THE UNIVERSITY OF TEXAS AT AUSTIN
Department of Statistics and Data Sciences
College of Natural Sciences

SDS 385: Stat Models for Big Data

Lecture 10: Pagerank and related methods

Purnamrita Sarkar
Department of Statistics and Data Science
The University of Texas at Austin

https://psarkar.github.io/teaching

Ranking and Pagerank

- Goal: obtain a ranking of webpages which are connected via hyperlinks
- Hope: webpages pointed to by other “important” webpages are also important.
- Developed by Brin and Page (1999)
- Many subsequent works:
    - HITS (Kleinberg, 1998)
    - Pagerank (Page and Brin, 1998)

Definitions

- $n \times n$ Adjacency matrix A
    - A_{ij} = weight on an edge from i to j
    - If graph is undirected $A(i,j) = A(j,i)$
- $n \times n$ Probability transition matrix P
    - P has rows summing to one, i.e. row stochastic
    - $P(i,j)$ is the probability that a random walker will step on j from i.
    - $P(i,j) = \dfrac{A(i,j)}{\sum_j A(i,j)}$
- $n \times n$ Laplacian matrix L
    - $L = D - A$, where D is the diagonal matrix of degrees
    - It is symmetric positive semidefinite for undirected graphs.
    - Singular, i.e. has a zero eigenvalue

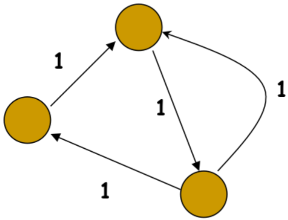
Definition

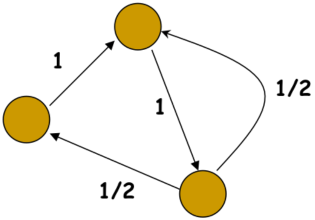
0	1	0
0	0	1
1	1	0

Adjacency matrix A

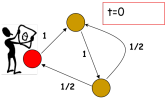
0	1	0
0	0	1
1/2	1/2	0

Transition matrix P

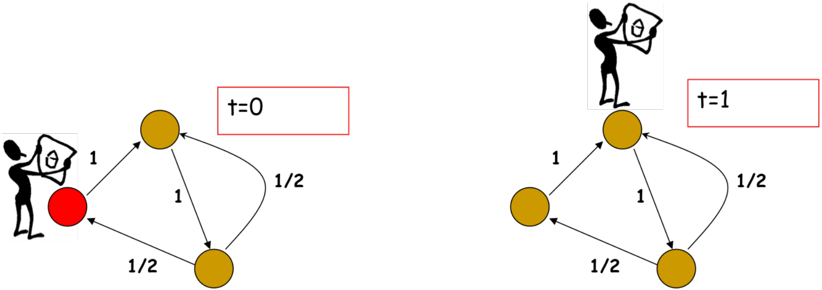




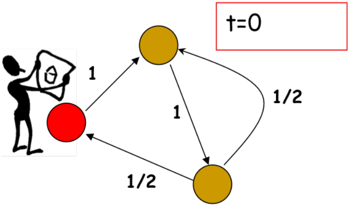
Random walks

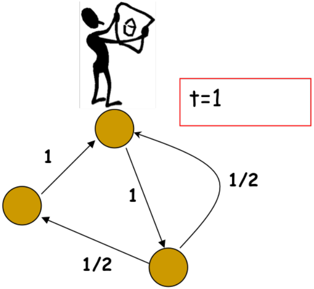


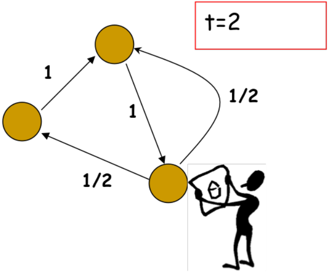
Random walks



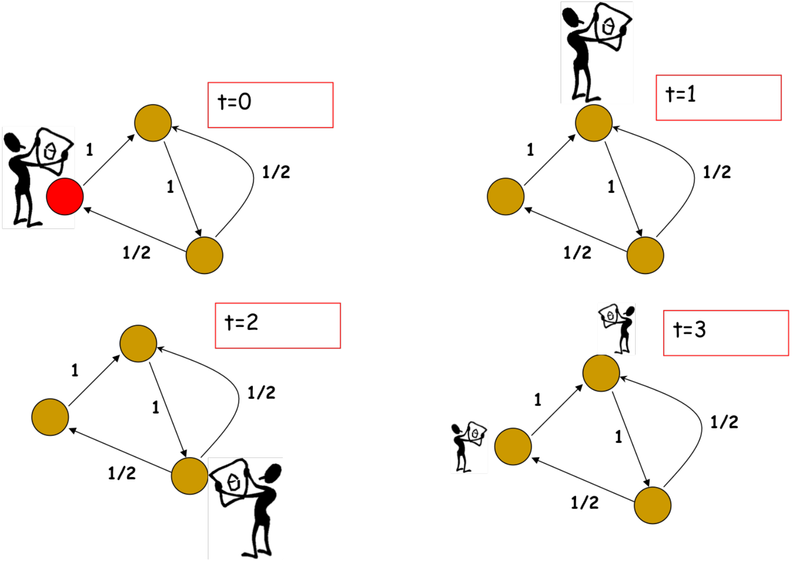
Random walks







Random walks



Probability Distributions

- $x_t(i)$ denotes the probability that the surfer is at node i at time t.

$$x_{t+1}(i) = \sum_j x_t(j) P(j,i)$$

$$x_{t+1}^T = x_t^T P = x_{t-1}^T P^2 = \cdots = x_0^T P^t$$

- What happens if the surfer keeps walking for a long time?

Stationary Distribution

- When the surfer keeps walking for a long time
- When the distribution does not change anymore i.e. $x_{T+1} = x_T$
- For well-behaved graphs this does not depend on the start distribution!!

Stationary distribution

- The stationary distribution at a node is related to the amount of time a random walker spends visiting that node.

Stationary distribution

- The stationary distribution at a node is related to the amount of time a random walker spends visiting that node.
- Remember that we can write the probability distribution at a node as

$$x_{t+1}^T = x_t^T P$$

Stationary distribution

- The stationary distribution at a node is related to the amount of time a random walker spends visiting that node.
- Remember that we can write the probability distribution at a node as

$$x_{t+1}^T = x_t^T P$$

- For the stationary distribution v0 we have

$$v_0^T = v_0^T P$$

Stationary distribution

- The stationary distribution at a node is related to the amount of time a random walker spends visiting that node.
- Remember that we can write the probability distribution at a node as

$$x_{t+1}^T = x_t^T P$$

- For the stationary distribution v0 we have

$$v_0^T = v_0^T P$$

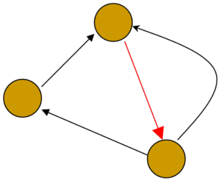
- Whoa! thats just the left eigenvectorof the transition matrix !

Stationary distribution

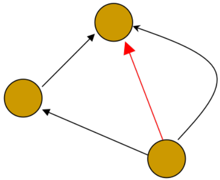
- Lot of theory hiding here.
- For example, what is the guarantee that there will be a unique left eigenvector, or the random walk will at all converge?
- Can't it just keep oscillating?

Well-behaved Markov chains

- Irreducible: There is a path from every node to every other node.



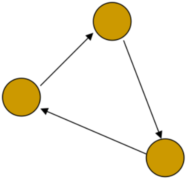
Irreducible



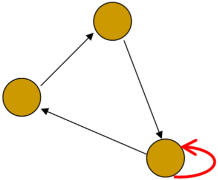
Not irreducible

Well-behaved Markov chains

- Aperiodic: The GCD of all cycle lengths is *1.* The GCD is also called period.



Periodicity is 3



Aperiodic

Well-behaved Markov chains

- If a markov chain is irreducible and aperiodic then the largest eigenvalue of the transition matrix will be equal to 1 and all the other eigenvalues will be strictly less than 1.
- These results imply that for a well behaved graph there exists an unique stationary distribution.

Pagerank and Perron Frobenius

- Perron Frobenius only holds if the graph is irreducible and aperiodic.
- But how can we guarantee that for the web graph? Do it with a small restart probability c.
- At any time-step the random surfer
  - jumps (teleport) to any other node with probability c
  - jumps to its direct neighbors with total probability $1 - c$.

$$\tilde{P} = (1 - c)P + c\mathbb{1}\mathbb{1}^T/n$$

Power iteration

- Power Iteration is an algorithm for computing the stationary distribution.
  - Start with any distribution x_0
  - Keep computing $x_{t+1}^T = x_t^T P$
  - Stop when x_{t+1} and x_t are almost the same

Power Iteration

- Why should this work?
- Write x_0 as a linear combination of the left eigenvectors $\{v_0, v_1, \ldots, v_{n-1}\}$ of P
- Remember that v_0 is the stationary distribution.

$$x_0 = c_0 v_0 + c_1 v_1 + c_2 v_2 + + c_{n-1} v_{n-1}$$

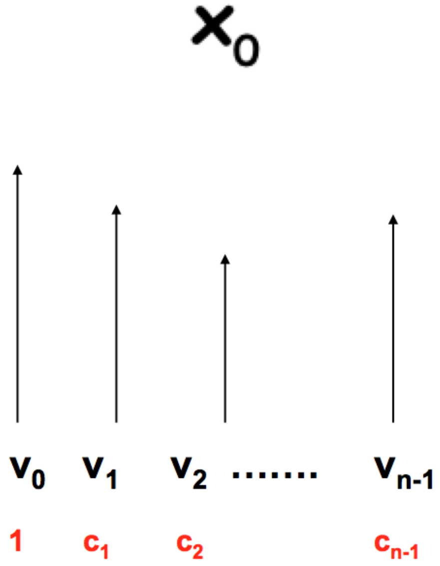
Power Iteration

- Why should this work?
- Write x_0 as a linear combination of the left eigenvectors $\{v_0, v_1, \ldots, v_{n-1}\}$ of P
- Remember that v_0 is the stationary distribution.

$$x_0 = c_0 v_0 + c_1 v_1 + c_2 v_2 + \quad + c_{n-1} v_{n-1}$$

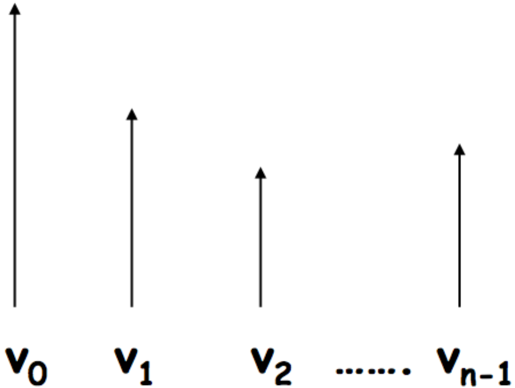
- $c_0 = 1$. Why?
  - First note that $1^T v_i = 0$ if $i \neq 1$
  - So $x_0^T 1 = c_0 = 1$, since both x_0 and v_0 are distributions.

Power Iteration



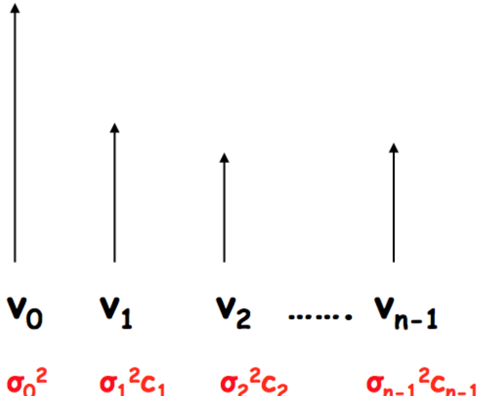
Power Iteration

$$x_1 = x_0 \tilde{P}$$



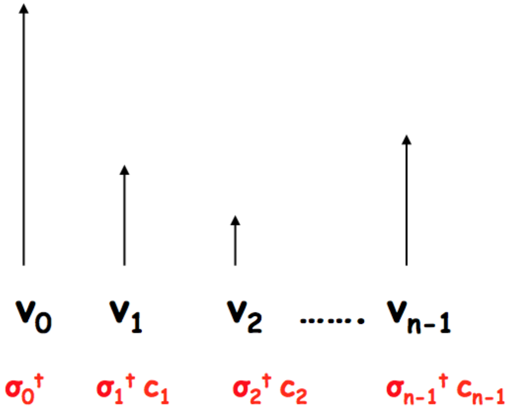
Power Iteration

$$x_2 = x_1 \tilde{P} = x_0 \tilde{P}^2$$



Power Iteration

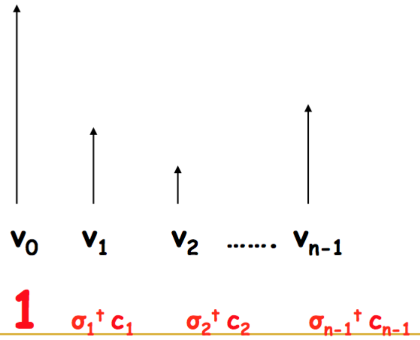
$$x_t = x_0 \tilde{P}^t$$



Power Iteration

$$x_t = x_0 \tilde{P}^t$$

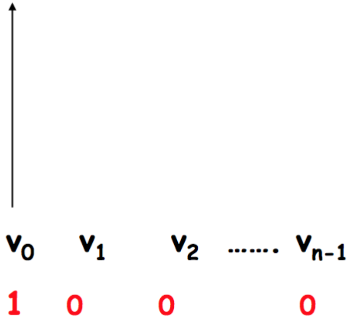
$\boxed{\sigma_0 = 1 > \sigma_1 \geq \ldots \geq \sigma_n}$



Power Iteration

x_∞

$\boxed{\sigma_0 = 1 > \sigma_1 \geq \ldots \geq \sigma_n}$



Second eigenvalue

- Smaller σ_2 is faster the chain mixes.
- For pagerank, we wonder what the second largest eigenvalue is of $\tilde{P} = (1 - c)P + cU$
- The largest eigenvalue is 1
- The second largest is less than $1 - c$ in magnitude.
- So pagerank computation converges fast.

Pagerank

- We are looking for the vector v s.t.

$$v^T = (1-c)v^T P + cr^T$$

- r is a distribution over web-pages.
- If r is the uniform distribution we get pagerank.
- What happens if r is non-uniform?

Pagerank

- We are looking for the vector v s.t.

$$v^T = (1-c)v^T P + cr^T$$

- r is a distribution over web-pages.
- If r is the uniform distribution we get pagerank.
- What happens if r is non-uniform?
- Personalization

Personalized Pagerank

- The only difference is that we use a non-uniform teleportation distribution, i.e. at any time step teleport to a set of webpages.
- In other words we are looking for the vector v s.t.

$$v^T = (1 - c)v^T P + cr^T$$

- r is a non-uniform preference vector specific to an user.
- v gives personalized views of the web.

Personalized Pagerank

- Pre-computation: r is not known from before
- Computing during query time takes too long
- A crucial observation1 is that the personalized pagerank vector is linear w.r.t r

$$\mathbf{r} = \begin{pmatrix} \alpha \\ 0 \\ 1-\alpha \end{pmatrix} \Rightarrow v(r) = \alpha v(r_0) + (1-\alpha) v(r_2)$$

$$\mathbf{r}_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathbf{r}_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

- Lots of literature for computing personalized pagerank fast, and on the go.

Rank Stability

- Pre-computation: r is not known from before
- Computing during query time takes too long
- A crucial observation1 is that the personalized pagerank vector is linear w.r.t r

$$\mathbf{r} = \begin{pmatrix} \alpha \\ 0 \\ 1-\alpha \end{pmatrix} \Rightarrow v(r) = \alpha v(r_0) + (1-\alpha) v(r_2)$$

$$\mathbf{r}_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathbf{r}_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

- Lots of literature for computing personalized pagerank fast, and on the go.

Rank Stability

- How does the ranking change when the link structure changes?
- The web-graph is changing continuously.
- How does that affect page-rank?

Rank Stability

Rank on the entire database.

Rank on 5 perturbed datasets by deleting 30% of the papers

1	"Genetic Algorithms in Search, Optimization and...", Goldberg	1	1	1	1	1
2	"Learning internal representations by error...", Rumelhart+al	2	2	2	2	2
3	"Adaptation in Natural and Artificial Systems", Holland	3	5	6	4	5
4	"Classification and Regression Trees", Breiman+al	4	3	5	5	4
5	"Probabilistic Reasoning in Intelligent Systems", Pearl	5	6	3	6	3
6	"Genetic Programming: On the Programming of ...", Koza	6	4	4	3	6
7	"Learning to Predict by the Methods of Temporal ...", Sutton	7	7	7	7	7
8	"Pattern classification and scene analysis", Duda+Hart	8	8	8	8	9
9	"Maximum likelihood from incomplete data via...", Dempster+al	10	9	9	11	8
10	"UCI repository of machine learning databases", Murphy+Aha	9	11	10	9	10
11	"Parallel Distributed Processing", Rumelhart+McClelland	-	-	-	10	-
12	"Introduction to the Theory of Neural Computation", Hertz+al	-	10	-	-	-

1. **Link analysis, eigenvectors, and stability,** Andrew Y. Ng, Alice X. Zheng and Michael Jordan, IJCAI-01
2. **Automating the contruction of Internet portals with machine learning,** A. Mc Callum, K. Nigam, J. Rennie, K. Seymore, In Information Retrieval Journel, 2000

66

Rank Stability

Rank on the entire database.

Rank on 5 perturbed datasets by deleting 30% of the papers

1	"Genetic Algorithms in Search, Optimization and...", Goldberg	1	1	1	1	1
2	"Learning internal representations by error...", Rumelhart+al	2	2	2	2	2
3	"Adaptation in Natural and Artificial Systems", Holland	3	5	6	4	5
4	"Classification and Regression Trees", Breiman+al	4	3	5	5	4
5	"Probabilistic Reasoning in Intelligent Systems", Pearl	5	6	3	6	3
6	"Genetic Programming: On the Programming of ...", Koza	6	4	4	3	6
7	"Learning to Predict by the Methods of Temporal ...", Sutton	7	7	7	7	7
8	"Pattern classification and scene analysis", Duda+Hart	8	8	8	8	9
9	"Maximum likelihood from incomplete data via...", Dempster+al	10	9	9	11	8
10	"UCI repository of machine learning databases", Murphy+Aha	9	11	10	9	10
11	"Parallel Distributed Processing", Rumelhart+McClelland	-	-	-	10	-
12	"Introduction to the Theory of Neural Computation", Hertz+al	-	10	-	-	-

1. **Link analysis, eigenvectors, and stability,** Andrew Y. Ng, Alice X. Zheng and Michael Jordan, IJCAI-01
2. **Automating the contruction of Internet portals with machine learning,** A. Mc Callum, K. Nigam, J. Rennie, K. Seymore, In Information Retrieval Journel, 2000

66

Rank Stability

- Ng et al 2001: $\tilde{P} = (1-c)P + cU$
- Theorem: if v is the left eigenvector of . Let the pages $i_1, i_2, \dots, i_k$ be changed in any way, and let v' be the new pagerank. Then

$$\|v - v'\|_1 \leq \frac{\sum_{j=1}^{k} v(i_j)}{c}$$

Rank Stability

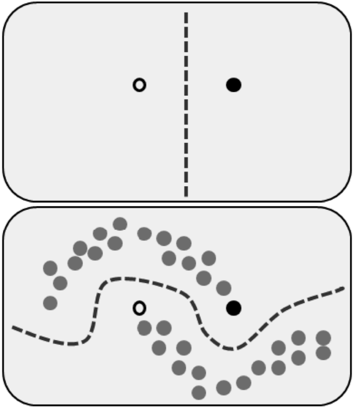
- Ng et al 2001: $\tilde{P} = (1-c)P + cU$
- Theorem: if v is the left eigenvector of . Let the pages $i_1, i_2, \ldots, i_k$ be changed in any way, and let v' be the new pagerank. Then

$$\|v - v'\|_1 \leq \frac{\sum_{j=1}^{k} v(i_j)}{c}$$

- So if c is not too close to 0, the system would be rank stable and also converge fast!

Semi-supervised learning

- You are given a lot of unlabeled data.
- Only a few points are labeled.
- Is this useful?



Semi-supervised learning

- Two broad ways
  - Label propagation:
    - Graph Based algorithm
    - Does not generalize to unseen data, i.e. Transductive
  - Manifold regularization
    - Graph Based regularization
    - Does generalize to unseen data, i.e. Inductive

Semi-supervised learning

- Input n data points $x_1, \ldots, x_n$
- Define similarity matrix $S \in \mathbb{R}^{n \times n}$

$$S_{ij} = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$$

- Since S is dense, often k nearest neighbor graphs are also used. (Your homework!)