DISTRIBUTED PRINCIPAL COMPONENT ANALYSIS ON NETWORKS VIA DIRECTED GRAPHICAL MODELS

Zhaoshi Meng **, Ami Wiesel* †*, and Alfred O. Hero III* *

* University of Michigan, Ann Arbor, MI 48109 USA
† Hebrew University of Jerusalem, Israel
* {mengzs,hero}@umich.edu, † amiw@cs.huji.ac.il

ABSTRACT

We introduce an efficient algorithm for performing distributed principal component analysis (PCA) on directed Gaussian graphical models. By exploiting structured sparsity in the Cholesky factor of the inverse covariance (concentration) matrix, our proposed DDPCA algorithm computes global principal subspace estimation through local computation and message passing. We show significant performance and computation/communication advantages of DDPCA for online principal subspace estimation and distributed anomaly detection in real-world computer networks.

Index Terms— Graphical models, principal component analysis, anomaly detection, distributed PCA, subspace tracking.

1. INTRODUCTION

We introduce a new approach to distributed principal component analysis called directed distributed PCA (DDPCA) that can be applied to networked data collection and analysis. As a widely used dimensionality reduction technique PCA estimates the principal subspace, which is the subspace spanned by the leading eigenvectors of the data covariance matrix. When applied to network data, distributed PCA results in an algorithm having improved scalability, decentralization of computation, and reduced communication cost. For recent contributions on distributed PCA, we refer the reader to related work [1–4].

Our proposed distributed PCA approach exploits a directed acyclic graphical model perspective of the network of measurements. Graphical models characterize conditional independence within the multivariate distribution of the data through the topology of a graph. The two most common families of graphical models are undirected graphical models and directed acyclic graph (DAG) models (also known as Bayesian networks). When the graph is sparse and the variables are jointly Gaussian, the graphical model imposes sparsity on the inverse covariance, variously called the information, concentration or precision matrix. Such a representation enables distributed and efficient inference algorithms. In [1], a distributed PCA algorithm called DPCA was introduced using a family of undirected graphical models, specifically decomposable Gaussian graphical models (DGGM).

The contribution of this paper is an extension of the DPCA framework to *directed* Gaussian graphical models. Instead of assuming sparsity in the concentration matrix, the proposed directed DPCA algorithm assumes sparsity in Cholesky factor of the concentration matrix, which results in closed-form distributed covariance estimation and reduced computation/communication complexity. DDPCA can equally be applied to non-directed decomposable graphical models by using a sparsity-preserving Markov-equivalent conversion.

DDPCA is a two-stage algorithm. In the first stage, DDPCA performs decoupled regressions to estimate the covariance matrix associated with the graphical model. In the second stage, a distributed orthogonal iterations method is implemented to accomplish global estimation of the principal subspace. In this way DDPCA requires only local computation and message passing between neighbors. The algorithm is also suitable for online incremental processing in subspace tracking problems.

The outline of the paper is as follows. In Section 2 we briefly review directed graphical models. In Section 3, a general network model is proposed, focusing on two specific network models that fit the DAG assumptions. Then in Section 4, we describe the DDPCA algorithm. In Section 5, DDPCA is applied to an online subspace estimation problem and distributed anomaly detection in a real-world computer network.

The following notation is used. Boldface upper case letters like $\mathbf{A}$ denote matrices, boldface lower case letters like $\mathbf{x}$ denote column vectors, and standard lower case letters like x denote scalars. The cardinality of a set $\mathcal{A}$ is denoted by $|\mathcal{A}|$. We use indices in the subscript $\mathbf{x}_\mathcal{A}$ or $\mathbf{X}_{\mathcal{A},\mathcal{B}}$ to denote sub-vectors or sub-matrices, respectively.

2. REVIEW OF DIRECTED GRAPHICAL MODELS

A directed graphical model $\mathcal{G}$ specifies conditional dependence among variable according to the existence of edges between nodes of a directed graph $\mathcal{G}$. If $i \rightarrow j$, then i is called a parent of j and j is called a child of i. The set of parents and children of i in $\mathcal{G}$ is denoted as $pa(i)$ and $ch(i)$, respectively. If there exists no directed cycle in $\mathcal{G}$, then it is called a *directed acyclic graph* (DAG).

A random vector $\mathbf{x}$ satisfies the Markov property with respect to a DAG $\mathcal{G}$ if the conditional independence between components of $\mathbf{x}$ are encoded in $\mathcal{G}$ through the notion of *d-separation* [5]. DAG models are most suitable for variables with natural ordering (such as temporal or spatial), where each variable only depends on a subset of its previous variables, which are denoted as its parent nodes in the DAG, i.e.

$$x_j = f(x_1, \dots, x_{j-1}) + \epsilon_j = f(\mathbf{x}_{pa(j)}) + \epsilon_j, \tag{1}$$

where ϵ_i's are uncorrelated residuals.

This research was supported in part by ARO grant W911NF-11-1-0391 and ISF 786/11.

A. Wiesel would like to thank M. Drton for introducing him to DAGs.

Another family of graphical models are the undirected graphical models, specified by a graph with undirected edges. We say that the set of nodes $\mathcal{C}$ *separates* sets $\mathcal{A}$ and $\mathcal{B}$, if there is no path between any node in $\mathcal{A}$ and node in $\mathcal{B}$ that does not include a node in $\mathcal{C}$. Unlike DAG models, the Markov property on undirected graphs states that random variables measured in two distinct subsets are conditionally independent given the separating subset. Under Gaussian assumptions the Markov property in undirected models imposes zeros in the concentration matrix [5].

Decomposable Gaussian Graphical Models (DGGM) define an important family of undirected Gaussian graphical models. The family can be defined by a junction tree of cliques $C_1, \ldots, C_K$ which satisfy a *perfect elimination order*. These ordered cliques are coupled through separators

$$S_j = (C_1 \cup C_2 \cup \cdots \cup C_{j-1}) \cap C_j \tag{2}$$

for $j = 2, \ldots, , K$, and satisfy the *running intersection property*: for all $j \geq 2$ there is a $k < j$ such that $S_j \subseteq C_k$.

DGGMs are closely related to Gaussian DAG models. Given a DGGM we can construct a vector-based DAG model by directing the edges in the junction tree. The resulting model preserves all the conditional independencies and the sparsity pattern in its information domain, and is called a *Markov equivalence*. The conversion from DAG models to DGGMs is also straightforward through moralization. However, this may require additional edges which reduce the sparsity level. In this sense the DAG model is a more efficient representation than its DGGM counterpart.

3. A DAG NETWORK MODEL

In this section we motivate the DAG models for network modeling and describe two specific models whose spatial covariance satisfies the DAG structure. Consider a computer network consisting of N nodes and L links (adjacent nodes are connected by two links). The network carries traffic flows from origin nodes to destination nodes, known as OD (Origin-Destination) flows, through routing over a predetermined ordered subset of links (called a *path*) of the network. On each link of the network, the accumulation of all OD flows that pass through this link is measured. Therefore the link flows and the OD flows can be related by a linear equation

$$\mathbf{y} = \mathbf{A}\mathbf{x}, \tag{3}$$

where $\mathbf{y} \in \mathbb{R}^L$ is the link-level flow measurement vector, and $\mathbf{x} \in \mathbb{R}^P$ is the OD flow vector, and P is the total number of OD paths. The *routing matrix* $\mathbf{A} = (a_{lp})_{L \times P}$ is defined as:

$$a_{lp} = \begin{cases} 1, & \text{if link } l \text{ is on path } p \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

As noted in the literature, under uncongested network conditions the OD traffic flows x_p's can be well approximated as statistically uncorrelated. Thus the statistical correlation between components of $\mathbf{y}$ are solely determined by the structure of routing matrix $\mathbf{A}$. We consider the following two models for link flow vector $\mathbf{y}$:

Scenario A (Decomposable model). In this scenario, two distantly separated link flow variables are modeled as conditionally independent given the separator flow variables between them. Therefore if the network topology satisfies decomposable separation as defined in (2), then $\mathbf{y}$ is readily modeled by a DGGM, which also can be converted into a DAG model as described in Section 2. This model is equivalent to the model [1] that specifies a decomposable sparsity structure on the precision matrix.

Scenario B (Single-source model). In this scenario, the network flow consists of OD flows originating from a single source node (See Fig. 3(a)). Due to the tree-structured routing in this network, each "parent" link carries the accumulated flows of all its descendants. Thus its corresponding flow variable depends only on its descendant flows variables. Therefore there exists a natural ordering of all the link flow variables, as in (1), and $\mathbf{y}$ naturally follows a DAG model. This single-source model arises in the context of flow-based network anomaly detection, such as spoofing detection, where an attacker makes independent and unauthorized connections and injects phony packets into the OD flow tree.

4. DISTRIBUTED PCA ON GRAPHICAL MODELS

4.1. Problem Formulation

Let $\mathcal{G}$ be a DAG with known edge topology, and let $\mathbf{x}$ be a p-dimensional zero mean Gaussian random vector that satisfies the Markov properties specified by the DAG $\mathcal{G}$. Measured is a set of T independent and identically distributed realizations of $\mathbf{x}$, denoted as $\{\mathbf{x}[t]\}_{t=1}^T$. We assume that there are p units, or agents, that collect and process the data in a distributed manner. Each agent i only collects and processes all the T samples of the component $\{x_i[t]\}_{t=1}^T$. Agents can communicate with their neighbors (called local communication) defined by DAG $\mathcal{G}$. Our goal is to perform global estimation of the principal subspace spanned by the first r leading eigenvectors of the covariance matrix. In other words, our algorithm searches for the the linear combination $X = \mathbf{V}^T\mathbf{x}$ having maximal variance, where $\mathbf{V} \in \mathbb{R}^{p \times r}$.

4.2. Distributed Covariance Estimation

In the first stage of DDPCA we perform distributed covariance estimation. It is well-known [6] that, given a random vector $\mathbf{x}$ which satisfies a known DAG model, its covariance matrix $\mathbf{\Sigma}$ can be transformed to a diagonal matrix $\mathbf{\Omega}$ as following:

$$(\mathbf{I} - \mathbf{\Lambda})\mathbf{\Sigma}(\mathbf{I} - \mathbf{\Lambda})^T = \mathbf{\Omega}, \tag{5}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{\Lambda}$ is a lower-triangular matrix with

$$\mathbf{\Lambda}_{jk} = \begin{cases} 0 & \text{if } k \notin pa(j) \\ \lambda_{jk} & \text{if } k \in pa(j). \end{cases} \tag{6}$$

This formulation leads to the following recursive linear system:

$$x_j = \sum_{k \in pa(j)} \lambda_{jk} x_k + \epsilon_j, \quad j = 1, ..., p, \tag{7}$$

where the variances of the uncorrelated residuals ϵ_j's form the diagonal matrix $\mathbf{\Omega}$.

The covariance matrix $\mathbf{\Sigma}$ is fully characterized by $\mathbf{\Lambda}$ and $\mathbf{\Omega}$. Thus, a simple approach to covariance estimation is to perform decoupled linear minimum mean squared error (LMMSE) regression in model (7) for each x_j to estimate the parameters. In fact, this procedure results in the maximum likelihood estimate [5]. For real-time applications, this distributed covariance estimation scheme can be recursively implemented, e.g., by using recursive least squares (RLS) to update the estimate. This enables fast distributed principal subspace tracking.

For the purpose of presentation, we develop the estimator for the case of scalar node variables, e.g. x_j. In the general vector case, (6) and (7) are modified by replacing λ_{jk}'s by matrices.

4.3. Distributed Orthogonal Iterations

The second step of DDPCA is a decentralized implementation of Orthogonal Iteration (OI) method, also known as Power Iteration, to estimate the leading eigenvectors of covariance matrix using the estimated Cholesky factor obtained in the first step. OI is one of the simplest yet prevailing iterative algorithms for estimating the principal subspace of large-scale matrices. In each iteration, the matrix is multiplied by the current estimate of the eigenvectors, then the resulting vectors are orthonormalized. Under broad assumptions the subspace spanned by the produced vectors converges to the true principal subspace of the matrix at a linear rate.

The most expensive operation in the implementation of OI is a matrix-vector multiplication. DDPCA simplifies this calculation by exploiting the structural sparsity of the Cholesky factor of the covariance matrix. Let $\mathbf{L}$ denote the *modified* Cholesky factor (also called the Backward Cholesky factor) of the concentration matrix, defined as:

$$\mathbf{\Sigma}^{-1} = \mathbf{L}^T\mathbf{L}. \tag{8}$$

From (5), we have

$$\mathbf{L} = \mathbf{\Omega}^{-1/2}(\mathbf{I} - \mathbf{\Lambda}). \tag{9}$$

It is easy to see that $\mathbf{L}$ shares the same lower-diagonal nonzero pattern as $\mathbf{\Lambda}$, which characterizes the topology of the DAG $\mathcal{G}$ through (6). Therefore the matrix-vector multiplication between the covariance matrix and a vector $\mathbf{q}$ can be structured as:

$$\mathbf{z} = \mathbf{\Sigma}\mathbf{q} = \mathbf{L}^{-1}\mathbf{L}^{-T}\mathbf{q}, \tag{10}$$

which is efficiently performed by introducing an auxiliary vector $\mathbf{y}$ and sequentially solving the following two linear triangular systems:

$$\mathbf{L}^T\mathbf{y} = \mathbf{q},\ \mathbf{L}\mathbf{z} = \mathbf{y}, \tag{11}$$

through backward substitution and forward substitution, respectively. Since the non-zero pattern of $\mathbf{L}$ matches the edge topology of $\mathcal{G}$, solving a given component of the solution vector only requires *linear* message-passing from its parents or children. For example, the j-th component of $\mathbf{y}$ is calculated by

$$y_j = \mathbf{L}_{jj}^{-1}(v_j - \sum_{m\in\mathrm{ch}(j)} \mathbf{M}_{m\to j}), \tag{12}$$

where the message is $\mathbf{M}_{m\to j} = \mathbf{L}_{mj}^T y_m$. Note that solving this local triangular system (12) requires a quadratic computation cost and a linear communication cost in the dimension of y_j.

For estimating the principal subspace, multiple principal components are estimated simultaneously. OI also requires orthonormalization in each iteration which can be efficiently implemented in distributed form [7] with only small additional cost.

4.4. Comparison with PCA and DPCA [1]

DDPCA, DPCA and standard PCA are matched to different models. Standard PCA makes no assumptions on covariance structure. DPCA assumes that the covariance follows a decomposable undirected graphical model with known sparsity pattern in precision matrix. DDPCA assumes that the Cholesky factor of the ~~covariance~~ follows a DAG model with known sparsity structure.

Moreover, in the case that Markov equivalent DAG model and DGGM are assumed for applying DDPCA and DPCA, respectively, DDPCA enjoys lower implementation complexities. DPCA performs a sequence of local PCA operations on the junction tree, which have cubic local computation complexity and quadratic inter-clique communication cost as a function of separator dimension. On the other hand, DDPCA exploits sparsity via distributed regression and substitution techniques, which requires quadratic computation and linear communication cost in the local dimension. Therefore, DPCA is suitable for problems with very small separators and relatively large cliques, whereas DDPCA is applicable to arbitrary DAGs.

5. EXPERIMENTS

5.1. Online subspace estimation example

We first illustrate DDPCA on a synthetic example. The objective is online estimation of the first principal component from an i.i.d. sequence of samples. The samples are generated from a 80-dimensional multivariate normal random vector using a four-node vector-based DAG model with star topology (three disjoint nodes point to the fourth node). Each node corresponds to a 20-dimensional random sub-vector. The Markov equivalent undirected graph of this sparse DAG via moralization is a fully-connected graph, which means that the precision matrix has no sparsity. Therefore DPCA cannot be used for distributing the computation. However, the proposed DDPCA can distribute the computation by exploiting sparsity in the Cholesky factor.

We perform DDPCA for an increasing window of successive samples, updating the previous estimate of Cholesky factor at each sample time. Using this evolving factor, we perform a few number of OI's at each sample time to update the previous eigenvector estimate. For comparison, we implement the standard PCA using OI with full sample covariance that does not account for the DAG model.

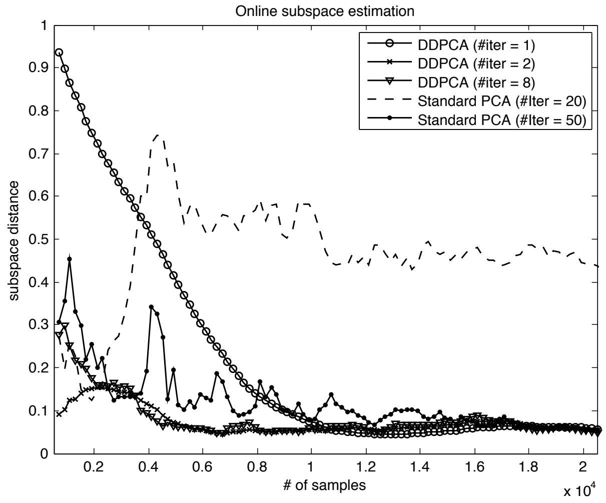
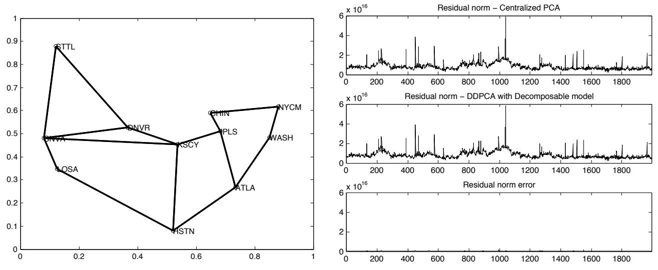


Fig. 1. Online estimation example

Fig. 1 shows the subspace estimation error, i.e., the distance between the estimated and the true subspace with respect to the total number of samples. It can be seen that for DDPCA even two iterations of OI's in each update result in a fast convergence to the steady state regime where the Cholesky factor estimate is stable; whereas standard PCA performs poorly even using 20 OI's for the updating, and requires 50 OI's to achieve similar performance to DDPCA. The better performance and faster convergence of DDPCA is due to the matched DAG model, which is a more parsimonious and lower dimensional covariance model. The decentralized framework of DDPCA also makes it computationally advantageous.

5.2. Distributed Anomaly Detection in Abilene Network



(a) Connectivity of Abilene network (b) Residual norm and error (decomposable model)

Fig. 2. Anomaly detection in Abilene with decomposable model

Finally, we apply the proposed DDPCA to anomaly detection in a real-world Abilene network [8] based on a DAG network model. Abilene is the Internet2 backbone which carries traffic between universities in the United States. Fig. 2(a) shows its connectivity map consisting of 11 nodes and 30 links. Measurements of link flow traffic data satisfy the network model (3), where the routing matrix $\mathbf{A}$ is known. Our goal is to detect the anomalies occurring in the OD flow vector $\mathbf{x}$ with observations of $\mathbf{y}$.

In this context, PCA is used to estimate a low dimensional principal subspace containing the nominal flow traffic. The test data are projected into the nullspace and the norm of the projected data is thresholded to indicate potential anomalies [8]. DDPCA enables distributed estimation of the principal subspace through decentralized computation and communication over the network.

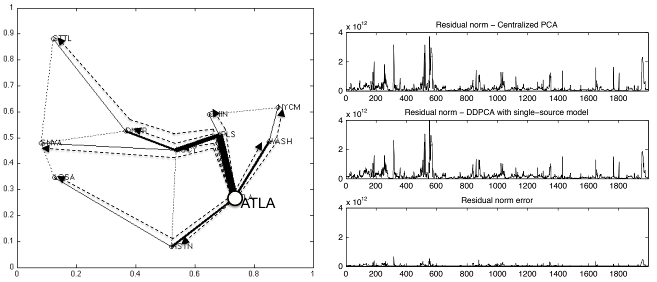
We consider the two models presented in Section 3 for implementing the PCA algorithms on this network data:

(A) Decomposable model. We consider all the OD flows and links. In view of Fig. 2(a) there exist two separators: $S_1 = \{$DNVR-KSCY, SNVA-KSCY, LOSA-HSTN$\}$, $S_2 = \{$KSCY-IPLS, HSTN-ATLA$\}$ which physically separate the other three subsets of links. Then we can model the link traffic variable by a three-clique DGGM, which can be equivalently converted into a block-sparse DAG model.

Note that in this model, the dimensions of the separating sets $(|S_1| = 6, |S_2| = 4)$ are not significantly smaller than the clique dimensions $(14, 12,$ and 14, respectively). Therefore the computation cost ($\sim 14^2$) and communication cost (~ 14) of DDPCA is less expensive than DGGM-based DPCA, which requires cubic local computation cost ($\sim 14^3$) and quadratic communication cost ($\sim 6^2$).

(B) Single-source model. In this scenario, we consider all the OD flows originating from node ATLA, as shown in Fig. 3(a). The dotted lines with arrows indicate OD flows, and the thickness of solid edges are proportional to the number of OD flows passing through it. As described above we can construct a 11-node sparse DAG model for the link traffic. Note that in this example, the equivalent DGGM has two additional edges and thus is less sparse than the DAG model.

In our experiments, we examined two weeks of real-world flows data. We learn the nominal principal subspace from the first week's data using standard centralized PCA and DDPCA, respectively. Then we project the second week's link traffic, the test data, on the nominal principal subspace and examine the residual. Fig. 2 and Fig. 3 show the norm of the residual signal using centralized PCA (first row), using DDPCA (second row) and showing their difference (third row). We can see that DDPCA successfully approximates anomaly detection performance of the centralized PCA. Anomaly detection can be performed by thresholding the residual norm and locating the peaks.



(a) OD tree from ATLA (b) Residual norm and error (single-source model)

Fig. 3. Anomaly detection in Abilene with single-source model

6. CONCLUSION

We have presented a distributed PCA algorithm on directed Gaussian graphical models, called DDPCA, designed for networked data whose covariance has sparse Cholesky factor with known zero pattern. We have illustrated DDPCA's performance for estimating subspace and detecting anomalies in link traffic in Abilene network data.

7. REFERENCES

[1] A. Wiesel and A.O. Hero III, "Decomposable principal component analysis," *Signal Processing, IEEE Transactions on*, vol. 57, no. 11, pp. 4369–4377, 2009.

[2] M. Gastpar, P.L. Dragotti, and M. Vetterli, "The distributed karhunen–loève transform," *Information Theory, IEEE Transactions on*, vol. 52, no. 12, pp. 5177–5196, 2006.

[3] L. Huang, X.L. Nguyen, M. Garofalakis, M.I. Jordan, A. Joseph, and N. Taft, "In-network PCA and anomaly detection," *Advances in Neural Information Processing Systems*, vol. 19, pp. 617, 2007.

[4] L. Li, A. Scaglione, and J.H. Manton, "Distributed principal subspace estimation in wireless sensor networks," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 5, no. 4, pp. 725 –738, aug. 2011.

[5] S.L. Lauritzen, *Graphical models*, vol. 17, Oxford University Press, USA, 1996.

[6] P. Rütimann and P. Bühlmann, "High dimensional sparse covariance estimation via directed acyclic graphs," *Electronic Journal of Statistics*, vol. 3, pp. 1133–1160, 2009.

[7] D.P. O'Leary and P. Whitman, "Parallel QR factorization by householder and modified gram-schmidt algorithms," *Parallel computing*, vol. 16, no. 1, pp. 99–112, 1990.

[8] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *Proceedings of ACM SIGCOMM 2004*, Aug. 2004, pp. 219–230.