EECS 127/227AT Optimization Models in Engineering
Spring 2019 Homework 5

Release date: 10/3/19
Due date: 10/10/19, 23:00 (11 pm). Please LaTeX or handwrite your homework solution and submit an electronic version.

Submission Format
Your homework submission should consist of a single PDF file that contains all of your answers (any handwritten answers should be scanned) as well as your IPython notebook saved as a PDF.

If you do not attach a PDF "printout" of your IPython notebook, you will not receive credit for problems that involve coding. Make sure that your results and your plots are visible. Assign the IPython printout to the correct problem(s) on Gradescope.

1. City migrations as matrix iterations: the transition matrix

In this exercise we are interested in the evolution of the demographics of cities. We represent migration as a transition process between cities inside a graph. This transition process can be represented in a matrix. We are interested in some properties of this matrix that can be exploited to better understand the system.

The exercise includes questions on the iPython notebook "city_migration.ipynb".

We consider that all the cities are part of a big network. The network is represented as a graph which is a set of vertices in $V = \{1, ..., n\}$, with an edge joining any pair of vertices in a set $E \subseteq V \times V$. Every vertex represents a city. There exists an edge between the city i and the city j if there is human migration between the city i and the city j.

We define the transition matrix of the graph T by:

$$T_{ij} = \begin{cases} p_{i,j} & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Where $p_{i,j}$ is the annual percentage of people that leave the city i to go to the city j.

Where $p_{i,i}$ represents the annual percentage of people that stay in the city i.

Because city population is non-negative, and because it is assumed that each city's population can only change from people entering/exiting from/to other cities (i.e. the total population in the network of cities remains constant over years), we have that:

$$\begin{aligned} p_{i,j} &\geq 0, \qquad \forall\ i,j \\ \sum_j p_{i,j} &= 1, \qquad \forall\ i \end{aligned}$$

We denote $m_{i,t} \geq 0$ the population of the city $i \in V$ at the year t. We note $\mathbf{m}_t = (m_{i,t})_{i \in V}$.

(a) Explain why, in this model, the population of every city at the year $t+1$ can be computed as $T^\top \mathbf{m}_t$.

(b) Show that if λ is an eigenvalue of T, then $|\lambda| \leq 1$.
Hint: Consider using $\|\cdot\|_\infty$.

(c) Show that 1 is always an eigenvalue of T. State an eigenvector $\mathbf{v}$ of T associated with the eigenvalue 1, such that $\|\mathbf{v}\|_2 = 1$.

(d) Show that the total population over every city is constant over time: e.g. show that $\|T^\top \mathbf{m}\|_1 = \|\mathbf{m}\|_1$.

(e) Show that finding all the eigenvectors associated with the eigenvalue 1 is equivalent to find $N(I-T)$. We do not consider the case of the zero vector.
Hint: Consider an eigenvector v of T associated with the eigenvalue 1 and show that $v \in N(I-T)$. Then consider $u \in N(I-T)$ and show that u is an eigenvector of T associated with the eigenvalue 1.

(f) Let's consider an eigenvector $\mathbf{v}$ of T associated with the eigenvalue 1. Consider $i = \arg\min_j v_j$ and $\mathbf{u} = \mathbf{v} - v_i\mathbf{1}$. Show that $T\mathbf{u} = \mathbf{u}$, $\mathbf{u} \geq 0$ and $u_i = 0$.

(g) Show that $e_i^\top T^n \mathbf{u} = 0$.

One can show that (the proof can be shown through calculus):

$$e_i^\top T^n \mathbf{u} = \sum_j \left(\sum_{i_1,\ldots,i_n} T_{i,i_1} \left(\prod_{k=1}^{n-1} T_{i_k,i_{k+1}} \right) T_{i_n,j} u_j \right)$$

(h) Show that $\forall j, i_1, \ldots, i_n,\ T_{i,i_1} \left(\prod_{k=1}^{n-1} T_{i_k,i_{k+1}} \right) T_{i_n,j} u_j \geq 0$.

(i) Conclude that $\forall j, i_1, \ldots, i_n,\ T_{i,i_1} \left(\prod_{k=1}^{n-1} T_{i_k,i_{k+1}} \right) T_{i_n,j} u_j = 0$

Definition (strongly connected): A graph is strongly connected if there is a path between any pair of vertices of the graph.

One can also show that if the graph is strongly connected then (the proof is similar to the one for the exercise on the representation of a graph as an adjacency matrix):

$$\forall j, \exists n, \exists i_1, \ldots, i_n,\ T_{i,i_1} \left(\prod_{k=1}^{n-1} T_{i_k,i_{k+1}} \right) T_{i_n,j} \neq 0$$

Now let assume that the graph is strongly connected. We are interested to now all the possible eigenvectors associated with the eigenvalue 1.

(j) Show that $\forall j, u_j = 0$.

(k) Conclude that $\mathbf{v} = v_i\mathbf{1}$.

(l) State what is $N(I-T)$ when the graph is strongly connected.

Now let assume that T is symmetric.

(m) Explain what is the meaning of a symmetric transition matrix for the city's population.

Now let assume that the graph is strongly connected, that T is symmetric and that -1 is not an eigenvector of T.

(n) Show that $\lim_{l\to\infty} T^l = \frac{\mathbf{1}\mathbf{1}^\top}{n}$.
Hint: use answers of question b., c. and l.

(o) Conclude that if T is constant over time, if T is symmetric, if the graph of the cities is strongly connected, and if -1 is not an eigenvalue of T, then the population of every city will be the same after a given time.

(p) The exercise continues on the iPython notebook "city_migration.ipybn". Please answer the question ask there.

Bonus. Show that the previous answer is false if Stanford and Berkeley are part of the cities considered.

Some of you might have recognize that p can be interpreted as a probability, and that the transition matrix can be interpreted as a stochastic matrix. The equation $\mathbf{m}_{t+1} = T^\top \mathbf{m}_t$ encodes a Markov process.

If you enjoyed this exercise feel free to take a look at:

- For the maths lovers
  - The Perron-Frobenius theorem: `https://en.wikipedia.org/wiki/PerronFrobenius_theorem#Applications`
  - Stochastic matrix on Wikipedia: `https://en.wikipedia.org/wiki/Stochastic_matrix`
- For the sociologists
  - Bonacich, P. (1987) Power and Centrality: A Family of Measures, American Journal of Sociology, 92, 1170 82.
  - Freeman, L. C. (1978/79) Centrality in Social Networks: Conceptual Clarification, Social Networks, 1, 215 39.

2. PCA and voting data In this problem, we look at Senate voting data. The data is contained in a $n \times m$ data matrix X, where each row corresponds to a Senator, and each column to a bill. Each entry of X is either $1, -1$ or 0 depending on whether the senator voted for, against or abstained.

(a) We want to assign a score to each senator based on their voting pattern. For this let us pick $a \in \mathbb{R}^m$, and a scalar b and define the score for senator i as:

$$f(x_i, a, b) = x_i^\top a + b, \quad i = 1, 2, \ldots, n.$$

Note that in our notation the rows of X are $x_i^\top$ and thus each $x_i^\top$ is a row vector of length m.

Let us denote by $f(X, a, b)$, the column vector of length n, obtained by stacking the scores for each senator. Then,

$$z = f(X, a, b) = Xa + b\mathbf{1} \in \mathbb{R}^n$$

where $\mathbf{1}$ is a vector with all entries equal to 1. Let us denote the mean value of z by $\mu_z = \frac{1}{n}\mathbf{1}^\top z$. Further let $\mu_x^\top$ denote the row vector corresponding to mean of each column of X. Then

$$\begin{aligned}\mu_z &= \frac{1}{n}\sum_{i=1}^{n} f(x_i, a, b) \\ &= a^\top \mu_x + b.\end{aligned}$$

Then the empirical variance of the scores can be obtained as:

$$\begin{aligned}Var(f(X, a, b)) &= Var(z) \\ &= \frac{1}{n}(z - \mu_z\mathbf{1})^\top(z - \mu_z\mathbf{1}) \\ &= \frac{1}{n}(Xa + b\mathbf{1} - a^\top\mu_x\mathbf{1} - b\mathbf{1})^\top(Xa + b\mathbf{1} - a^\top\mu_x\mathbf{1} - b\mathbf{1}) \\ &= \frac{1}{n}(Xa - \mathbf{1}\mu_x^\top a)^\top(Xa - \mathbf{1}\mu_x^\top a) \\ &= \frac{1}{n}a^\top(X - \mathbf{1}\mu_x^\top)^\top(X - \mathbf{1}\mu_x^\top)a\end{aligned}$$

We observe that the variance of score functions depends on "centered data" and does not depend on b.

For the remainder of the problem we assume that the data has been "centered" (i.e mean of each column of X is zero) and set $b = 0$ so that $\mu_z = 0$.
This leads us to the simpler formula,

$$Var(f(X, a)) = \frac{1}{n}a^\top X^\top Xa.$$

Suppose we restrict a to have unit-norm. Then,find a that maximizes $Var(f(X, a))$. What is the value of the maximum variance?

(b) From the previous part what can you say about how senators vote as compared to their party average? Compute the variance of the scores and comment on the projections along a for the following two cases:

i. a = a_mean_red, the average of rows of X corresponding to 'Red' senators.

ii. a = a_mean_blue, the average of rows of X corresponding to 'Blue' senators.

(c) We can compute the variance of scalar projections, z, of the the data points (rows of X) along a unit direction u as,

$$Var(z) = u^\top C u,$$

where $C = \frac{X^\top X}{n}$. Can you express the variance along the first two principal components of PCA, a_1, a_2 in terms of eigenvalues of C? What is the sum of variance along a_1 and a_2. Plot the data projected on the plane spanned by a_1 and a_2.

(d) Suppose we want to find the bills that are most/least contentious. That is bills for which there is most variability in voting pattern among senators and bills for which the voting is almost unanimous. We can do this in the following two ways:

i. For each basis vector associated with a bill (That is vector of all zeros with 1 in the entry corresponding to the bill), we can compute score using the basis vector as a, and look at variance of the score. This is equivalent to looking at variance of columns of X.

ii. We can look at those bills corresponding to highest and lowest absolute values in the first principal component. This is equivalent to taking inner-products of the first principal component with each of the basis vectors corresponding to the bill and picking the ones with highest/lowest absolute value of inner-products.

Fill in the code in the Jupyter notebook and comment on your observations in the space provided in the notebook.

(e) Finally we can classify senators as most/least "extreme" based on the absolute value of their scores along the first principal component. Comment on your observations in the space provided in the Jupyter notebook.

3. PCA and face recognition

One observation from the face dataset analysis exercise in HW3 (feel free to look over HW3 solutions) is that we may be able to take advantage of our analysis for developing a simple face recognition algorithm. Recall from HW3, that we let $x_i \in \mathbf{R}^m, i = 1, \ldots, n$, be the given data points (images represented as vectors) to analyze, and denoted $\hat{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, the mean of the data points with $\tilde{x_i} = x_i - \hat{x}$, the centered datapoint. In addition,

$$\tilde{X} = \begin{bmatrix} \leftarrow \tilde{x}_1^T \rightarrow \\ \leftarrow \tilde{x}_2^T \rightarrow \\ \vdots \\ \leftarrow \tilde{x}_n^T \rightarrow \end{bmatrix} \in \mathbf{R}^{n \times m}$$

is the matrix such that the i-th row is $\tilde{x}_i^T$.

We also observed that the direction of the i-th greatest variation corresponded to the right singular vector corresponding the i-th singular value (arranged in descending order) of $\tilde{X}$.

The idea for face recognition is this, we can approximately describe any face image from our dataset by subtracting the mean and projecting it onto the directions of maximal variation. Now, given a mean subtracted (centered) data point $\tilde{x}$ and the unit norm direction of i-th maximal variation v_i, this projection is the quantity:

$$\tilde{\alpha}_i(\tilde{x})v_i \coloneqq (\tilde{x}^T v_i)v_i$$

It is known as the i-th principal component of the data point $\tilde{x}$.
Given a datapoint $\tilde{x}$ and k directions of maximal variation $v_i,\ i = 1, \ldots, k$ we may project this datapoint onto all the k directions.
Let

$$\tilde{\alpha}(\tilde{x}) \coloneqq \begin{bmatrix} \tilde{\alpha}_1(\tilde{x}) \\ \tilde{\alpha}_2(\tilde{x}) \\ \vdots \\ \tilde{\alpha}_k(\tilde{x}) \end{bmatrix} \in \mathbf{R}^k$$

be a vector containing all these projections for a given $\tilde{x}$.

(a) Given the mean of the datapoints $\hat{x}$, the k directions of maximal variation $\{v_i,\ i = 1, \ldots, k\}$ and a data point x drawn from the same distribution as the dataset, derive a matrix vector equation for $\tilde{\alpha}(x)$. (*Hint*: You should first write an equation for $\tilde{\alpha}(\tilde{x})$ and then write $\tilde{x}$ as a function of x.)

(b) Implement this function in the Ipython notebook for this section.

(c) For any data point x, we may think of $\tilde{\alpha}(x)$ as a description of that data point. Consequently, to compare two data points, x_a, x_b, we can compare $\tilde{\alpha}(x_a)$ and $\tilde{\alpha}(x_b)$. Why is this more efficient than comparing just x_a, x_b? (Hint, typically $k \ll m$).

(d) For face recognition, the equality of two image matrices is sufficient but not necessary to imply that both images are of the same face. For example, if you take a picture frowning and another smiling, it is still your face even though the image matrices may not be the same.

By projecting the data on only the top k components, we hope to be able to discard unimportant information about a face and thus obtain a more robust way of comparing faces. In the Ipython notebook, run the first two examples (you should uncomment the lines to update the filenames, one at a time) to find the most similar face in the dataset to the query images. Note: The query images were not present in the dataset.

(e) Run it again for the next two images. In which cases does this face recognition algorithm work? Why would it not work sometimes? Note: There are more advance techniques for better face detection using deep learning that go beyond the limitations of our simple algorithm.

4. Traffic flow: the return of the transition matrix

In this exercise, we use the transition matrix to describe the traffic flow evolution inside a road network.

We are given a graph as a set of vertices in $V = \{1, ..., n\}$, with an edge joining any pair of vertices in a set $E \subseteq V \times V$. We define the transition matrix of the graph T by:

$$T_{ij} = \begin{cases} p_{i,j} & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Where $p_{i,j}$ is the probability to go from i to j:

$$\begin{aligned} p_{i,j} &\geq 0, \qquad \forall\ i,j \\ \sum_j p_{i,j} &= 1, \qquad \forall\ i \end{aligned}$$

In a graph where vertices represent road segment and edges represent intersections, we denote $f_i \geq 0$ the traffic flow on the road segment $i \in V$ at the time 0.

Remark: In the graph, vertices represent road segment and edges represent intersections. This might be counter-intuitive, as you might think about the road network where road segment are edges or links of the network. The graph representation where vertices represent road segment of the road network is called line graph of the road network.

We denote $p_{i,j}f_i$ the traffic flow on the road segment i that will go to the road segment j at the intersection (i,j) during the time step δt. The number of people staying on the road segment is then $p_{i,i}f_i$.

(a) Explain why if there is no input flow then the flow at the time δt will be $\mathbf{f_{\delta t}} = T^\top \mathbf{f_0}$.

Now let's assume that at each time step some people enter the road network and some people exit the road network. We define D as the equilibrium demand matrix of the road network. D_{ij} denotes the number of people entering i at each time step intending to exit at j. This means at the equilibrium, D_{ij} people enter road i in order to travel to road j and D_{ij} leave road j after having travelled from road i at every time step (time step of δt).

The amount of people entering the network is equal to the amount of people exiting the network at every time step; in this sense, D is the equilibrium demand matrix.

We are interested in the finding the flow at time $t + \delta t$ as a function of the demand D and of the flow at time t. We assume that the equilibrium is reached.

(b) Show that $\mathbf{f_{t+\delta t}} = T^\top \mathbf{f_t} + (D - D^\top)\mathbf{1}$.

Now we are interested in finding the equilibrium flow: f_{eq} such that $f_{eq} = T^\top f_{eq} + (D - D^\top)\mathbf{1}$.

(c) Show that finding $\mathbf{f_{eq}}$ is a solution of a linear equation $A\mathbf{x} = \mathbf{b}$. State A and $\mathbf{b}$.

(d) Show that there is a solution if and only if $\mathbf{b} \perp N(A^\top)$.

(e) Assuming that the graph is strongly connected, we know that $N(I - T) = Span(\mathbf{1})$ (c.f. first exercise on the transition matrix). Then, show that there exists a equilibrium flow if $\mathbf{1}^\top \mathbf{b} = 0$.

(f) Show that $\mathbf{1}^\top \mathbf{b} = 0$.

(g) Explain why $\mathbf{1}^\top \mathbf{b} = 0$ encodes the flow conservation inside the network.

(h) Let assume that, if the network is strongly connected, $\exists \mathbf{f}^* > 0$ such that $N(A) = Span(\mathbf{f}^*)$ (Perron-Frobenius theorem). Show that, if the network is strongly connected, there exists a equilibrium flow $\mathbf{f_{eq}}$ such that $\mathbf{f_{eq}} \geq 0$.

Remark: this was the technique used by Google in the beginning of their search engine to compute the page rank. $i \in V$ is a website, $e \in E$ is an hyperlink between two website, and $p_{i,j}$ is the probability to click on the hyperlink on the website $i \in V$ that will lead you to the website $j \in V$. $\mathbf{f_{eq}}$ is then the ranking of the websites.

Now, let assume that the network contains no cycle.

(i) Show that the network is not strongly connected.

One can show that if the network contains no cycle then $(T^\top)^n = 0$

(j) Show that 1 is not an eigenvalue of T.

(k) Show that $(I - T^\top)(\sum_{k=0}^{n-1} (T^\top)^k) = I$.

(l) Show that in this case $\mathbf{f_{eq}}$ exists and $\mathbf{f_{eq}} = (\sum_{k=0}^{n-1} (T^\top)^k)(D - D^\top)\mathbf{1}$.

Remark that we have not shown that $\mathbf{f_{eq}} \geq 0$. This would have required a little bit more questions.

Bonus. With regard to this exercise, explain why it is better to use neural networks without understanding the underlying math.

If you enjoyed this exercise, feel free to take a look at:

- A Google-like model of road network dynamics and its application to regulation and control, E.Crisostomi, S.Kirkland, R.Shorten