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

Отчёт по научно-исследовательской работе на тему: *Портирование алгоритма субдифференциального метода на язык программирования Python*

Студент группы 3640102/00201 ____________________ Курносов Д.А.

Оценка руководителя: ____________ ____________ Баженов А.Н.

____ ____________ 2021 г.

Санкт-Петербург
2021 г.

Содержание

1 Постановка задачи

Основной идеей данной работы было изучение субдифференциального метода Ньютона с целью реализации его на языке программирования Python. Была изучена соответствующая литература, а так же алгоритмы, представленные на других языках программирования.

2 Теория

2.1 Субдифференциальный метод Ньютона

Решение интервальных уравнений может быть сведено к решению индуцированных уравнений вида:

$$\begin{cases} \mathcal{F}(y) = 0, \\ \mathcal{F}(y) = sti(Csti^{-1}(y) \ominus sti^{-1}(y) + d) = sti(Csti^{-1}(y)) - y + sti(d) \end{cases} \tag{1}$$

$$\begin{cases} \mathcal{G}(y) = 0, \\ \mathcal{G}(y) = sti(Csti^{-1}(y) \ominus d) = sti(Csti^{-1}(y)) - sti(d) \end{cases} \tag{2}$$

при помощи погружения $sti : \mathbb{KR}^n \to \mathbb{R}^{2n}$, действующего по правилу:

$$(x_1, x_2, ..., x_n) \mapsto (-\underline{x_1}, -\underline{x_2}, ..., -\underline{x_n},\ \bar{x_1}, \bar{x_2}, ..., \bar{x_n}) \tag{3}$$

Иначе говоря, необходимо составить вектор, первые n элементов которого представляют левые концы интервалов, взятые с противоположным знаком, а последующие n компонент - правые концы интервалов.
Одним из инструментов решения уравнения такого вида является субдифференциальный метод Ньютона. Идея данного алгоритма следующая:

- Выбирается некоторое приближение $x^{(0)} \in \mathbb{R}^{2n}$
- Вычисляется субградиент $D^{(0)}$ отображения $\mathcal{F}$ в данной точке
- Полагаем $x^1 \leftarrow x^0 - \tau(D^{(k-1)})^{-1}\mathcal{F}(x^{(k-1)})$,
  где $\tau \in [0, 1]$

Условием остановки алгоритма является условие:

$$\|x_k - x_{k-1}\| \leq \varepsilon \tag{4}$$

Дополнительным условием остановки является заданное заранее допустимое количество итераций алгоритма. Важно заметить, что напрямую данный метод может использоваться для решения СЛАУ только с квадратными матрицами. Рассмотрим подход, позволяющий использовать его для решения задач иных размерностей.

2.2 Расширение алгоритма

Для решения задач, представляющих СЛАУ прямоугольного вида используется следующий алгоритм:

1. Матрица A исходной системы изменяется таким образом, чтобы оставшиеся в ней переменные имели достаточное значение в уравнениях, т.е. из исходной матрицы удаляются столбцы, сумма значений в которых равняется нулю.
2. Далее по такой матрице производится проход по окнам - квадратным матрицам, размерность которых совпадает с количеством столбцов этой матрицы. Для каждого окна выполяются следующие операции:
   - Если матрица, являющаяся окном имеет определитель, удовлетворяющий заранее заданному значению - находим для неё решение с помощью субдифференциального метода Ньютона.
   - Если определитель оконной матрицы равен нулю - для неё выполняется процедура п.1 до тех пор, пока определитель не будет удовлетворять заданному значению. Если такая матрица найдена в рамках этого окна - находим для неё решение. Иначе - переходим к следующему окну.
3. Полученные решения пересекаются в одно, являющееся окончательным ответом.

Перед выполнением пересечения каждое полечученное решение проверяется подстановкой в равеноство $Ax = b$. Умножение выполняется в соответствии с формулами Лакеева для полной интервальной арифметики. Равенство же проверяется как сумма норм разностей для нижних и верхних границ векторов.
Пересечение находится как минимум по включению для правильных проекций интервалов.

3 Результаты

Решим систему, представленную интервальной матрицей размерности 256x16. Задача была сгенерированна с помощью кода Никиты Суханова и интервализирована. Данная система описывает два геометрических объекта, первый - цилиндр(плазма), который разбивается на множество сегментов по трём полярным координатам (r, phi, z), второй - прямоугольник, на котором распологается множество точечек (детектор), распределенных в узлах прямоугольной сетки. Всего их $cols$ столбцов и в каждом столбце $rows$ точек. Соответственно размерность 256x16 в нашем случае получается из параметров: $n = n_r * n_{phi} * n_z, n_r = 2, n_{phi} = 4, n_z = 2$ сегментов цилиндра и $m = rows * cols, rows = 16, cols = 16$ точек (пикселей) на детекторе.
Элементы матрицы отображают длину пересечения некоторого луча, испускаемого каждым пикселем, с каждым из сегментов цилиндра. Так как пересечения не повсеместны, матрицы содержит большое количество нулевых элементов.
Матрица данной системы имеет следующий вид:

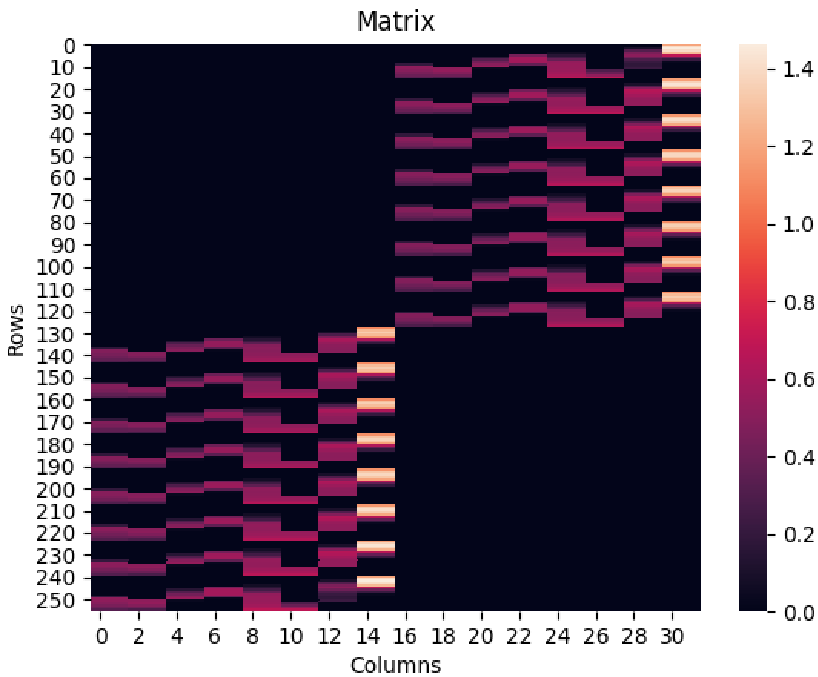

Рис. 1: Исходная матрица системы

Видно, что матрица имеет блочную структуру. Для нахождения решения разобьём её на четверти и получим решения для каждой из них. Значения $None$ означают, что решение не удалось получить.

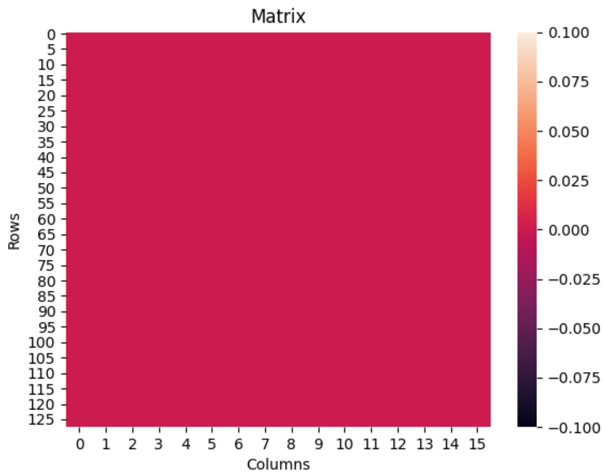

Рис. 2: Матрица первой четверти

Решение для первой четверти:

$$\begin{pmatrix} 1 : [None, None] \\ 2 : [None, None] \\ 3 : [None, None] \\ 4 : [None, None] \\ 5 : [None, None] \\ 6 : [None, None] \\ 7 : [None, None] \\ 8 : [None, None] \end{pmatrix}$$

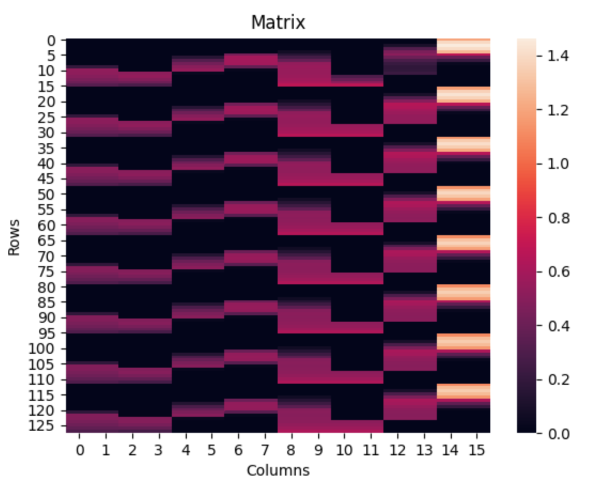

Рис. 3: Матрица второй четверти

Решение для второй четверти:

$$\begin{pmatrix} 1 : [1.000000000000001, 1.0000000000000004] <- \\ 2 : [1.0000000000000013, 1.0000000000000027] -> \\ 3 : [1.0000000000000548, 0.9999999999999444] <- \\ 4 : [1.000000000000018, 0.9999999999999918] <- \\ 5 : [1.0000000000000073, 0.9999999999999802] <- \\ 6 : [1.0000000000000038, 1.0000000000000044] -> \\ 7 : [1.0000000000000548, 0.9999999999999444] <- \\ 8 : [1.0, 1.0] -> \end{pmatrix}$$

Количество слияний ответов для матрицы второй четверти $= 251$

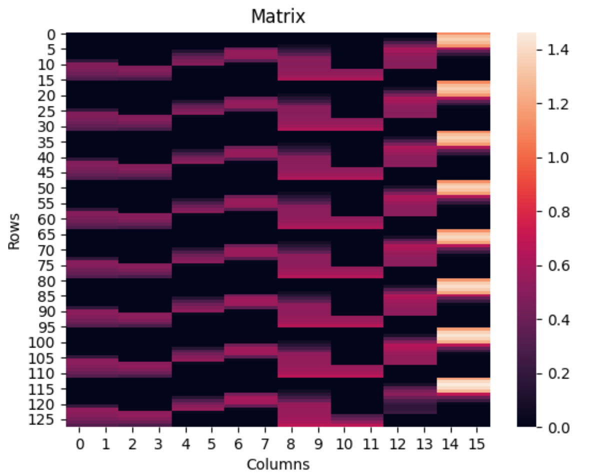

Рис. 4: Матрица третьей четверти

Решение для третьей четверти:

$$\begin{pmatrix} 1 : [1.000000000000001, 1.0000000000000004] < - \\ 2 : [1.0000000000000013, 1.0000000000000027] - > \\ 3 : [1.0000000000000548, 0.9999999999999914] < - \\ 4 : [1.000000000000018, 0.9999999999999918] < - \\ 5 : [1.0000000000000073, 0.9999999999999802] < - \\ 6 : [1.0000000000000013, 1.0000000000000044] - > \\ 7 : [1.0000000000000548, 0.9999999999999914] < - \\ 8 : [1.0, 1.0] - > \end{pmatrix}$$

Количество слияний ответов для матрицы третьей четверти $= 248$

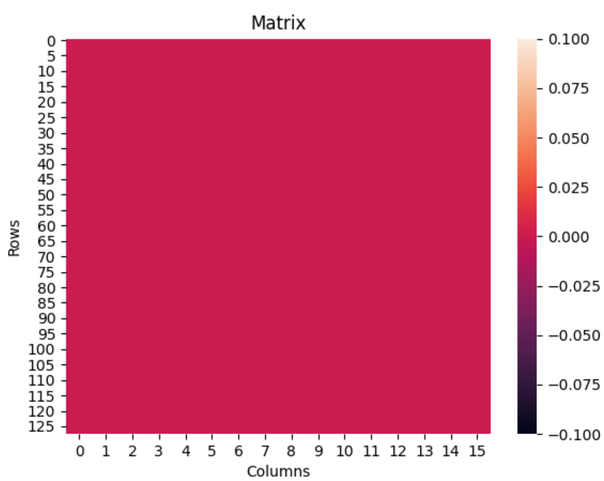

Рис. 5: Матрица четвертой четверти

Решение для четвертой четверти:

$$\begin{pmatrix} 1 : [None, None] \\ 2 : [None, None] \\ 3 : [None, None] \\ 4 : [None, None] \\ 5 : [None, None] \\ 6 : [None, None] \\ 7 : [None, None] \\ 8 : [None, None] \end{pmatrix}$$

4 Литература

- С. П. Шарый, Алгебраический подход во “внешней задаче” для интервальных линейных систем, Фундамент. и прикл. матем., 2002, том 8, выпуск 2, 567–610
- А.Н.Баженов, Интервальный анализ. Основы теории, практические применения и учебные примеры., 2020, 150-158
- Исходный код: https://github.com/ExpressFromSiberia/Interval_Analysis
- Код Никиты Суханова: https://github.com/NikitaSukhanov/Plasma3D