В этой лабораторной вы будете анализировать сети (графы) с помощью матриц.

Задание 1. Кластеризация социальной сети. Представьте, что вам нужно проанализировать социальную сеть – множество людей, часть из которых являются "друзьями" друг друга. Для простоты будем считать, что отношения между людьми всегда взаимные (иными словами, мы не будем рассматривать односторонних "подписчиков"). Наша задача заключается в том, чтобы выделить в этой социальной сети несколько **кластеров** – сообществ, которые в большей степени дружат внутри себя, чем с другими людьми. Как вы знаете, большой коллектив всегда разбивается на более-менее обособленные (хотя и не совсем изолированные) группы – вот их мы и будем искать.

- Придумайте *связный* граф из 15-30 вершин. Каждая вершина – это человек, а ребро – отношение дружбы. Постарайтесь сделать так, чтобы у вашего графа можно было "визуально" выделить несколько сообществ. Сообщества не должны быть полностью изолированными (всё-таки граф связный), но будет хорошо, если они будут образовывать заметные кластеры. Вы можете составить граф, основываясь на отношениях внутри вашей группы, или просто придумать красивую картинку.
- Пронумеруйте вершины графа от 1 до n.
- Составьте матрицу Лапласа для вашего графа в соответствии с нумерацией вершин. Найдите (не руками) её собственные числа и соответствующие им собственные вектора.
- Выберите число k желаемых компонент кластеризации графа.
- Возьмите k собственных векторов $v_1, \ldots, v_k$ матрицы Лапласа, соответствующих *самым маленьким* собственным числам. Составьте из них матрицу

$$V = \begin{bmatrix} | & & | \\ v_1 & \ldots & v_k \\ | & & | \end{bmatrix}.$$

  Ширина этой матрицы будет равна количеству компонент кластеризации k, а высота – числу вершин графа n.

- Рассмотрите *строки* составленной матрицы V как точки пространства $\mathbb{R}^k$. Примените к этим точкам любой метод кластеризации (например, метод k-means, реализованный как в MATLAB, так и на Python) для разбиения их на k кластеров.

$$V = \begin{bmatrix} — & x_1 & — \\ & \vdots & \\ — & x_n & — \end{bmatrix} \quad \Longrightarrow \quad \text{разбиваем точки } x_1, \ \ldots, \ x_n \text{ на } k \text{ кластеров.}$$

- Вернитесь к графу и кластеризуйте его, воспользовавшись тем, что вершины пронумерованы. Отнесите вершины с номерами $i, j \in \{1, \ldots, n\}$ к одному кластеру в том случае, если точки x_i и x_j оказались в одном кластере на предыдущем шаге.
- Представьте результат кластеризации в виде раскрашенного графа.
- Выполните кластеризацию для 5-ти разных значений k, сравните результаты.
- Разберитесь *почему это работает*.

Задание 2. Google PageRank алгоритм. В этом задании вы попробуете применить самый первый и исторически важный алгоритм сортировки страниц, который использовался поисковой системой Google – PageRank.

- Придумайте связный ориентированный (направленный) граф из 10-15 вершин и 25-50 стрелок (дуг, рёбер). Каждая вершина – это веб-страница, а стрелка – наличие ссылки, которая позволяет пользователю перейти с одной страницы на другую. Одна вершина может быть соединена с другой сразу несколькими стрелками.
- Пронумеруйте вершины графа от 1 до n.
- Составьте матрицу

$$M = \begin{bmatrix} m_{11} & m_{12} & \ldots & m_{1n} \\ m_{21} & m_{22} & \ldots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \ldots & m_{nn} \end{bmatrix},$$

  где m_{ij} – это отношение числа ссылок на j-й странице, которые ведут на i-ю страницу, к общему числу ссылок на j-й странице.

  Иными словами,

$$m_{ij} = \frac{\text{число стрелочек, выходящих из } j\text{-й вершины и входящих в } i\text{-ю вершину}}{\text{общее число стрелочек, выходящих из } j\text{-й вершины}}.$$

- Найдите собственный вектор матрицы M, соответствующий наибольшему собственному числу.
- Ранжируйте веб-страницы (вершины вашего графа) в соответствии с PageRank-алгоритмом при отсутствии затухания (то есть, при $d = 1$). Представьте результат.
- Разберитесь в логике этого алгоритма. Какой смысл имеет матрица M, почему она составлена именно так, и что она показывает? Как можно интерпретировать собственный вектор этой матрицы, соответствующий наибольшему собственному числу? Почему важен именно этот собственный вектор, а не какой-то другой? Какую роль играет параметр d? Какое отношение всё это имеет к марковским процессам?