Asmt 8: Graphs

Turn in (**a pdf**) through Canvas by 11:59pm, Monday, April 25

Overview

In this assignment we will explore regression techniques on high-dimensional data. We will be using following datasets for this assignment:

- Canvas/Files/Assignments/Graphs/M.csv

For python, we can use the following approach to load the data:

```
import numpy as np
M = np.loadtxt('M.csv', delimiter=',')
```

As usual, it is recommended that you use LaTeX for this assignment. If you do not, you may lose points if your assignment is difficult to read or hard to follow. Find a sample form in: Canvas/Files/Assignments/Assignment Latex Template.zip.

We also suggest you discussing the questions regarding the assignment in the official assignment discussion thread or in the office hours. Don't spam discussions.

If you don't feel confident about your answers, you are suggested to submit your implementation in a separate file (please don't include all the code in your write-up, it will make the write-up difficult to follow). For example, if you use jupyter notebook or google colab, you can submit a separate html file.

1 Finding q_* (100 points)

We will consider four ways to find $q_* = M^t q_0$ as $t \to \infty$.

- `Matrix Power`: choose some large enough value t and create M^t. Then apply $q_* = (M^t)q_0$; there are two ways to create M^t, first we can just let $M^{i+1} = M^i * M$, repeating this process $t-1$ times. Alternatively, (for simplicity assume t is a power of 2), then in $\log_2 t$ steps create $M^{2i} = M^i * M^i$
- `State Propagation`: iterate $q_{i+1} = M * q_i$ for some large enough number t iterations
- `Random Walk`: starting with a fixed state $q_0 = [0, 0, \ldots, 1, \ldots, 0, 0]^T$ where there is only a 1 at the ith entry, and then transition to a new state with only a 1 in the jth entry by choosing a new location proportional to the values in the ith column of M. Iterate this some large number t_0 of steps to get state q_0' (this is the *burn in period*).
  Now make t new step starting at q_0' and record the location after each step. Keep track of how many times you have recorded each location and estimate q_* as the normalized version (recall $||q_*||_1 = 1$) of the vector of these counts.
- `Eigen-Analysis`: Compute `np.linalg.eig(M)` and take the first eigenvector after it has been L_1-normalized.

A: (40 points): Run each method (with $t = 2048$, $q_0 = [1, 0, 0, \ldots, 0]^T$ and $t_0 = 100$ when needed) and report the answers.

B: (20 points): Rerun the `Matrix Power` and `State Propagation` techniques with $q_0 = [0.1, 0.1, \ldots, 0.1]^T$. For what value of t is required to get as close to the true answer as the older initial state?

C: (24 points): Explain at least one **Pro** and **Con** of each approach. The **Pro** should explain a situation when it is the best option to use. The **Con** should explain why another approach may be better for some situation.

D: (8 points): Is the Markov Chain *ergodic*? Explain why or why not.

E: (8 points) Each matrix M row and column represents a node of the graph, label these from 0 to 9 starting from the top and from the left. What nodes can be reached from node 5 in one step, and with what probabilities?

2 Bonus: Taxation (5 points)

Repeat the trials in part **1.A** above using taxation $\beta = 0.85$ so at each step, with probability $1 - \beta$, any state jumps to a random node. It is useful to see how the outcome changes with respect to the results from question 1. Recall that this output is the *PageRank* vector of the graph represented by M. **Briefly** explain what you need to do in order to alter the process in Question 1 to apply this taxation.