-b|| = 4.76044

1 X = 2.31294 1.79044 0.58911 3.96606 ||r|| = 0.71864 ||A\*x-b|| = 3.78799

q = 2.12081 2 X = 2.02240 -0.07300 -1.48809 7.02342 ||r|| = 3.05735 ||A\*x-b|| = 2.51970

q = 9.63797 3 X = 15.64870 0.75437 -11.01764 47.42178 ||r|| = 40.39836 ||A\*x-b|| = 0.00000

Ответ: CGM X = 15.64870 0.75437 -11.01764 47.42178

# Код на C++

#include <iostream>

#include <iomanip>

#include "Matrix.h"

using namespace std;

double\*\* init(int n, int m, bool fillByZero);

double\* ValueIterationMethod(Matrix C, double\* X);

void copy(double\* A, double\* B, int n);

double Max(double\* E, int n);

double Norm(double\*\* M, int n);

double scalar(double\* A, double\* B, int n);

double GetQ(double\* X\_1, double\* X0, double\* X, int n);

double\* VIM(Matrix C, double\* X); // - методом простой итерации

double\* SOR(Matrix C, double\* X); //- методом ПВР

double\* CGM(Matrix C, double\* X); //- методом сопряженных градиентов

double\* SDM(Matrix C, double \* X); //- градиентным методом наискорейшего спуска

const double c\_delta = 0.001;

double teta;

const bool OUT\_VIM = true;

const bool OUT\_SOR = true;

const bool OUT\_CGM = true;

const bool OUT\_SDM = true;

double w = 1.7;

double K = 0;

class Matrix

{

private:

public:

double\*\* value;

double\* b;

int n;

Matrix(int n, bool fillByZero = false)

{

this->n = n;

value = new double\*[n];

for (int i = 0; i < n; i++)

value[i] = new double[n];

b = new double[n];

if (fillByZero)

{

for (int i = 0; i < n; i++, b[i] = 0)

for (int j = 0; j < n; j++)

value[i][j] = 0;

}

}

Matrix(const char\* fileName)

{

std::ifstream fin(fileName);

fin >> n;

value = new double\*[n];

for (int i = 0; i < n; i++)

value[i] = new double[n];

b = new double[n];

for (int i = 0; i < n; i++)

for (int j = 0; j < n; j++)

fin >> value[i][j];

for (int i = 0; i < n; i++)

fin >> b[i];

fin.close();

}

void Print()

{

using std::cout;

using std::endl;

using std::setprecision;

using std::ios;

using std::setw;

using std::fixed;

cout.setf(ios::left);

for (int i = 0; i < n; i++, cout << endl)

for (int j = 0; j < n; j++)

{

if (value[i][j] > 0)

cout << " ";

cout << fixed << setprecision(1) << value[i][j] << " ";

}

cout.unsetf(ios::left);

}

void Delete()

{

for (int i = 0; i < n; i++)

delete[] value[i];

delete[] value;

delete[] b;

}

};

int main()

{

Matrix A("input.txt");

teta = 0.9 \* (2 / Norm(A.value, A.n));

cout << "A:\n";

A.Print();

cout << "b = ";

for (int i = 0; i < A.n; i++)

cout << A.b[i] << " ";

cout << endl;

double\* X = new double[A.n];

for (int i = 0; i < A.n; i++)

X[i] = A.b[i];

X = VIM(A, X);

cout << "VIM X = ";

for (int i = 0; i < A.n; i++)

cout << fixed << setprecision(5) << X[i] << " ";

cout << endl;

for (int i = 0; i < A.n; i++)

X[i] = A.b[i];

X = SDM(A, X);

cout << "SDM X = ";

for (int i = 0; i < A.n; i++)

cout << fixed << setprecision(5) << X[i] << " ";

cout << endl;

/\*for (w = 0.1; w <= 2; w += 0.1)

{

K = 0;\*/

for (int i = 0; i < A.n; i++)

X[i] = A.b[i];

X = SOR(A, X);

/\*cout << "w = " << w << " k = " << K << endl;

}\*/

cout << "SOR X = ";

for (int i = 0; i < A.n; i++)

cout << fixed << setprecision(5) << X[i] << " ";

cout << endl;

for (int i = 0; i < A.n; i++)

X[i] = A.b[i];

X = CGM(A, X);

cout << "CGM X = ";

for (int i = 0; i < A.n; i++)

cout << fixed << setprecision(5) << X[i] << " ";

cout << endl;

delete[] X;

A.Delete();

cin.get();

cin.get();

return 0;

}

double\* ValueIterationMethod(Matrix C, double\* X)

{

int n = C.n;

double\* X0 = new double[n];

copy(X0, X, n);

double\* E = new double[n];

cout << endl;

double delta;

do

{

for (int i = 0; i < n; i++)

{

X[i] = 0;

for (int j = 0; j < n; j++)

{

X[i] += C.value[i][j] \* X0[j];

}

X[i] += C.b[i];

E[i] = abs(X[i] - X0[i]);

}

delta = Max(E,n);

cout << "X = ";

for (int i = 0; i < n; i++)

cout << setw(5) << setprecision(5) << X[i] << " ";

cout << setw(15) << " E = ";

for (int i = 0; i < n; i++)

cout << setw(5) << setprecision(5) << E[i] << " ";

cout << " delta = " << delta << endl;

copy(X0, X, n);

} while (delta > c\_delta);

delete[] E;

delete[] X0;

return X;

}

double\* VIM(Matrix A, double\* X)

{

int n = A.n;

double\* X0 = new double[n];

copy(X0, X, n);

double\* X\_1 = new double[n];

double\* r = new double[n];

double\* E = new double[n];

double delta;

int k = 0;

do

{

for (int i = 0; i < n; i++)

{

X[i] = X0[i];

double sum = 0;

for (int j = 0; j < n; j++)

{

sum += A.value[i][j] \* X0[j];

}

X[i] += teta\*(A.b[i] - sum);

}

for (int i = 0; i < n; i++)

{

r[i] = fabs(X[i] - X0[i]);

E[i] = -A.b[i];

for (int j = 0; j < n; j++)

E[i] += A.value[i][j] \* X[j];

}

delta = Max(E, n);

if (OUT\_VIM)

{

if (k >= 2)

{

cout << "q = " << GetQ(X\_1, X0, X, n) << " ";

}

cout << k++ << " X = ";

for (int i = 0; i < n; i++)

{

if (X[i] > 0)

cout << " ";

cout << fixed << setprecision(5) << X[i] << " ";

}

cout << "||r|| = " << Max(r,n);

cout << " ||A\*x-b|| = " << delta << endl;

}

copy(X\_1, X0, n);

copy(X0, X, n);

} while (delta > c\_delta);

delete[] E;

delete[] X0;

delete[] X\_1;

return X;

}

double\* SOR(Matrix A, double\* X)

{

int k = 0;

int n = A.n;

double\* X0 = new double[n];

copy(X0, X, n);

double\* X\_1 = new double[n];

double\* L = new double[n];

double\* r = new double[n];

double\* E = new double[n];

double delta;

do

{

K++;

for (int i = 0; i < n; i++)

{

L[i] = A.b[i];

for (int j = 0; j < i; j++)

L[i] -= A.value[i][j] \* X[j];

for (int j = i + 1; j < n; j++)

L[i] -= A.value[i][j] \* X0[j];

L[i] /= A.value[i][i];

X[i] = X0[i] + w\*(L[i] - X0[i]);

}

for (int i = 0; i < n; i++)

{

r[i] = fabs(X[i] - X0[i]);

E[i] = -A.b[i];

for (int j = 0; j < n; j++)

E[i] += A.value[i][j] \* X[j];

}

delta = Max(E, n);

if (OUT\_SOR)

{

if (k >= 2)

{

cout << "q = " << fixed << GetQ(X\_1, X0, X, n) << " ";

}

cout << k++ << " X = ";

for (int i = 0; i < n; i++)

{

if (X[i] > 0)

cout << " ";

cout << fixed << setprecision(5) << X[i] << " ";

}

cout << "||r|| = " << Max(r, n);

cout << " ||A\*x-b|| = " << delta << endl;

}

copy(X\_1, X0, n);

copy(X0, X, n);

} while (delta > c\_delta);

delete[] E;

delete[] X0;

delete[] X\_1;

return X;

}

double\* CGM(Matrix A, double\* X)

{

int n = A.n;

int k = 0;

double\* X0 = new double[n];

copy(X0, X, n);

double\* X\_1 = new double[n];

double\* E = new double[n];

double\* \_r = new double[n];

double\* r = new double[n];

double\* z = new double[n];

double\* Az = new double[n];

for (int i = 0; i < n; i++)

{

\_r[i] = A.b[i];

for (int j = 0; j < n; j++)

\_r[i] -= A.value[i][j] \* X0[j];

z[i] = \_r[i];

}

double delta;

double a = 1;

double b = 1;

do

{

for (int j = 0; j < n; j++)

{

Az[j] = A.value[j][0] \* z[0];

for (int t = 1; t < n; t++)

Az[j] += A.value[j][t] \* z[t];

}

a = scalar(\_r, \_r, n) / scalar(Az, z, n);

b = 1 / scalar(\_r, \_r, n);

for (int i = 0; i < n; i++)

{

X[i] = X0[i] + a\*z[i];

\_r[i] -= a\*Az[i];

}

for (int i = 0; i < n; i++)

{

r[i] = fabs(X[i] - X0[i]);

E[i] = -A.b[i];

for (int j = 0; j < n; j++)

E[i] += A.value[i][j] \* X[j];

}

b \*= scalar(\_r, \_r, n);

for (int i = 0; i < n; i++)

{

z[i] = \_r[i] + b\*z[i];

}

delta = Max(E, n);

if (OUT\_CGM)

{

if (k >= 2)

{

cout << "q = " << GetQ(X\_1, X0, X, n) << " ";

}

cout << k++ << " X = ";

for (int i = 0; i < n; i++)

{

if (X[i] > 0)

cout << " ";

cout << fixed << setprecision(5) << X[i] << " ";

}

cout << " ||r|| = " << Max(r, n);

cout << " ||A\*x-b|| = " << delta << endl;

}

copy(X\_1, X0, n);

copy(X0, X, n);

} while (delta > c\_delta);

delete[] E;

delete[] X0;

delete[] X\_1;

delete[] \_r;

delete[] z;

delete[] Az;

return X;

}

double\* SDM(Matrix A, double \* X)

{

int n = A.n;

int k = 0;

double\* X0 = new double[n];

copy(X0, X, n);

double\* X\_1 = new double[n];

double\* E = new double[n];

double\* \_r = new double[n];

double\* r = new double[n];

double\* Ar = new double[n];

double delta;

double t = 1;

do

{

for (int i = 0; i < n; i++)

{

\_r[i] = -A.b[i];

for (int j = 0; j < n; j++)

\_r[i] += A.value[i][j] \* X0[j];

}

for (int i = 0; i < n; i++)

{

Ar[i] = A.value[i][0] \* \_r[0];

for (int j = 1; j < n; j++)

{

Ar[i] += A.value[i][j] \* \_r[j];

}

}

t = scalar(\_r, \_r, n) / scalar(Ar, \_r, n);

for (int i = 0; i < n; i++)

{

X[i] = X0[i]- t \* \_r[i];

}

for (int i = 0; i < n; i++)

{

r[i] = fabs(X[i] - X0[i]);

E[i] = -A.b[i];

for (int j = 0; j < n; j++)

E[i] += A.value[i][j] \* X[j];

}

delta = Max(E, n);

if (OUT\_SDM)

{

if (k >= 2)

{

cout << "q = " << GetQ(X\_1, X0, X, n) << " ";

}

cout << k++ << " X = ";

for (int i = 0; i < n; i++)

{

if (X[i] > 0)

cout << " ";

cout << fixed << setprecision(5) << X[i] << " ";

}

cout << "tau = " << t;

cout << " ||r|| = " << Max(r, n);

cout << " ||A\*x-b|| = " << delta << endl;

}

copy(X\_1, X0, n);

copy(X0, X, n);

} while (delta > c\_delta);

delete[] E;

delete[] X0;

delete[] X\_1;

delete[] \_r;

delete[] Ar;

return X;

}

double\*\* init(int n, int m, bool fillByZero)

{

double\*\* A = new double\*[n];

for (int i = 0; i < n; i++)

A[i] = new double[m];

if (fillByZero)

for (int i = 0; i < n; i++)

for (int j = 0; j < n; j++)

A[i][j] = 0.0;

return A;

}

void copy(double\* A, double\* B, int n)

{

for (int i = 0; i < n; i++)

A[i] = B[i];

}

double Max(double\* E, int n)

{

double res = fabs(E[0]);

for (int i = 1; i < n; i++)

if (res < fabs(E[i]))

{

res = fabs(E[i]);

}

return res;

}

double Norm(double\*\* M, int n)

{

double sum = 0;

for (int j = 0; j < n; j++)

sum += fabs(M[0][j]);

double norm = sum;

for (int i = 1; i < n; i++)

{

sum = 0;

for (int j = 0; j < n; j++)

sum += fabs(M[i][j]);

if (sum > norm)

norm = sum;

}

return norm;

}

double scalar(double\* A, double\* B, int n)

{

double sum = A[0] \* B[0];

for (int i = 1; i < n; i++)

sum += A[i] \* B[i];

return sum;

}

double GetQ(double\* X\_1, double\* X0, double\* X, int n)

{

double\* a = new double[n];

double\* b = new double[n];

for (int i = 0; i < n; i++)

{

a[i] = X[i] - X0[0];

b[i] = X0[i] - X\_1[0];

}

double res = Max(a, n) / Max(b, n);

delete[] a;

delete[] b;

return res;

}