Online Factorization and Partition of Complex Networks From Random Walks

Lin F. Yang lin.yang@princeton.edu
Princeton University
Vladimir Braverman vova@cs.jhu.edu
Johns Hopkins University
Tuo Zhao tourzhao@gatech.edu
Georgia Tech
Mengdi Wang mengdiw@princeton.edu
Princeton University

Abstract

Finding the reduced-dimensional structure is critical to understanding complex networks. Existing approaches such as spectral clustering are applicable only when the full network is explicitly observed. In this paper, we focus on the online factorization and partition of implicit large-scale networks based on observations from an associated random walk. We formulate this into a nonconvex stochastic factorization problem and propose an efficient and scalable stochastic generalized Hebbian algorithm. The algorithm is able to process dependent state-transition data dynamically generated by the underlying network and learn a low-dimensional representation for each vertex. By applying a diffusion approximation analysis, we show that the continuous-time limiting process of the stochastic algorithm converges globally to the “principal components" of the Markov chain and achieves a nearly optimal sample complexity. Once given the learned low-dimensional representations, we further apply clustering techniques to recover the network partition. We show that when the associated Markov process is lumpable, one can recover the partition exactly with high probability. We apply the proposed approach to model the traffic flow of Manhattan as city-wide random walks. By using our algorithm to analyze the taxi trip data, we discover a latent partition of the Manhattan city that closely matches the traffic dynamics.

1 Introduction

Network data arise in many applications and research areas, including but not limited to social science, economics, transportation, finance, power grid, artificial intelligence, etc. Examples include protein-protein interaction networks [9], phone communication networks [15], collaboration networks [4], and the gravitational interaction network of dark matter particles in cosmology [17, 12, 14]. Due to the highly complex nature of these networks, many efforts have been devoted to investigating their reduced-order representations from high-dimensional data (e.g. [6, 18, 16, 5]).

In this paper, we focus on learning from the dynamic “state-transition” data, which are snapshots of a random walk associated with the implicit network. For example, records of taxi trips can be used to reveal the traffic dynamics of a metropolitan. Each trip can be viewed as a fragmented sample path realized from a city-wide Markov chain that characterizes the traffic dynamics [11, 3]. None of the existing works has considered how to recover the latent network partition of an urban area from the taxi trip data. For another example, reinforcement learning applications such as autonomous driving and game AI are modeled as Markov decision processes [20], which unfortunately suffer from the curse of dimensionality of the state space. Given trajectories of game snapshots or a game simulator, it is of vital interest to identify the low-dimensional representation of the “state” of game. For the general problem of finding reduced-order representations, popular approaches such as principal component analysis and spectral clustering do not utilize the Markov nature of state-transition data. Existing computational methods often require explicit knowledge and pre-computation of large matrices, which cannot scale to large-scale problems and is not even possible for online learning applications. Efficient methods are in demand.

OPTML 2017: 10th NIPS Workshop on *Optimization for Machine Learning* (NIPS 2017).

Motivated by the need to analyze state-transition data, we propose an efficient and scalable approach for online factorization and partition of implicit complex networks. We start by employing a stochastic gradient-type algorithm, namely the generalized Hebbian algorithm (GHA), and tailor it towards processing Markov transition data. Then we show that the GHA learns low-dimensional representations of the network in an online fashion, and by further applying clustering techniques, we can recover the underlying partition structure with high probability. Our analysis is based on a diffusion approximation approach, which is widely used in stochastic analysis of complicated discrete processes such as queueing networks (see [7] for more related literature on diffusion approximation). By properly rescaling of time, we approximate the discrete-time dynamics generated by the GHA algorithm using its continuous-time limiting process, which is the solution to an ordinary differential equation (ODE). Though the stochastic optimization problem is highly nonconvex, we show that the limiting stochastic process of the GHA converges geometrically to the global optima, even if the initial solution is chosen uniformly at random. We further show that the process after sufficiently large time is well approximated by an Ornstein-Uhlenbeck process, whose stochastic fluctuation can be precisely characterized. Despite of the spherical geometry and many unstable equilibria of the optimization problem, we establish global convergence with a near-optimal sample complexity guarantee in an asymptotic manner.

Our work is partly motivated by [21], which establishes the connection between networks and a class of *lumpable* Markov chains. It proposes an optimization framework to identify the partition structure when the transition matrix is known *a priori*. Our method is also related to the class of online eigenvalue decomposition methods for representation learning [1, 10, 22, 2, 8]. However, none of the existing methods and analysis are applicable to Markov transition data and online network partition.

Notation: We denote $[n] = \{1, 2, \ldots, n\}$. Given two matrices $\boldsymbol{U} \in \mathbb{R}^{m\times r_1}, \boldsymbol{V} \in \mathbb{R}^{m\times r_2}$ with orthonormal columns, where $1 \leq r_1 \leq r_2 \leq m$, we denote the principle angle between two matrices by $\Theta\left(\boldsymbol{U}, \boldsymbol{V}\right) = \mathrm{diag}\big[\cos^{-1}\left(\sigma_1(\boldsymbol{U}^\top\boldsymbol{V})\right), \cos^{-1}\left(\sigma_2(\boldsymbol{U}^\top\boldsymbol{V})\right), \ldots, \cos^{-1}\left(\sigma_{r_1}(\boldsymbol{U}^\top\boldsymbol{V})\right)\big]$, where $\sigma_i(\boldsymbol{A})$ is the i-th largest singular value of matrix $\boldsymbol{A}$. We also use $\cos(\cdot)$ and $\sin(\cdot)$ to act on matrices and denote entry-wise functions. For a matrix $\boldsymbol{V}$, we denote by $\boldsymbol{V}_{*j}$ its j-th column vector and by $\boldsymbol{V}_{i*}$ its i-th row vector. We denote by $\boldsymbol{V}_{*1:r}$ the sub-matrix of the first r columns. We denote by $\|\cdot\|_{\mathrm{F}}$ the Frobenius norm of a matrix, and denote by $\|\cdot\|_2$ the Euclidean norm of a vector or the spectral norm of a matrix. We denote by $\boldsymbol{e}_i \in \mathbb{R}^s$ the i-th standard unit vector for any $s \geq i$: $(\boldsymbol{e}_i)_i = 1$ and $(\boldsymbol{e}_i)_j = 0$ for $j \neq i$. We also denote by $0_{m\times n} \in \mathbb{R}^{m\times n}$ the matrix with all 0 entries.

Networks and Associated Markov Chains: Let $G = (S, E)$ be a *connected* network with m vertices (a weighted directed graph), where $S = \{s_1, s_2, \ldots, s_m\}$ denotes the vertex set, $E = \{w_{i,j} \geq 0 : i, j \in [m]\}$ denotes the edge set, and $w_{i,j}$ denotes the weight of the edge (s_i, s_j). Consider the random walk that is naturally associated with the network G: We denote by $\boldsymbol{P} = (p_{i,j}) \in \mathbb{R}^{m\times m}$ its probability transition matrix, where each state of the Markov chain corresponds to a vertex in G. Since G is a connected network, all states of the Markov chain are recurrent. The Markov chain generated by the network G satisfies $\mathbb{P}\left[s^{(t)} = s_j \middle| s^{(t-1)} = s_i\right] = p_{i,j}$. Suppose that G is *undirected* (i.e., $w_{ij} = w_{ji}$), then $\forall i, j : p_{i,j} = \frac{w_{i,j}}{w_i}$ and $w_i = \sum_{j\in[m]} w_{i,j}$. The stationary distribution of the Markov chain is $\mu_i = \frac{w_i}{\sum_{j\in[m]} w_j}$. The corresponding Markov chain is *reversible* and satisfies the following *detailed balance condition* $\forall i \neq j,\ \mu_i p_{i,j} = \mu_j p_{j,i}$ and $\sum_{i\in[m]} \mu_i p_{i,j} = \mu_j$, i.e., $\boldsymbol{DP} = \boldsymbol{P}^\top\boldsymbol{D}$, where $\boldsymbol{D} = \mathrm{diag}\left(\mu_1, \mu_2, \ldots, \mu_m\right)$. Note that our subsequent analysis does not require the undirectedness assumption of the underlying network. In this paper, we focus on connected and undirected networks where $\mu_i > 0$ for all $i \in [m]$. For a non-connected network, our method still applies with the caveat that it recovers the structure of a connected component determined by the initial state.

Our Problem of Interest Given a sample trajectory $\{s^{(0)}, s^{(1)}, \ldots, s^{(t)}, \ldots\}$ of state transitions of the unknown Markov chain, our objective is to develop an online learning method to extract reduced-order information about the Markov chain and recover the latent network partition.

We are interested in complex networks that can be approximated using reduced-order representations. To be general, we consider networks with associated Markov chains *nearly low-rank*, which is defined as follows:

Definition 1 (Nearly Low-Rank Markov Chains)**.** *A Markov chain with transition matrix $\boldsymbol{P}$ is nearly low-rank if there exist matrices $\boldsymbol{F}_1, \boldsymbol{F}_2 \in \mathbb{R}^{m\times m}$, where $\mathrm{rank}(\boldsymbol{F}_1) = r$ and $\|\boldsymbol{F}_2\|_2 < \sigma_r\left(\boldsymbol{F}_1\right)$ such that*

$$\boldsymbol{DP} = \boldsymbol{F}_1 + \boldsymbol{F}_2 \quad \textit{and} \quad \boldsymbol{F}_1^\top\boldsymbol{F}_2 = 0_{m\times m}, \tag{1}$$

and $\boldsymbol{F}_1 = \boldsymbol{U\Sigma V}^\top$, *where* $\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$ *is a diagonal matrix with* $1 \geq \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$, *and* $\boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{m \times r}$ *are matrices with orthonormal columns.*

Consider the following *representation matrix* $\boldsymbol{M} := \boldsymbol{D}^{-1}\boldsymbol{V} \in \mathbb{R}^{m \times r}$, each row of which can be viewed as an r-dimensional representation of a vertex of G. The matrix $\boldsymbol{M}$ gives a set of approximate "principal components" of the Markov chain, which has a similar spirit as spectral clustering [6]. Note that Markov chains that are nearly low-rank are not necessarily reversible. When a Markov chain is both nearly low-rank and reversible, the conditions in Definition 1 shall hold with $\boldsymbol{U} = \boldsymbol{V}$.

In particular, we also consider an important special case of nearly low-rank Markov chains - "lumpable" Markov chains, which is introduced by [13] and formally in [21] as follows.

Definition 2 (Special Case: Lumpable Markov chains [21])**.** *A reversible Markov chain on states S with transition matrix $\boldsymbol{P}$ is* lumpable *with respect to the partition $S = S_1 \cup S_2 \ldots \cup S_r$ if the top r eigenvectors of $\boldsymbol{DP}$ are piecewise constant with respect to the $S_1, \ldots, S_r$.*

We can view S_1,...,S_r as"meta states" of the Markov chain. When the lumpability condition holds, the transitions between these sets satisfy the strong Markov property, i.e., for any $s_k, s_h \in S_i$, $\forall\, j$,

$$\sum_{s_\ell \in S_j} p_{k,\ell} = \sum_{s_\ell \in S_j} p_{h,\ell}.$$

Intuitively speaking, the meta states suffice to characterize the macro dynamics of a complex Markov chain. When the Markov chain is lumpable, it is nearly-low rank as in Definition 1 with $\boldsymbol{U} = \boldsymbol{V}$. In this case, the matrix $\boldsymbol{U}$ becomes a block matrix. For any $i, j \in [r]$, the vector $\boldsymbol{U}_{*i}$ restricted on coordinates S_j has constant values across all entries. The work [21] showed when the Markov chain is lumpable with respect to a partition $S = S_1 \cup S_2 \ldots \cup S_r$, one can recover the exact partition by clustering its r-dimensional representations (rows of $\boldsymbol{M} = \boldsymbol{D}^{-1}\boldsymbol{V}$). An example of a network and its lumpable Markov chain is given in the full version of this paper.

2 Method for Dynamic Network Partition

Recall that we are interested in learning from Markov transition data. In particular, consider the scenario where we only observe state-to-state transitions of a Markov process over S: $s^{(1)}, s^{(2)}, s^{(3)}, \ldots, s^{(n-1)}, s^{(n)}$, $\ldots$, without knowing the transition matrix $\boldsymbol{P}$ in advance. For notational convenience, we simplify the notation of the states to $S = \{1, 2, \ldots m\}$.

2.1 A Nonconvex Optimization Model for Markov Chain Factorization

To handle the dependency of the Markov process, we need to downsample the data. Specifically, we divide the trajectory of n state transitions into b blocks with block size τ for some $\tau \geq 2$:

$$\underbrace{s^{(1)}, s^{(2)}, ..., s^{(\tau)}}_{\text{the 1−st block}}, \underbrace{s^{(\tau+1)}, s^{(\tau+2)}, ..., s^{(2\tau)}}_{\text{the 2−nd block}}, \ldots, \underbrace{s^{(b-1)\tau+1}, s^{(b-1)\tau+2}, \ldots s^{(b\tau)}}_{\text{the b−th block}}.$$

For the k-th block, we select the last two samples and construct $\boldsymbol{Z}^{(k)} \in \mathbb{R}^{m \times m}$ to be the matrix with one entry equaling 1 and all other entries equaling 0, i.e.,

$$\boldsymbol{Z}^{(k)}_{s^{(k\tau-1)}, s^{(k\tau)}} = 1 \quad \text{and} \quad \boldsymbol{Z}^{(k)}_{s,s'} = 0 \text{ for all } (s, s') \neq \left(s^{(k\tau-1)}, s^{(k\tau)}\right). \tag{2}$$

Here we choose a large enough τ such that $\forall k \geq 1$, $\mathbb{E}\left[\boldsymbol{Z}^{(k)} \middle| s^{(0)}\right] \approx \boldsymbol{DP} = \boldsymbol{F}_1 + \boldsymbol{F}_2$,where $\boldsymbol{F}_1 = \boldsymbol{U}^\top \boldsymbol{\Sigma V}$ and $\boldsymbol{F}_2$ are given in Definition 1. Intuitively, the choice of τ shall be related to how fast the Markov chain mixes. We will specify the choice of τ in the full version of this paper. Let us formulate the Stochastic Transition Matrix Decomposition Problem as

$$(\boldsymbol{U}^*, \boldsymbol{V}^*) = \underset{\widetilde{\boldsymbol{U}}, \widetilde{\boldsymbol{V}} \in \mathbb{R}^{m \times r}}{\operatorname{argmax}} \operatorname{tr}\left[\widetilde{\boldsymbol{U}}^\top \mathbb{E}\boldsymbol{Z}\widetilde{\boldsymbol{V}}\right] \quad \text{subject to} \quad \widetilde{\boldsymbol{U}}^\top \widetilde{\boldsymbol{U}} = \widetilde{\boldsymbol{V}}^\top \widetilde{\boldsymbol{V}} = I_r. \tag{3}$$

where the expectation $\mathbb{E}\boldsymbol{Z} := \lim_{n \to \infty} n^{-1} \sum_{k=1}^{n} \boldsymbol{Z}^{(k)} = \boldsymbol{DP}$ is taken over the invariant distribution of the Markov chain. Note that $\boldsymbol{U}^*$ and $\boldsymbol{V}^*$ are global optima to (3), and they satisfy $\boldsymbol{U}^* = \boldsymbol{UO}$ and $\boldsymbol{V}^* = \boldsymbol{VO}$ for some orthonormal matrix $\boldsymbol{O} \in \mathbb{R}^{r \times r}$. By using a self-adjoint dilation, we recast (3) into a symmetric decomposition problem as follows

$$\boldsymbol{W}^* = \underset{\boldsymbol{W} \in \mathbb{R}^{2m \times r}}{\operatorname{argmax}} \operatorname{tr}\left[\boldsymbol{W}^\top \mathbb{E}\boldsymbol{A}\boldsymbol{W}\right] \quad \text{subject to} \quad \boldsymbol{W}^\top \boldsymbol{W} = I_r, \tag{4}$$

where $\mathbb{E}\boldsymbol{A} = \begin{bmatrix} 0_{m \times m} & \mathbb{E}\boldsymbol{Z} \\ \mathbb{E}\boldsymbol{Z}^\top & 0_{m \times m} \end{bmatrix} \in \mathbb{R}^{2m \times 2m}$ and $\boldsymbol{W} = \frac{1}{\sqrt{2}}\left[\boldsymbol{U}^\top, \boldsymbol{V}^\top\right]^\top \in \mathbb{R}^{2m \times r}$.

2.2 Algorithm for Online Factorization of Markov Chains

To solve (4), we adopt the Generalized Hebbian Algorithm (GHA) which was originally developed for training neural nets and principal component analysis [19]. GHA, also referred as Sanger's rule, is essentially a stochastic primal-dual algorithm. The k-th iteration of GHA takes the form

$$\boldsymbol{W}^{(k+1)} = \boldsymbol{W}^{(k)} + \eta(\boldsymbol{A}^{(k)}\boldsymbol{W}^{(k)} - \boldsymbol{W}^{(k)}\boldsymbol{W}^{(k)\top}\boldsymbol{A}^{(k)}\boldsymbol{W}^{(k)}).$$

where $\eta > 0$ is the learning rate. Note that the columns of $\boldsymbol{W}^{(k)}$ are not necessarily orthogonal. But when $\boldsymbol{W}^{(0)}$ has orthonormal columns, then $\boldsymbol{W}^{(k)}$ tends to have orthonormal columns as $\eta \to 0$. The formal procedure is presented in Algorithm 1.

Algorithm 1 SGA for Online Factorization of Markov Chains

```
Input: A stream of Markov transition data $s^{(1)}, s^{(2)}, s^{(3)}, \ldots, s^{(n-1)}, s^{(n)}, \ldots$
Initialize: Sample matrix $G \in \mathbb{R}^{2m\times r}$ with i.i.d. entries from $\mathcal{N}(0,1)$;   $\boldsymbol{W}^{(0)} \leftarrow QR(G)$, $k \leftarrow 0$;
Repeat: For every $\tau$ state transitions, obtain $\boldsymbol{A}^{(k)}$ using Eqs. (2),(4);
  $\boldsymbol{W}^{(k+1)} \leftarrow \boldsymbol{W}^{(k)} + \eta\left[\boldsymbol{A}^{(k)}\boldsymbol{W}^{(k)} - \boldsymbol{W}^{(k)}\boldsymbol{W}^{(k)\top}\boldsymbol{A}^{(k)}\boldsymbol{W}^{(k)}\right]$;   $k \leftarrow k+1$;
Until stopping condition is satisfied
Output $[\widehat{\boldsymbol{U}}; \widehat{\boldsymbol{V}}] \leftarrow \sqrt{2}\boldsymbol{W}^{(k)}$
```

Algorithm 1 is a globally convergent method which does not require any warm-up initialization or prior knowledge. The initial solution $\boldsymbol{W}^{(0)}$ is drawn uniformly from the set of all orthonormal matrices by applying a QR decomposition to a matrix with i.i.d. Gaussian entries. Algorithm 1 makes update online and uses $O(mr)$ space, while a batch method needs $O(m^2)$ space to store the explicit transition matrix.

2.3 Recovering The Network Partition from Random Walks

Recall that in Definition 1 the $m \times r$ matrix $\boldsymbol{M} = \boldsymbol{D}^{-1}\boldsymbol{V}$ gives a reduced-order representation for each vertex of the network. As long as we can estimate $\boldsymbol{D}, \boldsymbol{V}$, we would be able to partition the network by applying a clustering algorithm such as the k-means. Let us describe the overall procedure:

(1) Run Algorithm 1 on the Markov transition data and obtain $[\widehat{\boldsymbol{U}}; \widehat{\boldsymbol{V}}]$.

(2) Let $\widehat{\boldsymbol{\mu}}$ be the empirical estimate of the stationary distribution, i.e., $\widehat{\mu}_i = \sum_{k=1}^{n} \mathbb{I}(s^{(k)} = i)/n$. Let $\widehat{\boldsymbol{D}} = \mathrm{diag}(\widehat{\mu}_1, \widehat{\mu}_2, \ldots, \widehat{\mu}_m)$. Now each row of $\widehat{\boldsymbol{M}} = \widehat{\boldsymbol{D}}^{-1}\widehat{\boldsymbol{V}}$ gives an approximate r-dimensional representation for the corresponding state/vertex.

(3) Find a set of centers $C = \{c_1, c_2, \ldots, c_r\} \subset \mathbb{R}^r$ by solving the following problem:

$$\widehat{C} = \underset{C}{\mathrm{argmin}} \sum_{i=1}^{m} \min_{c\in C} d^2(\widehat{\boldsymbol{M}}_{i*}, c), \tag{5}$$

where $d(\widehat{\boldsymbol{M}}_{s_i*}, c_j) = ||\widehat{\boldsymbol{M}}_{s_i*} - c_j||_2$ is the Euclidean distance.

(4) Output the partition by assigning each state to its closest center.

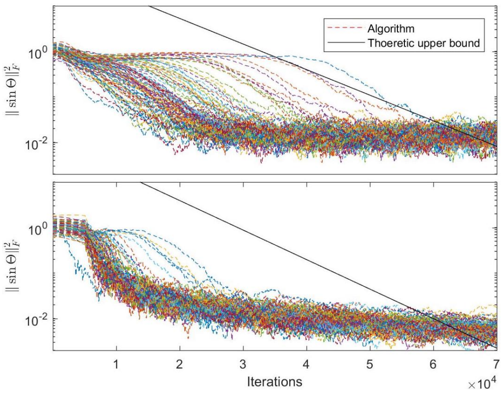
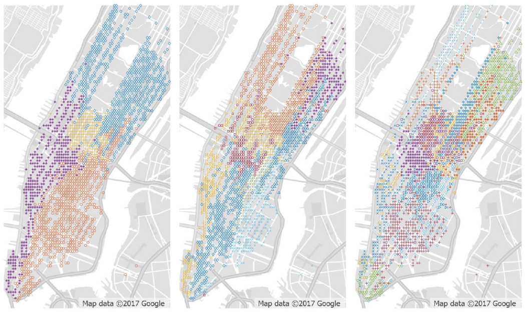


Figure 1: The convergence in subspace angle of 100 simulations: fixed stepsize (Top) and diminishing step size (Bottom)



(1) 4 States; (2) 10 States; (3) 15 States;

Figure 2: We apply our approach to partition the taxi trip data of Manhattan. We obtain informative structures that preserve the dynamics of the traffic network. Each color or symbol represents a meta-state.

References

[1] Z. Allen-Zhu and Y. Li. First efficient convergence for streaming k-pca: a global, gap-free, and near-optimal rate. *arXiv preprint arXiv:1607.07837*, 2016.

[2] R. Arora, P. Mianjy, and T. Marinov. Stochastic optimization for multiview representation learning using partial least squares. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1786–1794, 2016.

[3] A. R. Benson, D. F. Gleich, and L.-H. Lim. The spacey random walk: A stochastic process for higher-order data. *SIAM Review*, 59(2):321–345, 2017.

[4] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[5] J. Chen and B. Yuan. Detecting functional modules in the yeast protein–protein interaction network. *Bioinformatics*, 22(18):2283–2290, 2006.

[6] F. R. Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

[7] J. Harold, G. Kushner, and G. Yin. Stochastic approximation and recursive algorithm and applications. *Application of Mathematics*, 35, 1997.

[8] P. Jain, C. Jin, S. M. Kakade, P. Netrapalli, and A. Sidford. Streaming pca: Matching matrix bernstein and near-optimal finite sample guarantees for oja's algorithm. In *29th Annual Conference on Learning Theory*, pages 1147–1164, 2016.

[9] B. H. Junker and F. Schreiber. *Analysis of biological networks*, volume 2. John Wiley & Sons, 2011.

[10] C. J. Li, M. Wang, H. Liu, and T. Zhang. Near-optimal stochastic approximation for online principal component estimation. *arXiv preprint arXiv:1603.05305*, 2016.

[11] Y. Liu, C. Kang, S. Gao, Y. Xiao, and Y. Tian. Understanding intra-urban trip patterns from taxi trajectory data. *Journal of geographical systems*, 14(4):463–483, 2012.

[12] R. N. Mantegna. Hierarchical structure in financial markets. *The European Physical Journal B-Condensed Matter and Complex Systems*, 11(1):193–197, 1999.

[13] M. Meila and J. Shi. A random walks view of spectral segmentation. 2001.

[14] S. Nerur, R. Sikora, G. Mangalaraj, and V. Balijepally. Assessing the relative influence of journals in a citation network. *Communications of the ACM*, 48(11):71–74, 2005.

[15] M. E. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409, 2001.

[16] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[17] P. J. E. Peebles. *The large-scale structure of the universe*. Princeton university press, 1980.

[18] C. Pizzuti. Ga-net: A genetic algorithm for community detection in social networks. *Parallel problem solving from nature–PPSN X*, pages 1081–1090, 2008.

[19] T. D. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural networks*, 2 (6):459–473, 1989.

[20] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[21] E. Weinan, T. Li, and E. Vanden-Eijnden. Optimal partition and effective dynamics of complex networks. *Proceedings of the National Academy of Sciences*, 105(23):7907–7912, 2008.

[22] B. Xie, Y. Liang, and L. Song. Scale up nonlinear component analysis with doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pages 2341–2349, 2015.