Random Walk Fundamental Tensor and Graph Importance Measures

Daniel Boley[1] Alejandro Buendia[2]

[1]University of Minnesota

[2]Microsoft AI Development Acceleration Program, Columbia University

BSMDMA Workshop, IJCAI 2019

Outline

1. Preliminaries
2. Random walk fundamental tensor
3. Example digraph
4. Laplacian and its extension to digraphs
5. Computing the Moore-Penrose pseudoinverse of the random walk Laplacian
6. Computing the random walk fundamental tensor
7. Applications to centrality measures
8. Further reductions in complexity
9. Experiment on small-world graphs

Preliminaries

- Directed graph represented by adjacency matrix $A = [a_{ij}]$ with

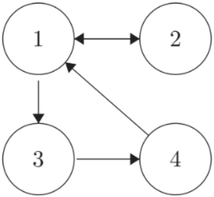
$$a_{ij} = \begin{cases} w_{ij} & \text{weight on directed edge } i \to j \\ 0 & \text{if no edge exists} \end{cases}$$

- Markov chain over digraph has probability transition matrix $P = D^{-1}A$, for diagonal matrix D of vertex out-degrees
- Let the digraph be strongly connected: strongly connected $\iff$ no absorbing states in the Markov chain
- Let $\boldsymbol{\pi}$ be the vector of stationary probabilities for the random walk, scaled to unit length in the 1-norm, and let $\Pi = \mathrm{Diag}(\boldsymbol{\pi})$ be the corresponding diagonal matrix.

Random walk fundamental tensor

- The fundamental matrix $N = [n_{ij}]$ of an absorbing Markov chain gives the expected number of random walk passages through node j starting from node i.
- Golnari et al. [1] introduce the third-order *random walk fundamental tensor* $\boldsymbol{N} = \boldsymbol{N}(i, j, k)$, where entry $\boldsymbol{N}(i, j, k)$ gives the expected number of passages through intermediate node j when starting a random walk from node i absorbed by node k.
- Each slice of the tensor is a fundamental matrix N for a random walk over the digraph, treating node k as the absorbing state.

Example digraph



$$\boldsymbol{N}_{::1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} \quad \boldsymbol{N}_{::3} = \begin{pmatrix} 2 & 1 & 1 & 0 \\ 2 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 2 & 1 & 1 & 1 \end{pmatrix}$$

$$\boldsymbol{N}_{::2} = \begin{pmatrix} 2 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 2 & 1 & 2 & 2 \\ 2 & 1 & 1 & 2 \end{pmatrix} \quad \boldsymbol{N}_{::4} = \begin{pmatrix} 2 & 1 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

We define $\boldsymbol{N}(i, j, k) = 0$ for $i = k \neq j$ and $\boldsymbol{N}(i, j, k) = 1$ for $j = k$.

Laplacian and its extension to digraphs

- Several Laplacians exist in the literature (see e.g. [2])
- Consider a *random walk Laplacian* $\boldsymbol{L} = \boldsymbol{\pi}(I - P)$ with rank $n-1$ and nullity 1 ($\boldsymbol{L} \cdot \mathbf{1} = \mathbf{0}$ and $\mathbf{1}^T \boldsymbol{L} = \mathbf{0}^T$)
- The Moore-Penrose pseudoinverse of L provides an efficient way to compute the random walk fundamental tensor: we show an algorithm using a single matrix inverse of complexity $O(n^3)$ and other lower-order computations of complexity $O(n^2)$.

Computing the Moore-Penrose pseudoinverse of the random walk Laplacian

Algorithm 3 (Computation of pseudoinverse)

1. Compute probability transition matrix $P = D^{-1}A$
2. Compute *normalized Laplacian* $L = I - P$
3. Compute inverse of the upper $(n-1) \times (n-1)$ part of L, $L_{\alpha,\alpha}$
4. Solve the linear system $(\pi_1, \ldots, \pi_{n-1}) = -L_{\alpha,\alpha}^{-1} \boldsymbol{l}_{\alpha,n} \pi_n$, where π_n is scaled so that $\boldsymbol{\pi}$ has unit length
5. Form $\Pi = \mathrm{Diag}(\boldsymbol{\pi})$ and $\boldsymbol{L} = \Pi(I - P)$, partitioned as in (3)
6. Compute the inverse of the upper-left block of $\boldsymbol{L}$, $\boldsymbol{L}_{\alpha,\alpha}^{-1} = (I - P_{\alpha,\alpha}^{-1})\Pi_1^{-1}$, using the previously computed inverse
7. Compute desired pseudoinverse $\boldsymbol{M}$ of $\boldsymbol{L}$ using Lemma 1 from [2], exploiting that the annihilating vectors for $\boldsymbol{L}$ and $\boldsymbol{M}$ are both $\mathbf{1}$

Computing the random walk fundamental tensor

- A corollary of Lemma 1 [2] provides an efficient formula to go from pseudoinverse $\boldsymbol{M}$ to fundamental matrix $N = \boldsymbol{N}(\alpha, \alpha, n)$.
- Elementwise, we can then derive the formula

$$N(i, j, k) = (\boldsymbol{m}_{ij} - \boldsymbol{m}_{kj} - \boldsymbol{m}_{ik} + \boldsymbol{m}_{kk})\pi_j \tag{1}$$

- Computing from scratch, we require one matrix inverse $O(n^3)$ and other $O(n^2)$ operations to compute $\boldsymbol{M}$. The fundamental tensor $\boldsymbol{N}$ can then be computed via (1) in constant time per entry $\boldsymbol{N}(i, j, k)$.

Applications to centrality measures

Hitting times and centrality measures are easily computed from the random walk fundamental tensor and useful in quantifying the importance of nodes within the graph:

- Hitting time: Expected time for a random walk starting at source i to reach k is

$$H(i,k) = \sum_j \boldsymbol{N}(i,j,k)$$

- Random walk closeness [3]:

$$\text{closeness}(k) = \sum_i H(i,k) = \sum_{i,j} \boldsymbol{N}(i,j,k)$$

- Random walk betweenness [4, 5]:

$$\text{betweenness}(j) = \sum_{i \neq j, k \neq j} \Pr(i \to j \to k)$$

Further reductions in complexity

- Cohen et al. [6] show that $\frac{1}{2}(\boldsymbol{L} + \boldsymbol{L}^T)$ can be considered the Laplacian for an undirected graph with the same link structure as the original digraph.
- An approximate sparse LU factorization for $\boldsymbol{L}$ whose fill is linear in the fill of the original $\boldsymbol{L}$ can be found with high probability [7].
- Though a fast exact algorithm for the general matrix inverse does not exist, this leads to an approximate algorithm with complexity slightly over $O(n^2)$ to find an approximation to the inverse of $L_{\alpha,\alpha}$.

Experiment on small-world graphs

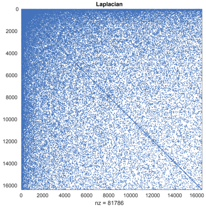
- Certain graph structures naturally lead to fast deterministic algorithms.
- *Small-world networks* are characterized by a high clustering coefficient but small expected path length (e.g. social networks).
- Using preferential attachment, we generate synthetic small-world networks and construct digraphs by randomly deleting edges.
- We compute a deterministic LU factorization with the approximate minimum degree ordering.

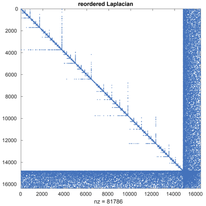
Experiment on small-world graphs (cont.)

Computation of the LU factorization takes $O(n^2)$ time and space, leading to faster generation of the inverse matrix.

number of			time in csec	
vertices	edges	LU fill	LU	backsolve
1,024	4,059	20,620	5	2
2,048	8,140	66,851	2	<1
4,096	16,314	205,826	4	<1
8,192	32,671	763,440	12	1
16,384	65,402	2,804,208	56	5
32,768	130,884	10,740,194	250	19
65,536	261,882	43,504,911	1,363	82
131,072	523,920	168,455,437	7,989	328

Table 1: Cost of Gaussian elimination for a sample of synthetic scale-free graphs using Matlab R2018a.





Bibliography

- G. Golnari, D. Boley, Y. Li, and Z.-L. Zhang, "Pivotality of nodes in reachability problems using avoidance and transit hitting time metrics," in *7th Ann. Wshp on Simplifying Complex Networks for Practitioners*, SIMPLEX, 2015.
- D. Boley, G. Ranjan, and Z.-L. Zhang, "Commute times for a directed graph using an asymmetric Laplacian," *Lin. Alg. & Appl.*, vol. 435, pp. 224–242, 2011.
- J. D. Noh and H. Rieger, "Random walks on complex networks," *Physical Review Letters*, vol. 92, no. 11, 2004.
- U. Kang, S. Papadimitriou, J. Sun, and H. Tong, "Centralities in large networks: Algorithms and observations," in *SIAM Intl Conf Data Mining*, 2011.
- M. E. J. Newman, "A measure of betweenness centrality based on random walks," *Social Net.*, vol. 27, no. 1, pp. 39–54, 2005.
- M. B. Cohen, J. Kelner, J. Peebles, R. Peng, A. Sidford, and A. Vladu, "Faster algorithms for computing the stationary distribution, simulating random walks, and more," in *IEEE 57th Annual Symp. on Found. Comput. Sci. (FOCS)*, pp. 583–592, Oct 2016.
- M. B. Cohen, J. Kelner, R. Kyng, J. Peebles, R. Peng, A. B. Rao, and A. Sidford, "Solving directed laplacian systems in nearly-linear time through sparse LU factorizations." arxiv.org/abs/1811.10722, 2018.