Lecture 6: Eigenvalues

CSC 338: Numerical Methods

Ray Wu

University of Toronto

February 15, 2023

Overview

- Eigenvalues review
- Pagerank
- Power Method/Iteration
- Inverse Iteration
- Rayleigh Quotient Iteration

Eigenvalues

- For a real, square matrix A, an eigenpair $(\lambda \in \mathbb{R}, v \in \mathbb{R}^n)$ satisfy

$$Av = \lambda v \tag{1}$$

- If A is non-defective – that is, has n linearly independent eigenvectors – then A can be decomposed into

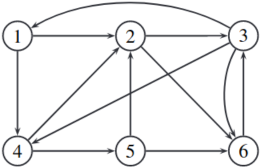
$$A = MDM^{-1} \tag{2}$$

where M contains the eigenvectors and the diagonal entries of D the corresponding eigenvalues.

- The **singular values** of a general matrix A are the square roots of the eigenvalues of $A^T A$.

Motivating example: PageRank

- Consider a web of n webpages, and we want to judge which webpages are the most important.



- Represent the importance of each webpage as a real number in a vector x, with all entries positive: $x_i > 0$
- Only the relative sizes of x_i matter, so we can normalize: $\sum x_i = 1$.
- Assume that from an arbitrary webpage, a user is equally likely to click on any link.
- We want to find a vector that satisfies $Ax = x$, where A is the probability transition matrix.

Example of PageRank

- Probability transition matrix:

$$A = \begin{bmatrix} 0 & 0 & 1/3 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 1/2 & 1/2 & 0 \\ 0 & 1/2 & 0 & 0 & 0 & 1 \\ 1/2 & 0 & 1/3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 1/2 & 1/3 & 0 & 1/2 & 0 \end{bmatrix} \tag{3}$$

- Perron-Frobenius Theorem implies that there exists a vector v such that $Av = v$.
- Can also interpret v as a probability distribution: v_j is the percentage of time a user surfing randomly expected to be on page j.
- Our model is a user randomly surfing, we are interested in the average time spent on each page

Issues with PageRank

There are two main issues with PageRank that need correcting:

- Dangling Node
- Cyclic path/Disjoint component

Dangling Node

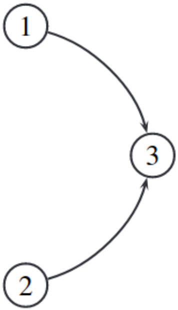


Figure 1: A dangling node

Correcting for Dangling Nodes

- The issue with dangling nodes is that once we enter the node, we cannot leave.
- In our matrix A, this would be a zero column.
- It can be a very important webpage, for example: https://laws-lois.justice.gc.ca/eng/const/
- It can also be an unimportant webpage, for example: http://www.cs.toronto.edu/ rwu/csc338/danglingpage.html
- Solution: assume the user jumps randomly to another page in the network.
- This is equivalent

Cyclic Components

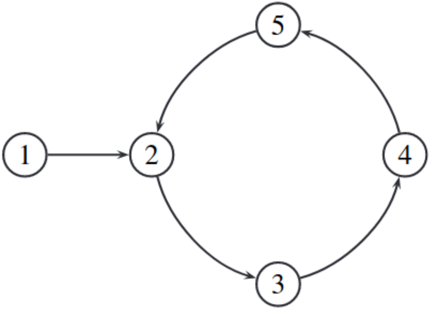


Figure 2: A terminal-strong cyclic component

Correcting Cyclic Components

- The issue here is similar; once we are in a cyclic component, we are not leaving.
- One example could be the documentation for a programming language.
- One solution is to replace A with a rank-1 update:

$$\alpha A + (1 - \alpha) u e^T \tag{4}$$

- convex combimation of A and ue^T (outer product).
- e is all ones, and u is a personalization vector that sums to 1.
- α indicates what weight we give to the links, and what weight we give to the user.
- Choice of α: modelling problem.
- Smaller α means faster convergence, because non-dominant eigenvalues are bounded by α.

Power Method

- The Power Method starts from an initial guess of an eigenvector v_0 and iteratively computes $v_{k+1} = Av_k$, and normalizes v_{k+1}.
- After termination, v approximates the normalized dominant eigenvector, and the corresponding eigenvalue λ is given by the Rayleigh quotient $v^T Av / v^T v$.
- Ensures convergence if the dominant eigenvalue is unique (has only one linearly independent corresponding eigenvector).
- Suppose A has n linearly independent eigenvectors x_i, with the dominant eigenvector being x_1.
  - Then, we can write $v = \sum \beta_i x_i$. As long as $\beta_1 > 0$, we have

$$Av = \sum \beta_i \lambda_i x_i \tag{5}$$

  - For k iterations we have

$$A^k v = \sum \beta_i \lambda_i^k x_i \tag{6}$$

Power Method continued

- Recall from the previous slide, after k iterations we have

$$A^k v = \sum \beta_i \lambda_i^k x_i \tag{7}$$

- Now, recall that we normalized every iteration: let γ_k be a normalization factor:

$$v_k = \gamma_k \lambda_1^k \sum \beta_i (\frac{\lambda_i}{\lambda_1})^k x_i = \gamma_k \lambda_1^k \beta_1 x_1 + \gamma_k \lambda_1^k \sum_{i=2}^{n} \beta_i (\frac{\lambda_i}{\lambda_1})^k x_j \tag{8}$$

- Since λ_1 is the dominant eigenvalue, the remaining terms converge to 0.
- Therefore,

$$v_k \to \gamma_k \lambda_1^k \beta_1 x_1 = x_1 \tag{9}$$

Issues with the Power Method – slow convergence

- Consider two matrices,

$$A = \begin{bmatrix} 32 & 0 \\ 0 & 31 \end{bmatrix}, \quad B = \begin{bmatrix} 32 & 0 \\ 0 & 30 \end{bmatrix} \tag{10}$$

- Of course, the dominant eigenvalues of A and B are 32, with corresponding eigenvector $[1, 0]^T$.
- The convergence rate is given by

$$-\log(\frac{\lambda_2}{\lambda_1}) = -\log\frac{31}{32} = 0.0138 \text{ or } -\log\frac{30}{32} = 0.028 \tag{11}$$

  if they are almost the same size then convergence may be slow.
- The **inverse iteration** addresses this issue of slow convergence – at the cost of solving a linear system of equations.

Inverse Iteration

- The idea of the inverse iteration is: If the eigenvalues of A are λ_i, then the eigenvalues of $A - \alpha I$ are $\lambda_i - \alpha$, and the eigenvalues of $B = (A - \alpha I)^{-1}$ are

$$\mu_j = \frac{1}{\lambda_i - \alpha} \tag{12}$$

- Convergence rate is

$$|\frac{\lambda_1 - \alpha}{\lambda_2 - \alpha}| \tag{13}$$

- We essentially run the power method on B by alternating between normalization and solving the linear system

$$(A - \alpha I)v_{k+1} = v_k \tag{14}$$

- Difficulties: Choosing a value of α close to the dominant eigenvalue.
- Advantages: Can use this method to find any eigenvalue λ_i, not just the dominant one (pick α close to λ_i)

Rayleigh Quotient Iteration

- Recall the inverse iteration:

$$(A - \alpha I)v_{k+1} = v_k \tag{15}$$

- No reason to use the same α on each iteration.
- Rayleigh Quotient Iteration chooses α to be the estimated value of the eigenvalue λ_1.
- λ_1 is estimated with the Rayleigh Quotient $v^T A v / v^T v$.