Санкт-Петербургский политехнический университет Петра Великого
Институт прикладной математики и механики
Высшая школа прикладной математики и вычислительной физики

Вычислительные комплексы

Отчёт по заданию №4

Работу выполнил:
Д. А. Козлов
Группа:
3630102/70201
Преподаватель:
А. Н. Баженов

Санкт-Петербург
2020

Содержание

1. Постановка задачи **3**

2. Конкретизация задачи и метод решения **3**

3. Решение **4**

4. Приложение **6**

1. Постановка задачи

Решить ИСЛАУ с применением линейного программирования для проведения регуляризации системы. Рассмотреть несовместную ИСЛАУ с точечной матрицей A и интервальным вектором b. Провести l_1-регуляризацию так, чтобы сумма масштабирующих множителей была наименьшей.

2. Конкретизация задачи и метод решения

При решении данной задачи будем рассматривать ИСЛАУ $Ax = b$ с точечной матрицей A и интервальной правой частью b, при которых система не имеет решений до проведения регуляризации. В данной работе выбрана несовместная ИСЛАУ

$$A = \begin{pmatrix} 1.2 & -2 & 1 \\ 1 & -2 & 1 \\ 7 & 0 & 0 \end{pmatrix}$$

$$mid\_b = \begin{pmatrix} 1.5 & 3.5 & 2.5 \end{pmatrix}$$

Положим все радиусы вектора b равными 0.5. Тогда вектор b:

$$b = \begin{pmatrix} [1;2] \\ [3;4] \\ [2;3] \end{pmatrix}$$

Проверим отсутствие решений у данной системы. С помощью программы tolsolvty были найдем максимум функционала распознающего функционала max Tol и значение аргумента, в которой он достигался arg max Tol .

$$maxTol = -0.52$$

$$argmaxTol = \begin{pmatrix} 0.21 & -0.91 & 0.45 \end{pmatrix}$$

Поскольку $maxTol < 0$, то допусковое множество ИСЛАУ пусто.

Для получения решения проводится l_1 - регуляризация, заключающуюся в изменении радиусов компонент вектора b их поэлементным умножением на вектор масштабирующих множителей w

$$b = \begin{pmatrix} [midb_1 - radb_1; midb_1 + radb_1] \\ [midb_2 - radb_2; midb_2 + radb_2] \\ [midb_3 - radb_3; midb_3 + radb_3] \end{pmatrix} \longrightarrow \tilde{b} = \begin{pmatrix} [midb_1 - \omega_1 radb_1; midb_1 + \omega_1 radb_1] \\ [midb_2 - \omega_2 radb_2; midb_2 + \omega_2 radb_2] \\ [midb_3 - \omega_3 radb_3; midb_3 + \omega_3 radb_3] \end{pmatrix}$$

При этом масштабирующие множители подбираются так, чтобы регуляризованная ИСЛАУ $Ax = \tilde{b}$ стала разрешима, но сумма этих множителей $\sum \omega_i$ была минимально возможной. Так же понятно, что масштабирующие множители не могут быть отрицательными.

Введем вектор $u = \begin{pmatrix} x \\ \omega \end{pmatrix}$, тогда задана следующая задача линейного программирования:

$$\begin{cases} u_{4,5,6} \geq 0 \\ cu = (0,0,0,1,1,1) \cdot (x_1, x_2, x_3, \omega_1, \omega_2, \omega_3)^T = \sum \omega_i \to \min_u \\ Cu \leq r \end{cases}$$

где

$$C = \begin{pmatrix} -A - diag(radb)) \\ A - diag(radb)) \end{pmatrix}$$

$$r = \begin{pmatrix} -midb \\ midb \end{pmatrix}$$

Полученную задачу будем решать с применением функции linprog пакета scipy.optimize. В результате решения определяются одновременно и необходимые масштабирующие множители, и соответствующее им появившееся в результате регуляризации решения ИСЛАУ.

Для реализации решения была написана программа на языке Python. Ссылка на репозиторий приведена в приложении. Так же используется функция **tolsolpy**.

3. Решение

Найдем решение двумя разными методами: симплекс методом (simplex) и методом внутренней точки (interior-point):

- Simplex:

$$x \approx \begin{pmatrix} 0.36 \\ -1.57 \\ 0. \end{pmatrix}, \omega \begin{pmatrix} 4.14 \\ 0 \\ 0 \end{pmatrix}, \sum \omega_i \approx 4.14$$

- Interior-point:

$$x \approx \begin{pmatrix} 0.36 \\ -0.83 \\ 0.41 \end{pmatrix}, \omega \approx \begin{pmatrix} 2.00 \\ 2.14 \\ 0.00 \end{pmatrix}, \sum \omega_i \approx 4.14$$

Оба метода пришли к одинаковому минимуму функции $\sum \omega_i \approx 4.14$, что не удивительно. При этом в зависимости от метода решения задачи линейного программирования, оптимизация сходится к разным решениям. Это может означать, что задача имеет множество одинаково хороших решений. Для исследования этого вопроса рассмотрим задачу условной оптимизации, в рамках которой на вектор решения x накладываются дополнительные ограничения:

$$x_{imin} \leq x_i \leq x_{imax}$$

В результате было обнаружено, что при нижней границе для первой координаты большей 0.36, условная оптимизация сходится к x_{1min}, а $\sum \omega_i$ при этом растет линейно. Это означает, что решение $x_1 = 0.36$ действительно является оптимальным и устойчивым.

Если не накладывать ограничений на первую координату, но при этом накладывать решения на одну из других координат, то получаем одинаковое значение целевой функции (суммы масштабирующих коэффициентов).

Покажем это. Ограничим одну из координат некоторой полосой. Например, x_3 полосой шириной 0.5: $x_3 \in [x_{3min}; x_{3min} + 0.5]$. Посмотрим на зависимость получающегося решения от x_3. Как видим, в обоих случаях $x_1 = 0.36$, а так же наблюдается линейная зависимость между x_3 и x_2.

Рисунок 3.1. Графики зависимости x_1 и x_2 при накладывании ограничений на x_3. Симплекс метод

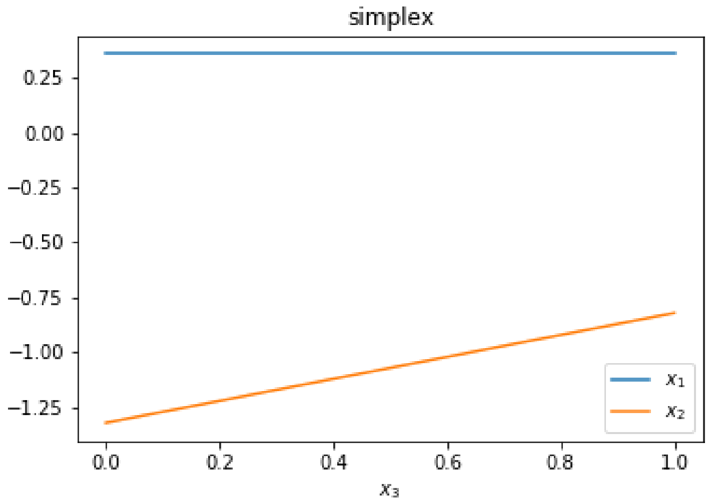
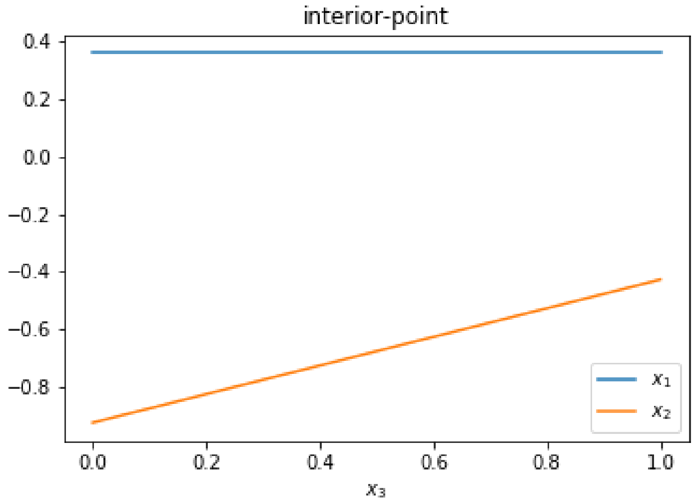
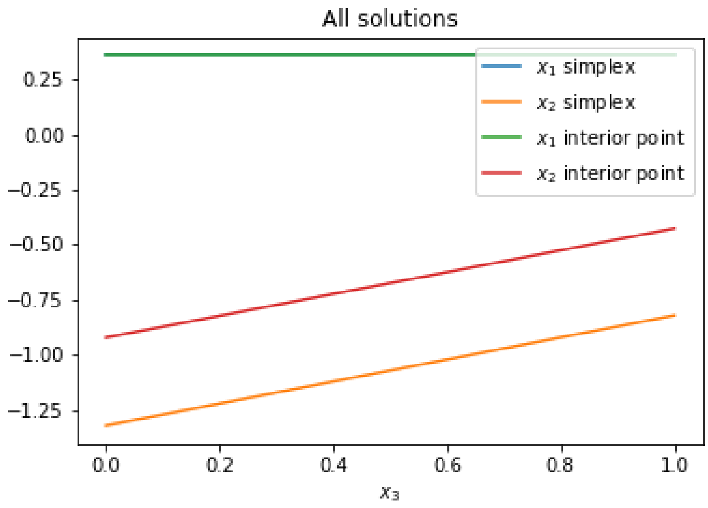


Рисунок 3.2. Графики зависимости x_1 и x_2 при накладывании ограничений на x_3. Метод внутренней точки



Для сравнения решений, полученных разными методами, построим общий график.

Рисунок 3.3. Объединение зависимостей $x_3(x_1)$ с обоих методов



Таким образом, можно заключить, что множество одинаково хороших решений поставленной задачи линейного программирования соответствует фиксированному значению $x_1 = 0.36$, а координаты x_3 и x_1 имеют при этом линейную связь.

4. Приложение

Репозиторий с кодом: https://github.com/KoDim97/Computer-Systems/tree/lab4