Московский авиационный институт
(национальный исследовательский университет)

Институт №8 "Информационные технологии и прикладная математика"
Кафедра 806 «Вычислительная математика и программирование»

Курсовая работа по курсу: “Численные методы”

Тема: “Решение систем линейных алгебраических уравнений с симметричными разреженными матрицами большой размерности. Метод сопряженных градиентов.”

Выполнил студент: М.Р. Чибисов

Группа: М8О-409Б-20

Преподаватель: Д.Е.Пивоваров

Дата:

Оценка:

Подпись:

Москва, 2023

Постановка задачи

Задача: Дана система линейных уравнений в матричном виде. Найти решение системы с помощью метода сопряженных градиентов.

Сравнить производительность метода сопряженных градиентов с методом Зейделя.

1 Описание

Метод сопряженных градиентов

Рассмотрим следующую задачу оптимизации: $F(x) = 1/2*(Ax, x) - (b, x) -> inf, x \in R^n$. Здесь A – положительно определенная симметричная разреженная матрица размера $n \cdot n$. Заметим, что $F'(x) = Ax - b$ и условие $F'(x) = 0$ эквивалентно $Ax - b = 0$. Функция F достигает своей нижней грани в единственной точке x^*, определяемой уравнением $Ax^* = b$.
Будем говорить, что ненулевые векторы $p^{(0)}, p^{(1)}, \ldots, p^{(m-1)}$ называются *взаимно сопряженными* относительно матрицы A, если $(Ap^{(n)}, p^{(l)}) = 0$ для всех $n \neq l$.
Под *методом сопряженных направлений* для минимизации квадратичной функции будем понимать метод

$$x^{(n+1)} = x^{(n)} + \alpha_n p^{(n)} (n = 0, 1, 2, \ldots, m-1),$$

в котором направления $p^{(0)}, p^{(1)}, \ldots, p^{(m-1)}$ взаимно сопряжены, а шаги

$$\alpha_n = \frac{(r^{(n)}, p^{(n)})}{(Ap^{(n)}, p^{(n)})},$$

где $r^{(n)} = r^{(n-1)} - \alpha_n Ap^{(n)}$ – антиградиент, получаются как решение задач одномерной минимизации:

$$\phi_n(\alpha_n) = min_{\alpha \geq 0} \phi_n(\alpha), \phi_n(\alpha) = F(x^{(n)}) + \alpha p^{(n)}$$

В методе сопряженных градиентов направления $p^{(n)}$ строят по правилу:

$$p^{(0)} = r^{(0)}, p^{(n)} = r^{(n)} + \beta_{n-1} p^{(n-1)}, n \geq 1,$$

где

$$\beta_{n-1} = \frac{(r^{(n)}, r^{(n)})}{(r^{(n-1)}, r^{(n-1)})},$$

$$r^{(n)} = r^{(n-1)} - \alpha_{n-1} Ap^{(n-1)}.$$

Анализ метода

Для метода сопряжённых градиентов справедлива следующая теорема: Теорема Пусть $F(x) = \frac{1}{2}(Ax, x) - (b, x)$, Aгде - симметричная положительно определённая матрица размера n. Тогда метод сопряжённых градиентов сходится не более чем за n шагов. В компьютерном представлении, однако, существуют проблемы с представлением вещественных чисел, в связи с чем, количество итераций может превышать n.

Сходимость

Более тонкий анализ показывает, что число итераций не превышает m, где m - число различных собственных значений матрицы A. Для оценки скорости сходимости верна следующая (довольно грубая) оценка:

$$||x_k - x_*||_A \leq (\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1})||x_0 - x_*||_A,$$

где $\kappa(A) = ||A||\,||A^{-1}|| = \lambda_1/\lambda_n$. Она позволяет оценить скорость сходимости, если известны оценки для максимального λ_1 и минимального λ_n собственных значений матрицы A. На практике чаще всего используют следующий критерий останова:

$$||r_k|| < \varepsilon.$$

Вычислительная сложность

На каждой итерации метода выполняется $O(n^2)$ операций. Такое количество операций требуется для вычисления произведения $Ap^{(n)}$ - это самая трудоёмкая процедура на каждой итерации. Отальные вычисления требуют $O(n)$ операций. Суммарная вычислительная сложность метода не превышает $O(n^3)$ - так как число итераций не больше n.

2 Исходный код

Код метода сопряженных градиентов:

```
class Solver:
    def __init__(self, matrix, b, x0=None, eps=1e-5):
        self.matrix = matrix
        self.b = b
        self.eps = eps
        self.shape = matrix.shape[0]
        self.x0 = np.array([0] * self.shape) if x0 is None else x0
        self.k = 0

    def solve(self, max_iter=100000):
        x0 = self.x0
        r0 = self.b - self.matrix.dot(x0)
        p0 = np.copy(r0)
        for _ in range(max_iter):
            temp = self.matrix @ p0
            norm_0 = np.dot(r0, r0)
            alpha_i = norm_0 / (temp @ p0)
            x_new = x0 + p0 * alpha_i
            r_new = r0 - temp * alpha_i
            norm_new = r_new @ r_new
            beta_i = norm_new/norm_0
            p_new = r_new + p0 * beta_i
            r0 = r_new
            p0 = p_new
            x0 = x_new
            self.k += 1
            if norm(r_new) < self.eps:
                break

        return x0

    def solve_and_print(self):
        start = time()
        x = self.solve()
        end = time()
        start2 = time()
        x2 = np.linalg.solve(self.matrix.toarray(), self.b)
        end2 = time()
        print('Метод сопряженных градиентов:\n')
        print(f'Ответ:\nX = {x.round(6)}\n')
        print(f'eps = {self.eps}; Размерность задачи = {self.shape}\
n'
```

```
              f'Количество итераций = {self.k}; Время решения = 
{round(end - start, 8)} сек.\n')
        print('Встроенный метод NumPy:\n')
        print(f'Ответ: X = {x2.round(6)}\n')
        print(f'Время решения = {round(end2 - start2, 8)} сек.\n')
```

3 Входные данные

1) Размерность 10x10:

Файл: “matrix10.txt”

Матрица:

0.681 0.0 0.0 0.511 0.0 0.0 0.055 0.09 0.0 0.594
0.0 0.0 0.0 0.801 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.175 0.384 0.405 0.179 0.0 0.0
0.511 0.801 0.0 0.438 0.0 0.188 0.0 0.0 0.645 0.167
0.0 0.0 0.175 0.0 0.688 0.0 0.0 0.0 0.0 0.541
0.0 0.0 0.384 0.188 0.0 0.0 0.349 0.221 0.0 0.0
0.055 0.0 0.405 0.0 0.0 0.349 0.0 0.714 0.813 0.916
0.09 0.0 0.179 0.0 0.0 0.221 0.714 0.136 0.0 0.0
0.0 0.0 0.0 0.645 0.0 0.0 0.813 0.0 0.0 0.944
0.594 0.0 0.0 0.167 0.541 0.0 0.916 0.0 0.944 0.0

Вектор правой части:

11 48 6 39 16 42 28 41 40 24

2) Матрица размерности 50x50

Файл: “matrix50.txt”

3) Матрица размерности 100x100

Файл: “matrix100.txt”

4 Результат программы

1) Размерность 10x10:

```
Метод сопряженных градиентов:

Ответ:
X = [   0.665834  -92.060344  282.159872   59.925094  -61.723385  213.14983
  -17.846231 -323.01798    67.093753   16.797988]

eps = 1e-05; Размерность задачи = 10
Количество итераций = 10; Время решения = 0.0 сек.

Встроенный метод NumPy:

Ответ:
X = [   0.665834  -92.060344  282.159872   59.925094  -61.723385  213.14983
  -17.846231 -323.01798    67.093753   16.797988]

Время решения = 0.00752091 сек.
```

2) Размерность 50x50:

```
Метод сопряженных градиентов:

Ответ:
X = [ -6.763391 -44.594026  23.229361 -15.250167  45.298561 -49.541085
 -69.913598  85.16005    9.096029   4.378954  20.283822  -4.458019
 -32.415857  20.579247 -78.780728 -12.514714  27.274292  42.80142
 -46.09007   58.679656  46.142755 -99.820086   8.17838   25.230273
  -7.681621 -13.054742 -18.024317   5.707361  95.62445   27.517424
  31.522898 -16.568125  17.684954   6.730134   5.935751  21.344254
 -42.942265   7.746614  12.849293 -17.852811  60.627065 -75.402058
  37.689047  -0.740842  24.533548  10.481654 -30.573812  68.471723
 -12.541523  75.670899]

eps = 1e-05; Размерность задачи = 50
Количество итераций = 65; Время решения = 0.00200105 сек.

Встроенный метод NumPy:

Ответ:
X = [ -6.763391 -44.594026  23.229361 -15.250168  45.298561 -49.541085
 -69.913598  85.16005    9.096029   4.378954  20.283822  -4.458019
 -32.415857  20.579246 -78.780728 -12.514714  27.274292  42.80142
 -46.09007   58.679657  46.142755 -99.820086   8.17838   25.230273
  -7.681622 -13.054742 -18.024317   5.707361  95.62445   27.517424
  31.522898 -16.568125  17.684954   6.730134   5.935751  21.344254
 -42.942265   7.746614  12.849293 -17.852811  60.627066 -75.402057
  37.689048  -0.740842  24.533547  10.481654 -30.573812  68.471723
 -12.541523  75.670899]

Время решения = 0.01356912 сек.
```

3) Размерность 100x100:

```
Метод сопряженных градиентов:

Ответ:
X = [  0.274577 -11.739124  18.164421 -19.363398 -43.613846 -20.934536
   6.99926   -4.040206  37.831838 -25.539045  19.871305 -48.619783
 -26.61444   60.577003 -14.17344   14.40377  -14.649436 -22.052706
  -5.293717  23.808237  -4.528203 -16.643478  12.901231  24.865092
 -10.697894 -13.379116 -23.948959  19.094514 -13.28568   -8.370087
 -55.664525  57.665223   3.601418  -5.731843 -33.077901 -11.110348
  45.586919 -29.959717  48.138249 -17.029823  17.566529 -47.597814
  -8.680617  31.294793  45.392373  32.811753  11.253839 -47.175023
  -8.42033    0.464242  25.704512  19.225267 -35.487869  -0.326243
 -16.648957   7.504197  -6.152218 -19.475281  29.815533  -6.074269
  -5.371827  57.166822 -29.52573   28.275852 -21.932059  22.17499
  52.306801  43.651264   1.637413 -40.545267  53.360007  -3.665139
  10.932137 -18.109924  -6.213701  55.900645 -12.727063 -11.816374
  -9.751129 -22.190445   3.867457 -24.871975  -7.635048 -14.7743
  47.211649  23.940891  10.339103   7.355336  31.260534  -8.099518
  -2.268315 -27.703597   6.432415 -11.139453  50.397292 -21.7007
  27.80699   10.940165  -3.340575   4.867272]

eps = 1e-05; Размерность задачи = 100
Количество итераций = 155; Время решения = 0.00751328 сек.

Встроенный метод NumPy:

Ответ:
X = [  0.274577 -11.739125  18.16442  -19.363398 -43.613847 -20.934536
   6.99926   -4.040206  37.831838 -25.539045  19.871305 -48.619783
 -26.614439  60.577003 -14.173439  14.40377  -14.649436 -22.052706
  -5.293717  23.808237  -4.528203 -16.643477  12.90123   24.865092
 -10.697894 -13.379115 -23.948959  19.094514 -13.28568   -8.370087
 -55.664525  57.665223   3.601418  -5.731843 -33.077902 -11.110348
  45.586919 -29.959717  48.138249 -17.029823  17.566529 -47.597814
  -8.680617  31.294793  45.392373  32.811753  11.253839 -47.175023
  -8.42033    0.464243  25.704512  19.225268 -35.487869  -0.326243
 -16.648957   7.504196  -6.152218 -19.475281  29.815533  -6.074269
  -5.371827  57.166822 -29.52573   28.275852 -21.93206   22.17499
  52.306801  43.651264   1.637413 -40.545267  53.360006  -3.665139
  10.932137 -18.109924  -6.213701  55.900645 -12.727062 -11.816374
  -9.751129 -22.190445   3.867457 -24.871975  -7.635048 -14.7743
  47.211649  23.940891  10.339103   7.355336  31.260534  -8.099518
  -2.268315 -27.703597   6.432415 -11.139454  50.397292 -21.7007
  27.80699   10.940165  -3.340575   4.867272]

Время решения = 0.00500441 сек.
```

5 Вывод

В курсовой работе был рассмотрен метод сопряженных градиентов для решения систем линейных алгебраических уравнений с симметричными разреженными матрицами большой размерности. Этот метод позволяет эффективно находить корни таких систем, в том числе и в случае, когда встроенные методы решения не применимы.

При сравнении метода сопряженных градиентов с методом NumPy, следует отметить, что последний не предназначен для работы с разреженными матрицами большой размерности. В то же время метод сопряженных градиентов демонстрирует высокую эффективность при работе с такими матрицами.

Таким образом, метод сопряженных градиентов является эффективным инструментом для решения систем линейных алгебраических уравнений с симметричными разреженными матрицами большой размерности, и может быть использован в тех случаях, когда встроенные методы решения не применимы.