Consensus-Based Distributed Principal Component Analysis in Wireless Sensor Networks

Sergio Valcarcel Macua, Pavle Belanovic, Santiago Zazo

ETS Ingenieros de Telecomunicación
Universidad Politécnica de Madrid
Madrid, Spain

ABSTRACT

Principal component analysis is a powerful technique for data analysis and compression, with a wide range of potential applications in wireless sensor networks. However, its centralized implementation, with a fusion center collecting all the samples, is inefficient in terms of energy consumption, scalability, and fault tolerance. Previous distributed approaches reduce the communication cost, but not the lack of flexibility, as they require multi-hop communications if the network is not fully connected. We present two fully distributed consensus-based algorithms that are guaranteed to converge to the global results, using only local communications among neighbors, regardless of the data distribution or the sparsity of the network: CB-DPCA is based on finding the eigenvectors of local covariance matrices, while CB-EM-DPCA is a distributed version of the expectation maximization algorithm. Both offer a flexible trade-off between the tightness of the achieved approximation and the associated communication cost.

1. INTRODUCTION

1.1. Motivation

Wireless sensor networks (WSNs) are typically made up of small, inexpensive, and unreliable nodes, each equipped with a set of sensors and a wireless communication interface. Due to the independent power supply and small size of the nodes, coupled with the need for long useful lifetimes between recharges, energy efficiency is a key requirement for any WSN. In particular, the communication tasks performed by the nodes carry a heavy energy cost and must be avoided whenever possible.

Each application-layer algorithm running in a WSN can be either centralized, in that it is performed at a fusion center, or distributed, such that the nodes cooperatively arrive at the same result without the need for a fusion center. Distributed approaches are typically significantly more energy efficient, because they avoid costly multi-hop transmissions. Also, they make the WSN more scalable and fault-tolerant, since a fusion center is a potential single point of failure for the network.

Principal Component Analysis (PCA) is a dimensionality reduction technique widely used as a feature extraction step for visualization, data compression, clustering and classification, and it has a wide range of potential applications in WSNs.

1.2. Related work

PCA was first introduced by Pearson in 1901, as a methodology for linear regression, and later, in 1933, independently derived and given its current name by Hotelling, as a technique for the analysis of correlation between many random variables. For a detailed review on PCA see [1] and references therein. The most common implementation of PCA is based on getting the eigenvectors of the sample covariance matrix, and we will show an algorithm (CB-DPCA) to do so in a distributed fashion.

In this paper we will also focus on a probabilistic latent variable model-based method [2, 3, 4], which uses an expectation maximization (EM) algorithm to estimate the latent principal subspace with maximum likelihood of causing the observed data. EM for PCA is specially suitable for iterative processing of large data sets using limited computational resources. We will also provide a distributed EM algorithm (CB-EM-DPCA) to approximate PCA.

Consensus algorithms were originally proposed in the 1970’s [5] as a method for reaching agreement on the probability distribution of an unknown shared parameter by a group of peers with subjective probability distributions for this unknown value. Out of this theory grew the now well known average consensus algorithms for undirected [6, 7] and directed network graphs [8], which have direct implications to the WSN setting. As their name suggests, these algorithms assure that each node asymptotically converges to the arithmetic mean of the parameter values of all the nodes in the network. Average consensus algorithms require only local communications between neighbors, making them efficient and scalable. As such, they present a powerful tool for distributing centralized methods, such as PCA.

The use of a consensus-based distributed EM algorithm was introduced by [9] for density estimation and classification. However, it relies on bridge sensors which should be found during the startup phase, therefore reducing the scalability of the algorithm. Our CB-EM-DPCA algorithm overcomes this limitation by dealing with all nodes homogeneously.

Partially distributed implementations of PCA have been previously proposed. They focus on saving part of the multi-hop communication cost by either local computations [10] or aggregation services [11, 12, 13], but they still require a fusion center to merge the local results. In the context of distributed compression and source coding, [14] proposed a distributed Karhunen-Loève transform which is posed as an optimization problem, where convergence to the global optimum is, in general, not assured. The two consensus-based dis-
ticular, these distributed PCA algorithms presented in this paper outperform all the others because both guarantee convergence, with no fusion center, just by single-hop communications.

In addition, [15] presented a distributed distance matrix estimation algorithm for manifold learning and dimensionality reduction. It builds on a distributed implementation of the power method [16], which is also based on a standard average consensus algorithm. Again, our algorithms have clear advantages over this one: first, the distributed power method does not perform well in sparse networks, since it requires a post-consensus flooding of intermediate results; moreover, it is not scalable, as each node is required to know the size of the network (i.e. the total number of samples); for the same reason, our formulation is more general, since each node can have any number of samples, only known by itself; finally, although the communication cost is approximately the same as our CB-EM-DPCA algorithm, the latter performs the communication in larger blocks (since all the eigenvectors are calculated in parallel, rather than sequentially), thus saving transmission overheads and time. On the other hand, the distributed power method was showcased on a non-linear technique, whereas we demonstrate the use of our algorithms on simple linear PCA. Nevertheless an extension to a non-linear example is straightforward.

1.3. Contributions

Two distributed algorithms for computing PCA that asymptotically match the performance of the centralized algorithm are presented. The first one, named consensus-based distributed PCA (CB-DPCA), is based on finding the dominant eigenvectors of the sample covariance matrix, which is approximated, in a distributed fashion and arbitrarily close, using consensus. The second one, named consensus-based EM distributed PCA (CB-EM-DPCA) is a distributed version of the EM method for iteratively approximating the PC subspace.

These algorithms are the first solutions for fully distributed PCA computation, where no fusion center is required, using only single-hop communications between neighbors regardless of the sparsity of the network, and where convergence to the global solution is guaranteed for any data distribution. A trade off between communication cost and accuracy can be tuned by the number of consensus rounds. By way of simulations we illustrate that in both cases distributed PCA can be approximated very closely in few consensus iterations even in relatively sparse large-scale networks. This brings about a significant energy saving compared to the centralized solution, and a large performance improvement compared to the case with no cooperation among the nodes.

The key to our algorithms is a consensus matrix product operation, a simple but powerful idea that we use for computing the sample covariance matrix as well as least squares in a distributed fashion, and that can be easily applied for distributing many other algorithms.

1.4. Outline

A description of our system model is presented in Section 2. The consensus matrix-product operator, and its use for the distributed approximation of the data covariance matrix, as well as the resulting CB-DPCA are introduced in Section 3. The standard centralized version of the EM algorithm for calculating the PC, together its distributed implementation using consensus least squares (CB-EM-DPCA), are shown in section 4. Some simulation results illustrating the performance of both solutions are presented in Section 5. Finally, some concluding remarks are made in Section 6.

2. SYSTEM MODEL

Let us assume that a WSN gathers data samples by sensing its environment and that PCA is to be performed on the global data set for the purpose of, say, data compression or event detection.

Let the $N$ nodes that compose the network gather $t_n$ samples each, such that the number of samples gathered may differ from node to node. The number of samples in the global data set is thus $T := \sum_{n=1}^{N} t_n$.

Each node is equipped with $D$ sensors, each of which is capable of producing real-valued measurements of the environment, such as its temperature, humidity, light intensity, or similar. Hence, the $i^{th}$ data sample measured at node $n$ is the column vector $\mathbf{y}_{n,i} \in \mathbb{R}^D$.

The set of data samples gathered at node $n$ is then $\mathbf{Y}_n := \{\mathbf{y}_{n,1}, \mathbf{y}_{n,2}, \ldots, \mathbf{y}_{n,t_n}\}$ so that $\mathbf{Y}_n \in \mathbb{R}^{D \times t_n}$. Similarly, the global data set is defined as $\mathbf{Y} := \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_N\}$ so that $\mathbf{Y} \in \mathbb{R}^{D \times T}$.

This structure is illustrated in Figure 1.

![Fig. 1. Distributed structure of the global dataset](image)

We also define the centered data set

$$\mathbf{Y}^* := \mathbf{Y} - \mathbf{\mu} 1_T,$$  \hspace{1cm} (1)

where the column vector $\mathbf{\mu} \in \mathbb{R}^{D \times 1}$ is the sample mean of each sensor of the global data set, and $1_T$ is a row vector of $T$ ones.

When the processing is done centrally, i.e. when the global data set is available in the fusion center, $\mathbf{\mu}$ (and hence $\mathbf{Y}^*$) can be calculated easily. In the distributed case however, no fusion center exists and a distributed method of calculating the global sample mean $\mathbf{\mu}$ has to be employed by the WSN, which is immediate from standard average consensus [8]. Once this is done, each node can in isolation derive the local centered data set $\mathbf{Y}_n^* = \mathbf{Y}_n - \mathbf{\mu} 1_{t_n}$. Obviously, $\mathbf{Y}^*$ has the same block-wise structure of $\mathbf{Y}$, as illustrated in Figure 1.

For the sake of simplicity, in the explanations that follow, we will assume a zero centered data set $\mathbf{Y} = \mathbf{Y}^*$. The entire discussion can easily be extended to a non-zero-mean data set $\mathbf{Y}$ simply by substituting for $\mathbf{Y}^*$ throughout using (1).

3. CONSENSUS BASED DISTRIBUTED PCA (CB-DPCA)

3.1. PCA from the data covariance matrix

The standard centralized PCA algorithm requires that all the data samples from all the nodes be gathered at the fusion center, assuming there is one in the WSN. With this access to the full data set $\mathbf{Y}$, the fusion center can perform PCA directly by deriving the data covariance matrix and hence its $P$ dominant eigenvectors, as summarized in Algorithm 1.

In a similar but distributed approach, CB-DPCA (Algorithm 2) shows how every node can approximate the global data covariance matrix, and then derive its PC from this.
3.2. Consensus matrix-product operator

For a given network with $N$ nodes, where each node has two local sets of samples $A_n := \{a_{n,1}, \ldots, a_{n,t_n}\} \in \mathbb{R}^{P \times t_n}$ and $B_n := \{b_{n,1}, \ldots, b_{n,t_n}\} \in \mathbb{R}^{D \times t_n}$, let us define global data matrices $A := [A_1, \ldots, A_N]$ and $B := [B_1, \ldots, B_N]$, similar to the structure shown in Figure 1.

Since $AB^T = \sum_{n=1}^{N} A_n B_n^T = \sum_{n=1}^{N} \sum_{i=1}^{t_n} a_{n,i} b_{n,i}^T$, we note that a simple average consensus algorithm can directly produce the same global product as a fusion center, but scaled by $\frac{1}{N}$. Based on standard average consensus algorithms [8], we define a consensus matrix-product operator, $(A_n B_n^T)_c$, to run in every node, as

$$ (A_n B_n^T)_c := \frac{1}{N} \sum_{n=1}^{N} A_n B_n^T $$

By combining (2), (3) and (4), every node has a distributed way to approximate (arbitrarily close) the scaled global data covariance matrix (3) that a fusion center would see

$$ S_c := (Y_n Y_n^T)_c = \frac{1}{N} \sum_{n=1}^{N} Y_n Y_n^T = \frac{1}{N} \sum_{n=1}^{N} S_n = \frac{1}{N} S $$

3.3. Distributed data covariance matrix approximation

For the system model described in section 2, let $S$ be the global data covariance matrix defined by

$$ S := YY^T = \sum_{n=1}^{N} Y_n Y_n^T $$

By combining (2), (3) and (4), every node has a distributed way to approximate (arbitrarily close) the scaled global data covariance matrix (3) that a fusion center would see

$$ S_c := (Y_n Y_n^T)_c = \frac{1}{N} \sum_{n=1}^{N} Y_n Y_n^T = \frac{1}{N} \sum_{n=1}^{N} S_n = \frac{1}{N} S $$

Since the averaging factor $\frac{1}{N}$ does not affect the eigenvectors of $S$ (though it does its eigenvalues), after consensus each node can compute the PCA as the subspace spanned by the dominant eigenvectors of $S_c$, which will be (arbitrarily close) to those of $S$. This is the base of CB-DPCA, as shown in Algorithm 2.

Algorithm 2 CB-DPCA

1. For each node $n$, let $Y_n$ be the local observed data set of dimensionality $P$.
2. Compute $S_n := Y_n Y_n^T$.
3. Consensus rounds:
   - Compute $S := \frac{1}{N} \sum_{n=1}^{N} S_n$.
   - Let $A := [A_1, \ldots, A_N]$ and $B := [B_1, \ldots, B_N]$ be the consensus matrices, scaled by $\frac{1}{N}$.
4. For each node $n$, let $Y_n$ be the local observed data set.
5. Compute $S_n := Y_n Y_n^T$.
6. Compute $S := \frac{1}{N} \sum_{n=1}^{N} S_n$.
7. Let $A := [A_1, \ldots, A_N]$ and $B := [B_1, \ldots, B_N]$ be the consensus matrices, scaled by $\frac{1}{N}$.
8. Compute $S_c := (Y_n Y_n^T)_c$.
9. Compute $S := \frac{1}{N} \sum_{n=1}^{N} S_n$.
10. Let $A := [A_1, \ldots, A_N]$ and $B := [B_1, \ldots, B_N]$ be the consensus matrices, scaled by $\frac{1}{N}$.

Thus, in the $k^{th}$ iteration, the E step is given by

$$ X^{(k)} = (C^{(k-1)T} C^{(k-1)})^{-1} C^{(k-1)T} Y $$

4. CONSENSUS BASED DISTRIBUTED EM-PCA (CB-EM-PCA)

4.1. PCA in the context of continuous latent models

We can think of the observed data as the projection of a lower dimensional underlying latent data. Specifically, we let

$$ Y = CX + \varepsilon $$

where $X \sim N(0, I) \in \mathbb{R}^{P \times T}$ is the latent data set and $Y \in \mathbb{R}^{D \times T}$ is the observed data, with $\varepsilon \sim N(0, \sigma^2 I)$. Under this assumption, the maximum likelihood estimate for $C$ is the set of $P$ dominant eigenvectors of the sample covariance matrix of the global data set $Y$ (see [4], Appendix A.2).

Taking the limit $\varepsilon = \lim_{\sigma \to 0}$ then it results the standard PCA model [2]

$$ Y = CX $$

so that given the global projection matrix $C$, each node can in isolation derive its part of the global projected data set.

In the context of WSN, when nodes deal with huge data sets or low computational resources an iterative algorithm to compute $C$ is desirable. Based on the probabilistic latent variable model described above (7), it is possible to find an iterative EM like algorithm, which will converge to a global maximum independent on the distribution of the observed data set [2].

The EM algorithm is a two steps iterative technique for computing maximum likelihood solutions. In the E-step, it computes sufficient statistics, while in the M-step, the parameters of the model are estimated. For a detailed review on EM see [17].

The centralized EM algorithm for PCA, running in a fusion center, is shown in Algorithm 3.

Algorithm 3 EM-PCA

1. For each node $n$, let $Y_n$ be the local observed data set.
2. EM loop ($k^{th}$ iteration)
   - **E step**
     - $\beta^{(k)} := (C^{(k-1)T} C^{(k-1)})^{-1} C^{(k-1)T} Y$
     - $X^{(k)} := \beta^{(k)} Y$
   - **M step**
     - $\delta^{(k)} := YX^{(k)T}$
     - $\gamma^{(k)} := X^{(k)}X^{(k)T}$
     - $C^{(k)} := \delta^{(k)} \gamma^{(k)^{-1}}$

Thus, in the $k^{th}$ iteration, the E step is given by

$$ X^{(k)} = (C^{(k-1)T} C^{(k-1)})^{-1} C^{(k-1)T} Y $$
while the M step is given by
\[ C^{(k)} = YX^{(k)T} (X^{(k)}X^{(k)T})^{-1}. \]

It is worth noting that both the columns of \( C \) and the dominant \( P \) eigenvectors of the sample covariance matrix span the same principal subspace. However, they are not equal. In fact the columns of \( C \) are not orthogonal. In the case orthogonality is required [18] showed that adding lower/upper triangularization operators to the E/M steps, the EM loop will output directly an orthogonal basis of the principal subspace.

**4.2. Consensus Least Squares**

Given a linear system of equations \( QR = P \), the least squares solution of \( Q \) is given by \( Q = PR^T (RR^T)^{-1}. \) We can immediately split the pseudo-inverse into two factors
\[ \delta = PR^T \]
\[ \gamma = RR^T. \]

By using the consensus matrix-product operator defined in (2) we can easily approximate, in a distributed fashion, the averaged factors
\[ (P_nR_n^T)_c = \frac{1}{N} \sum_{n=1}^{N} \delta_n = \frac{1}{N} \delta = \delta_* \]
\[ (R_nR_n^T)_c = \frac{1}{N} \sum_{n=1}^{N} \gamma_n = \frac{1}{N} \gamma = \gamma_* \]

Moreover, since (14) has to be inverted, the scaling factors cancel out, and hence the required least squares solution of \( Q \) is immediately achieved in a distributed way
\[ \hat{Q}_c = (P_cR_c^T) \left( (R_cR_c^T)_c \right)^{-1} = \delta_*^{-1} = Q \]

**4.3. Distributed EM**

We first note that, assuming the same \( C^{(k)} \) is available at all the nodes at the start of every new iteration, each node can carry out the E step in isolation and derive its own portion of the projections \( X_n. \)

In order to distribute the EM-PCA (Algorithm 3), we can follow the same approach as in consensus least squares. The local variables \( \delta^{(k)}_n \) and \( \gamma^{(k)}_n \) are inputs into an embedded consensus algorithm, whose results are \( \delta^{(k)}_* \) and \( \gamma^{(k)}_* \). These outputs asymptotically approach the same values in all the nodes in the network.

Consequently, all the nodes approach the same value for \( C^{(k)} \), since it is based solely on the outputs of the embedded consensus algorithm. It turns out that this value of \( C^{(k)} \) is the same as that calculated by Algorithm 3, since
\[ C^{(k)} = \delta^{(k)}_* (\gamma^{(k)}_*)^{-1} = (Y_nX_n)_c \left( (X_nX_n^T)_c \right)^{-1} \]
\[ = \frac{1}{N} \sum_{n=1}^{N} Y_nX_n^{(k)T} \left( \frac{1}{N} \sum_{n=1}^{N} X_n^{(k)}X_n^{(k)T} \right)^{-1} \]
\[ = YX^{(k)T} (X^{(k)}X^{(k)T})^{-1} \] (15)
due to the column-block-wise structure of \( Y \) and \( X \) illustrated in Figure 1. And this is the base of the proposed CB-EM-DPCA, as it is shown in Algorithm 4.

<table>
<thead>
<tr>
<th>Algorithm 4 CB-EM-DPCA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EM loop</strong> (( k^{th} ) iteration)</td>
</tr>
<tr>
<td><strong>E step</strong></td>
</tr>
<tr>
<td>( \beta^{(k)} \leftarrow (C^{(k-1)T} C^{(k-1)})^{-1} C^{(k-1)T} )</td>
</tr>
<tr>
<td>( X_n^{(k)} \leftarrow \beta^{(k)} Y_n )</td>
</tr>
<tr>
<td><strong>M step</strong></td>
</tr>
<tr>
<td>( \delta^{(k)}_n \leftarrow Y_nX_n^{(k)T} )</td>
</tr>
<tr>
<td>( \gamma^{(k)}_n \leftarrow X_n^{(k)T} X_n^{(k)T} )</td>
</tr>
<tr>
<td><strong>Consensus rounds</strong></td>
</tr>
<tr>
<td>( \delta^{(k)}_c \leftarrow (Y_nX_n)<em>c = \frac{1}{N} \sum</em>{n=1}^{N} \delta^{(k)}_n )</td>
</tr>
<tr>
<td>( \gamma^{(k)}_c \leftarrow (X_nX_n^T)<em>c = \frac{1}{N} \sum</em>{n=1}^{N} \gamma^{(k)}_n )</td>
</tr>
<tr>
<td>( C^{(k)} \leftarrow \delta^{(k)}<em>* (\gamma^{(k)}</em>*)^{-1} )</td>
</tr>
</tbody>
</table>

Therefore Algorithm 4 asymptotically matches the results of both the E and M steps of Algorithm 3 at every iteration \( k \), and this without the need for a fusion center, by using local communications only.

**5. SIMULATIONS**

In this section we illustrate the characteristics of the algorithms CB-DPCA and CB-EM-DPCA through simulation. The application scenario is that of a large-scale relatively sparsely connected WSN made up of 500 nodes \( (N = 500) \), each equipped with two sensors \( (D = 2) \). The nodes are randomly placed and have an average degree of 9. We let each node take five samples \( (t_n = 5) \) from one of three Gaussian distributions. We also let \( P = 1 \), making the principal subspace a line onto which both (centralized and distributed) PCA algorithms project the 2-dimensional data set.

If the nodes operate in isolation, each node \( n \) executes PCA or EM-PCA algorithm on its own data set \( Y_n \) without any cooperation with the neighboring nodes. As can be seen in Figure 2 (a) and (c), the resulting principal components, although locally optimal, are very different from node to node. This makes the use of the results in a compression scheme practically impossible.

If on the other hand, the data is gathered at the fusion center, it is possible to derive the globally optimal principal subspace. This subspace is represented by the broken lines in Figure 2: PCA in (a) and (b) and EM-PCA in (c) and (d). However, these solutions are prohibitively expensive in terms of the energy required to gather all the data samples from all the nodes at the fusion center.

As shown in Sections 3.3 and 4.3, and illustrated in Figure 2 (b) and (d) respectively, CB-DPCA and CB-EM-DPCA asymptotically match the result that a fusion center would achieve, but using only limited local communications among the nodes and only 10 rounds of the embedded consensus algorithm.

A quick comparison between these algorithms can be done in terms of communication cost. CB-DPCA requires to achieve a consensus for \( O(D \cdot D) \) values, while CB-EM-DPCA requires to agree in \( O(P \cdot (D + P) \cdot k) \) values, being \( k \) the number of EM iterations. Hence, in the case \( D \) is very high (e.g. each node is equipped with a camera), CB-EM-DPCA will offer a much cheaper energy budget, which is approximately \( O(P \cdot D \cdot k) \).
6. CONCLUSIONS

We presented two distributed PCA algorithms: CB-DPCA and CB-EM-DPCA. They are the first solutions for fully distributed PCA computation, where no fusion center is required, where convergence to the global solution can be guaranteed, and using only single-hop communications among neighbors. In fact, both algorithms provide a flexible trade-off between the desired performance (tightness of agreement on the local values with the global solution of PC) and the resulting energy expenditure (number of consensus iterations).

Furthermore, the consensus matrix product operator, the distributed data covariance matrix approximation, and the distributed consensus least squares show the power of average consensus as a general tool that can be used to distribute many other centralized algorithms. In particular, the spirit of our solution can be easily extended to many other linear and non-linear subspace methods.

7. REFERENCES