**Численное решение антагонистической матричной игры. Визуализация спектров оптимальных стратегий игроков.**

**Автор:** Ефремова Ольга Игоревна, 311 группа

**Необходимое дополнительное ПО:** библиотеки SciPy, Numpy, Plotly

**Для запуска:** необходимо разместить файл fun.py и notebook.ipynb в одной директории, открыть файл notebook.ipynb в Jupyter, запустить выполнение программы с первой строчки.

**Описание подхода к решению:**

1. fun.py: содержит необходимую для численного решения задачи функцию *“Nash\_Equilibrium(a)”*, которая принимает матрицу выигрыша и возвращает значение игры и оптимальные стратегии первого и второго игроков. Функция работает по принципу сведения матричной игры к задачи линейного программирования. Для использования функции *“lingprog”*, решающей задачу симплекс-методом, необходимо построить верное уравнение. Т.к. *“lingprog”* решает только задачу минимизации, то для второго игрока то целевую задачу мы умножаем на -1, приводя ее к задачи минимизации. Получая решения прямой задачи, находим значение игры и оптимальную стратегию второго игрока. Для первого игрока необходимо транспонировать матрицу выигрышей (чтобы уравнение приняло правильный вид), и умножить правые и левые части уравнения на -1, чтобы задача для симплекс-метода приняла требуемый вид. Далее по аналогии со вторым игроком находим оптимальную стратегию для первого игрока. Функция завершает свою работу, возвращая ответ
2. Jupyter: 1-2: Импортируется необходимое ПО и функция из предыдущего пункта. 3: Задаем матрицу и находим решение матричной игры, после чего выводим полученные значения, предварительно округлив их до 3х знаков после запятой для удобства. 4: Строим графики секторов оптимальных значений для первого и второго игроков с помощью функции *“scatter”*. С помощью функций *“update\_xaxes”* и *“update\_yaxes”* график приводится к опрятному виду: убираются заголовки, добавляются оси и отметки. Аналогично делается для остальных матриц.