Neural Networks 34 (2012) 96–105

Contents lists available at SciVerse ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Distributed static linear Gaussian models using consensus☆

Pavle Belanovic [a,*], Sergio Valcarcel Macua [b], Santiago Zazo [b]

[a] Telecommunications Circuits Laboratory (TCL), École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
[b] Escuela Técnica Superior de Ingenieros de Telecomunicación, Universidad Politécnica de Madrid (UPM), Spain

ARTICLE INFO

Article history:
Received 18 October 2011
Revised and accepted 12 July 2012

Keywords:
Principal component analysis
Factor analysis
Distributed systems
Consensus
Gossip

ABSTRACT

Algorithms for distributed agreement are a powerful means for formulating distributed versions of existing centralized algorithms. We present a toolkit for this task and show how it can be used systematically to design fully distributed algorithms for static linear Gaussian models, including principal component analysis, factor analysis, and probabilistic principal component analysis. These algorithms do not rely on a fusion center, require only low-volume local (1-hop neighborhood) communications, and are thus efficient, scalable, and robust. We show how they are also guaranteed to asymptotically converge to the same solution as the corresponding existing centralized algorithms. Finally, we illustrate the functioning of our algorithms on two examples, and examine the inherent cost-performance trade-off.

1. Introduction

There exists a strong trend in modern signal processing research of moving away from classical centralized processing architectures towards fully distributed ones. These new platforms rely on a (possibly large) number of interconnected nodes to perform any given task without relying on any central entity such as a fusion center. This lends them unmatched adaptability, robustness, and fault-tolerance.

However, this shift of paradigm also creates a need to reinvent traditional algorithms, since they are no longer applicable, for the lack of a fusion center. This is done by designing new distributed implementations aimed at such distributed platforms.

Two distinct and equally important parts of any such implementation are the *intra-node* local processing at each node, and the *inter-node* communications which provides coordination. Only a coupled design of these two components assures the emergence of global behavior matching, or at least approximating, that of the original centralized algorithm.

Investigation of such distributed platforms may be motivated by the wish to understand and recreate emergent behavior in large-scale decentralized biological systems, such as groups of animals (e.g. schools of fish), cardiomyocyte cells, or even the nervous system. Also, the same type of algorithm appears in large-scale computer networks, such as wireless sensor networks.

In this work, we will examine the processing and communications (intra- and inter-node) aspects of performing static linear Gaussian models (SLGMs), all in a distributed way based on consensus and gossip algorithms. As we will see, SLGMs include principal component analysis (PCA), as well as two closely related algorithms, factor analysis (FA) and probabilistic PCA (PPCA).

1.1. Related work

PCA is one of the most fundamental and best known feature extraction algorithms (Hotelling, 1933; Jolliffe, 2002; Pearson, 1901), dating back to the 1930s. Since then it has enjoyed tremendous success in many diverse fields, inspiring numerous variations and extensions.

There exist various partially distributed implementations of PCA. They focus on saving part of the multi-hop communication cost by either local computations (Kargupta, Huang, Sivakumar, & Johnson, 2001) or aggregation services (Bai, Chan, & Luk, 2005; Le Borgne, Raybaud, & Bontempi, 2008; Qi & Wang, 2004), but they still rely on a fusion center for merging the local results. In the context of distributed compression and source coding, Gastpar, Dragotti, and Vetterli (2006) proposed a distributed Karhunen–Loéve transform which is posed as an optimization problem, where convergence to the global optimum is, in general, not assured. Our two consensus-based distributed PCA algorithms (Valcarcel Macua, Belanovic, & Zazo, 2010) outperform all the above because they both guarantee convergence, with no fusion center, just by local neighborhood communications.

☆ This work was supported in part by the Spanish Ministry of Science and Innovation under the grant TEC2009-14219-C03-01; the Spanish Ministry of Science and Innovation in the program CONSOLIDER-INGENIO 2010 under the grant CSD2008-00010 COMONSENS; the European Commission under the grant FP7-ICT-2009-4-248894-WHERE-2; the European Commission under the grant FP7-ICT-223994-N4C and the Spanish Ministry of Science and Innovation under the complementary action grant TEC 2008-04644-E; Spanish Ministry of Science and Innovation under the grant TEC2010-21217-C02-02-CR4HFDVL.

* Corresponding author. Tel.: +41 216931027.
E-mail address: pavle.belanovic@epfl.ch (P. Belanovic).

0893-6080/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2012.07.004

Meanwhile, computing over networks of processing elements is a potent paradigm, offering robust and scalable implementations for a variety of different algorithms (Bertsekas & Tsitsiklis, 1997). The rich body of knowledge on this topic includes parallel, decentralized, and distributed implementations. Parallel computing focuses on splitting the input data, available at once on an incoming bus, into many processing units, each in charge of processing its own slice of information, and the final results are again gathered at the output. Performing algebraic operations on such platforms is well known (van de Geijn, 1997). On the other hand, when we talk of decentralized processing, the original data is spatially dispersed, i.e. partially available at each node, and some local processing is performed before the intermediate results are passed on to a single fusion center, which produces the final result. An excellent example is given in Tsitsiklis (1993).

Finally, in distributed processing, the spatially dispersed data is processed locally, usually in an iterative way, and the intermediate results are communicated only among neighboring nodes. Hence, no fusion center exists, and the final result is available at all the nodes when the iterative process stops. Consensus algorithms, including gossip, have in recent years provided a powerful tool for distributing existing centralized algorithms. For a comprehensive review of consensus and gossip, the reader is directed to Garin and Schenato (2011) and the references therein. Examples of algorithms distributed using consensus are the Kalman filter (Olfati-Saber, 2005), detection (Bajovic, Jakovetic, Xavier, Sinopoli, & Moura, 2011), clustering (Forero, Cano, & Giannakis, 2011), support vector machines (Forero, Cano, & Giannakis, 2010), linear discriminant analysis (Valcarcel Macua, Belanovic, & Zazo, 2011), and many others. In this work we explore the application of consensus algorithms to SLGMs.

1.2. Contributions

The first contribution we present is a "toolkit": a set of matrix operations useful in distributing existing algorithms over networks of nodes. These operations are based on the well-studied average consensus algorithms and include the distributed matrix product, distributed least-squares, and distributed estimation of the first two moments of a multi-dimensional data set.

The main contribution of this article is a direct application of the toolkit: a set of fully distributed algorithms to systematically distribute SLGMs. We begin with two distributed algorithms to perform PCA. The first is a *direct method*, deriving local approximations of the sample covariance matrix of the global data set, and hence the dominant eigenvectors and the principal subspace spanned by these. The other is an *iterative method*, based on an expectation maximization (EM) procedure, producing local approximations of the global principal subspace. Both algorithms are guaranteed to asymptotically converge to the centralized solution given by classical PCA.

In addition, both algorithms are based only on local computations, with strictly limited communications among nodes, only via consensus iterations. These low-volume communications do not grow with the number of data samples and involve only neighboring (1-hop) nodes. Hence, the presented algorithms scale excellently and are applicable to arbitrarily large networks.

We then present two extensions of our iterative algorithm, to distribute FA and PPCA algorithms. Although, as already stated, a multitude of variations of the PCA, PPCA, and FA algorithms exists, here we focus only on their basic forms in order to illustrate our key contribution. The application of this or similar methods to the numerous other variants of each algorithm falls outside the scope of this particular contribution.

Our final contribution are two experimental examples of the use of our algorithms, in distributed scenarios over large, sparse networks, representing the most difficult type of system configuration.

1.3. Outline

We start this article in Section 2 with a brief unifying review of SLGMs. Then in Section 3 we offer a description of the system model we will be considering, followed by an overview of distributed agreement algorithms, focusing on average consensus. These algorithms are summarized in their scalar, vector, and matrix forms.

Based on these well-known algorithms we present our first contribution in Section 4: a toolkit of techniques for distributing algorithms using consensus interactions.

Then, in Section 5, we present two distributed forms of performing PCA based on average consensus; one direct and the other iterative. In the same section we also show two further algorithms, to distributed FA and PPCA, which are extensions of the iterative PCA algorithm.

In order to illustrate the functioning of these algorithms, we show two experiments in Section 6. The first demonstrates the gain achieved by all the nodes through (limited) cooperation via consensus interactions, while the second shows the application of our algorithms in typical real-world scenarios where PCA is known to be useful.

Finally, we conclude the article in Section 7.

2. Static linear Gaussian models

Roweis and Ghahramani (1999) showed how a single mathematical model, and a rather simple and commonly seen one at that, can be used to represent fully many different popular algorithms, such as PCA, FA, PPCA, Independent Component Analysis (ICA), Hidden Markov Models (HMM) and the Kalman filter, among others.

The mathematical model is that of a hidden system being imperfectly observed. It is composed of two rather generic, linear, discrete-time, difference equations for the *state* and the *observation*. In the first, the hidden system progresses through a number of states, $\mathbf{x}$, as governed by the state transition matrix $\mathbf{A}$, and affected by state noise $\mathbf{w}_\bullet \sim \mathcal{N}(0, \mathbf{Q})$.

$$\mathbf{x}[k] = \mathbf{A}\mathbf{x}[k-1] + \mathbf{w}_\bullet. \tag{1}$$

In the second equation, noisy observations $\mathbf{y}$ are produced from the system state through an observation matrix $\mathbf{C}$, and are also affected by observation noise $\mathbf{v}_\bullet \sim \mathcal{N}(0, \mathbf{R})$.

$$\mathbf{y}[k] = \mathbf{C}\mathbf{x}[k] + \mathbf{v}_\bullet. \tag{2}$$

We note that both noise variables are represented without a time index k, to emphasize the fact that their realizations are iid, i.e. not to be seen as a sequence.

For obvious reasons the term *linear Gaussian models* is used to refer to all the models united under this umbrella. A particular subset of these are the static linear Gaussian models (SLGMs), in which $\mathbf{A} = \mathbf{0}$, so that (1) reduces to

$$\mathbf{x}_\bullet = \mathbf{w}_\bullet \tag{3}$$

and (2) becomes

$$\mathbf{y}_\bullet = \mathbf{C}\mathbf{x}_\bullet + \mathbf{v}_\bullet. \tag{4}$$

In other words, in these models the time index is lost, as all the states $\mathbf{x}_\bullet$ (and consequently the observations $\mathbf{y}_\bullet$ as well) are iid realizations without any particular (temporal) ordering.

The differentiation among the different SLGMs comes from the ways of constraining, or modeling $\mathbf{R}$, the covariance matrix that controls the observation noise. As we will see later, $\mathbf{R}$ may be assumed to vanish (PCA), be a scaled identity matrix (PPCA), or diagonal (FA).

In this paper we present a systematic way of distributing SLGMs using distributed agreement, and present how the particular assumptions on $\mathbf{R}$ of each SLGM affects the distributed algorithms we propose.