Санкт-Петербургский политехнический университет Петра Великого
Институт прикладной математики и механики
Кафедра «Прикладная математика»

ОТЧЁТ ПО ЛАБОРАТОРНЫМ РАБОТАМ ПО ДИСЦИПЛИНЕ «ВЫЧИСЛИТЕЛЬНЫЕ КОМПЛЕКСЫ»

Выполнил
студент группы 3630102/70201

Крупкина Дарья

Проверил
к. ф.-м. н., доцент

Баженов Александр Николаевич

Санкт-Петербург
2020

Содержание

Список иллюстраций

1 Постановка задачи

Требуется решить ИСЛАУ с применением аппарата линейного программирования для проведения регуляризации рассматриваемой системы.

2 Конкретизация задачи и теория

При решении данной задачи имеет рассмотреть ИСЛАУ $Ax = b$ точечной марицей Aи интервальной правой частью $\mathbf{b}$ при которых система не имеет решений до проведения регуляризации. В данной работе выбрана несовместная ИСЛАУ:

$$\begin{pmatrix} 10 & -11 & -12 \\ 1.1 & 0 & 0 \\ -5 & 11 & 12 \end{pmatrix} \cdot x = \begin{pmatrix} [-1;1] \\ [-5;-3] \\ [6;8] \end{pmatrix} \tag{1}$$

В первую очередь с помощью распознающего функционала $Tol(x)$ проверяется отсутствие решений у данной системы. С помощью программы tolsolvty были найдены максимум функционала распознающего функционала $maxTol$ и значение аргумента, в которой он достигался $argmaxTol$:

$$maxTol = -2.8472; argmaxTol = \begin{pmatrix} -0.1389 \\ 0.102 \\ 0.1112 \end{pmatrix} \tag{2}$$

Поскольку $maxTol < 0$, допусковое множество ИСЛАУ пусто и система несовместна. Далее для получения решения проводится l_1-регуляризация, заключающуюся в изменении радиусов компонент вектора $\mathbf{b}$ их поэлементным домножением на вектор масштабирующих множителей ω:

$$\mathbf{b} = \begin{pmatrix} [midb_1 - radb_1; midb_1 + radb_1] \\ [midb_2 - radb_2; midb_2 + radb_2] \\ [midb_3 - radb_3; midb_3 + radb_3] \end{pmatrix} \to \bar{\mathbf{b}} = \begin{pmatrix} [midb_1 - \omega_1 radb_1; midb_1 + \omega_1 radb_1] \\ [midb_2 - \omega_2 radb_2; midb_2 + \omega_2 radb_2] \\ [midb_3 - \omega_3 radb_3; midb_3 + \omega_3 radb_3] \end{pmatrix} \tag{3}$$

При этом масштабирующие множители подбираются так, чтобы регуляризованная ИСЛАУ $A \cdot x = \bar{\mathbf{b}}$ стала разрешима, но сумма этих множителей $\sum_i \omega_i$ была минимально возможной.
Накладывая на масштабирующие множители естественное требование их неотрицательности, и введя вектор $u = \begin{pmatrix} x \\ \omega \end{pmatrix}$, можно записать полученную задачу в виде:

$$
\begin{cases}
u_{4,5,6} \geq 0 \\
c \cdot u = (0,0,0,1,1,1) \cdot u = (0,0,0,1,1,1) \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} = \sum_i \omega_i = \min\limits_u \\
C \cdot u \leq r, \text{где } C = \begin{pmatrix} -A & -diag(rad(\mathbf{b})) \\ A & -diag(rad(\mathbf{b})) \end{pmatrix}, r = \begin{pmatrix} -mid(\mathbf{b}) \\ mid(\mathbf{b}) \end{pmatrix}
\end{cases}
\tag{4}
$$

Полученная задача и решается линейным программированием с применением стандартной функции linprog пакета scipy.optimize. В результате решения определяются одновременно и необходимые масштабирующие множители, и соответствующее им появившееся в результате регуляризации решения ИСЛАУ.

3 Реализация

Лабораторная работа выполнена с помощью встроенных средств языка программирования Matlab и Python. Использованы библиотеки IntLab для интервальной арифметики, tolsolovty для нахождения решения ИСЛАУ. Также используется оптимизатор scipy.optimize на Python с различными методами решения задачи линейного программирования. Исходный код лабораторной работы приведён в приложении в виде ссылки на репозиторий GitHub.

4 Результаты

Имеем $rad(\mathbf{b}) = 1, mid(\mathbf{b}) = \begin{pmatrix} 0 & -4 & 7 \end{pmatrix}$.
После регуляризации получено:

$$
c = (0,0,0,1,1,1); C = \begin{pmatrix} 10 & -11 & -12 & -1 & 0 & 0 \\ 1.1 & 0 & 0 & 0 & -1 & 0 \\ -5 & 11 & 12 & 0 & 0 & -1 \\ -10 & 11 & 12 & -1 & 0 & 0 \\ -1.1 & 0 & 0 & 0 & -1 & 0 \\ 5 & -11 & -12 & 0 & 0 & -1 \end{pmatrix}; r = \begin{pmatrix} 0 & -4 & 7 & 0 & 4 & -7 \end{pmatrix}
\tag{5}
$$

В результате применения стандартного linprog для решения задачи линейного программирования с использованием значений из (5) без дополнительных ограничений получны следующие результаты:

- Решение регуляризованной ИСЛАУ методом method = 'interior-point':

$$x \approx \begin{pmatrix} 1.4 \\ 0.5807 \\ 0.6344 \end{pmatrix}, \omega \approx \begin{pmatrix} 0 \\ 5.54 \\ 0 \end{pmatrix} \tag{6}$$

- Решение регуляризованной ИСЛАУ методом method = 'simplex':

$$x \approx \begin{pmatrix} 1.4 \\ 0.0 \\ 1.667 \end{pmatrix}, \omega \approx \begin{pmatrix} 0 \\ 5.54 \\ 0 \end{pmatrix} \tag{7}$$

Результаты из программы:

```
Interior-method results:
x: (1.4,0.5807,0.6344)
w: (0.0,5.54,0.0)
Simplex results:
x: (1.4,0.0,1.1667)
w: (0.0,5.54,0.0)
```

Рис. 1: Решения регуляризованной ИСЛАУ методами linprog без дополнительных ограничений

Заметно, что масштабирующие коэффициенты в обеих задачах совпали и их сумма равна 5.54. Кроме того, первые компоненты вектора решений оказались равны(первая - с точностью до 10^{-10}).
Это может означать, что для третьей компоненты решения может существовать множество решений, удовлетворяющих задаче минимизации.
Рассмотрим изменения нижних границ для 2-й и 3-й границ соответственно, чтобы убедиться в расширении интервала для достоверных решений.
Во всех графиках ниже по оси абцисс - значение нижней границы компоненты, по оси ординат - значение самой компоненты.
При изменении границ для одной из координат в обоих методах компоненты вектора решения симметрично меняются:

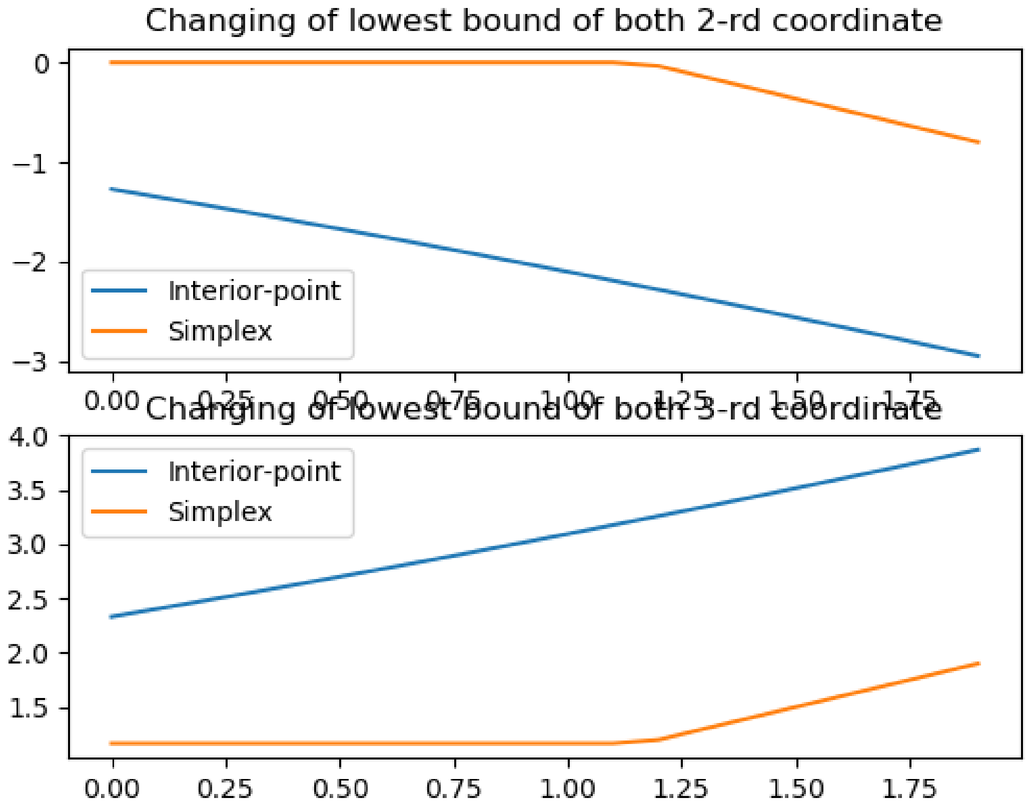

Рис. 2: Изменения нижней границы одной координат решения в обоих методах

Стоит также отметить, что в случае изменения границ только в симплекс-методе, в определенный момент решения совпадают, причем по всем компонентам(в точке $(1.40.58070.6344)^T$).

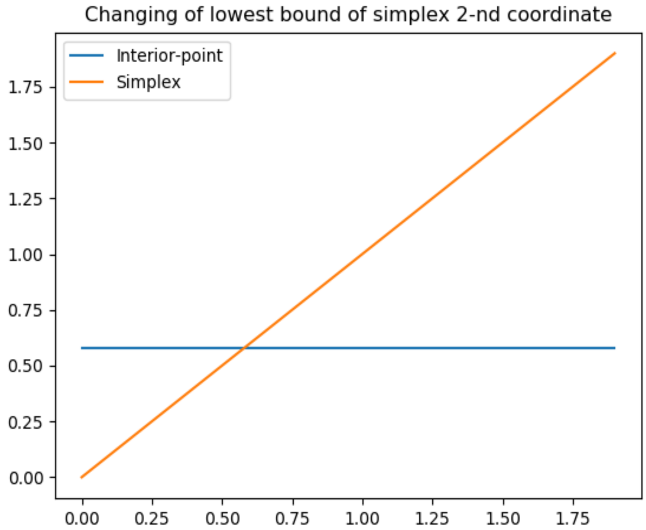

Рис. 3: Изменение нижней границы второй координаты вектора решений для симплекс-метода

В случае, если вносить изменения в границы 2-х компонент решения, будет заметно, что с той же точки решения будут совпадать:

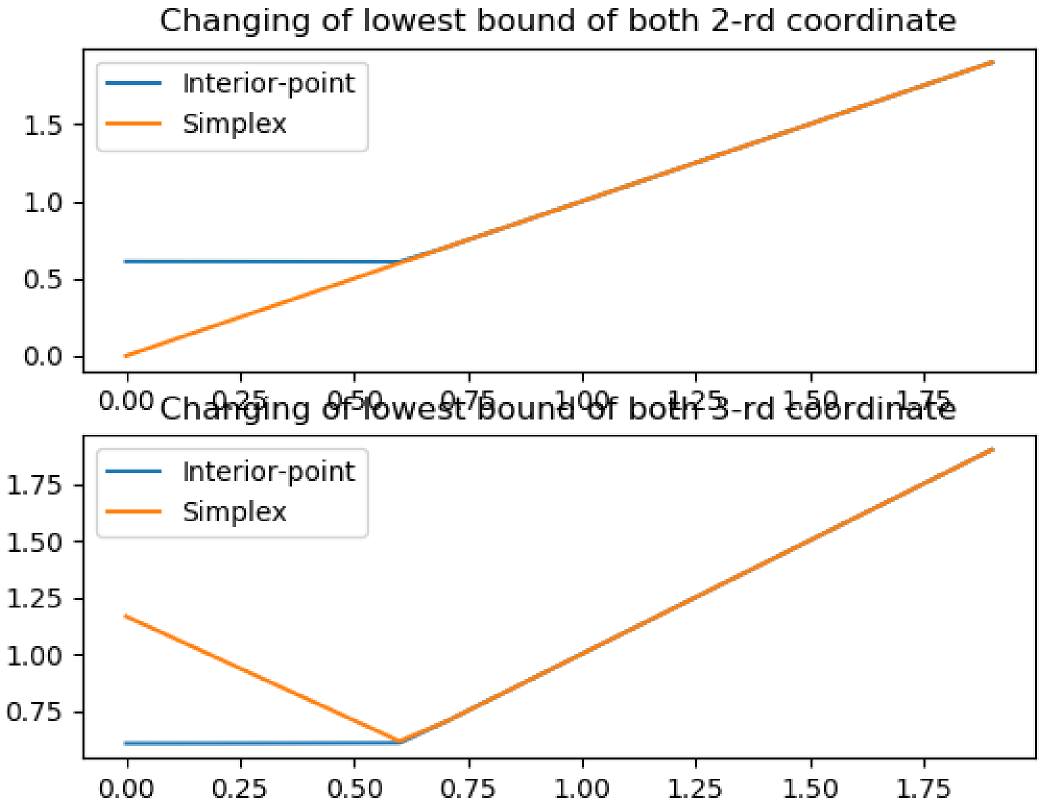

Рис. 4: Изменение нижней границы обеих компонент вектора решений в обоих методах

Таким образом, можно заключить, что существует множество решений поставленной задачи линейного программирования, соответствующее фиксированному значению $x_1 = 1.4$ и целой полосе возможных значений по остальным компонентам.

5 Приложения

Код программы на GitHub, URL: `https://github.com/DariaKrup/Computational_complexes`