A transfer learning approach for network modeling

Shuai Huang, Jing Li, Kewei Chen, Teresa Wu, Jieping Ye, Xia Wu & Li Yao

To cite this article: Shuai Huang, Jing Li, Kewei Chen, Teresa Wu, Jieping Ye, Xia Wu & Li Yao (2012) A transfer learning approach for network modeling, IIE Transactions, 44:11, 915-931, DOI: 10.1080/0740817X.2011.649390

To link to this article: http://dx.doi.org/10.1080/0740817X.2011.649390
A transfer learning approach for network modeling

SHUAI HUANG1, JING LI1,*, KEWEI CHEN2, TERESA WU1, JIEPING YE3, XIA WU4 and LI YAO4,5

1Industrial Engineering, School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85287, USA
2Banner Alzheimer’s Institute, Phoenix, AZ 85006, USA
3Computer Science, School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85287, USA
4School of Information Science and Technology, Beijing Normal University, Beijing, People’s Republic of China
5State Key Laboratory of Cognitive Neuroscience and Learning, Beijing Normal University, Beijing, People’s Republic of China

Received November 2010 and accepted August 2011

Network models have been widely used in many subject areas to characterize the interactions between physical entities. A typical problem is to identify the network for multiple related tasks that share some similarities. In this case, a transfer learning approach that can leverage the knowledge gained during the modeling of one task to help better model another task is highly desirable. This article proposes a transfer learning approach that adopts a Bayesian hierarchical model framework to characterize the relatedness between tasks and additionally uses $L_1$-regularization to ensure robust learning of the networks with limited sample sizes. A method based on the Expectation–Maximization (EM) algorithm is further developed to learn the networks from data. Simulation studies are performed that demonstrate the superiority of the proposed transfer learning approach over single-task learning that learns the network of each task in isolation. The proposed approach is also applied to identify brain connectivity networks associated with Alzheimer’s Disease (AD) from functional magnetic resonance image data. The findings are consistent with the AD literature.

Keywords: Transfer learning, graphical models, brain networks, fMRI, Bayesian hierarchical models

1. Introduction

Network models have been extensively used in many subject areas to characterize the interactions between physical entities. For example, they have been used to model how different genes interact in a biological process, and the resulting networks are called gene association networks (Friedman et al., 2000). They have been used to model how different brain regions interact to jointly deliver a brain function such as cognition and emotion, and the resulting networks are called brain connectivity networks (Huang et al., 2010). They have also been used to model the relationship between process and product quality variables for quality control of manufacturing processes (Li and Shi, 2007). The advances in sensing technologies have made it possible to collect large amounts of data from which network models can be learned/identified. These data include, for example, gene micro-arrays, brain images, and production data for the aforementioned networks, respectively. Note that the network models focused on in this article are also called graphical models. Obtaining graphical models from data is a popular research area in statistics and machine learning.

Existing research in graphical models focuses on obtaining a network/graphical model for a single task. However, many real-world problems involve learning network models for multiple related tasks (i.e., one model for each task). For example, there may be a group of Alzheimer’s Disease (AD) patients for each of whom we want to learn a brain connectivity network based on his/her Functional Magnetic Resonance Image (fMRI) data. The purpose of such a study may be to identify brain connectivity patterns common to AD patients, which have the potential for being used as AD biomarkers to help clinical diagnosis. Here, each patient is a task; these tasks/patients are related in the sense that they have the same disease and thus their respective brain connectivity networks may share some similarities. Because of the similarities, the networks of the AD patients should be learned jointly, rather than independently, to leverage the knowledge gained in the network modeling of one patient to help better model another patient. This kind of joint learning is called transfer learning in this article. Transfer learning is especially useful when the data of each task has a low sample size, such as the fMRI data of each AD patient. In this case, transfer learning allows for use of data of other related tasks, in an appropriate
way, to compensate for the sample shortage in each task.

Transfer learning is useful not only in the aforementioned multi-subject studies but also in studies on the evolution of a process. For example, it may be of interest to learn brain connectivity networks at several different time points of an AD patient to track the disease progression. Despite their differences, these networks should share some similarities because they all correspond to the same patient and disease progression is a continuous process. As a result, transfer learning can be used to obtain these networks jointly to enable knowledge transfer between them.

In addition to brain connectivity networks, transfer learning may be useful in other applications. For example, it may be used for modeling gene association networks of patients with the same type of cancer or gene association networks at various time points of the progression of the disease suffered by a cancer patient. As another example, transfer learning may be used for modeling process–quality interactions of several products belonging to the same product family or the process–quality interactions of different generations of a product.

Transfer learning is a natural skill of human beings. For example, we may find that learning to recognize apples may help the recognition of pears; learning to play an electronic organ may help learning the piano. Transfer learning in statistics and machine learning has focused on predictive models such as regressions and neural networks (Caruana, 1997; Baxter, 2000; Bakker and Heskes, 2003; Lawrence and Platt, 2004; Zhang et al., 2006). One major difference of these existing works is that they characterize the relatedness between tasks in different ways. Thrun and O’Sullivan (1996) proposed the use of a distance metric to evaluate relatedness between tasks. In the setting of neural networks, task relatedness can be reflected by shared hidden nodes between tasks (Caruana, 1997). The recent work by Zhang et al. (2006) assumed that task parameters are generated from independent sources that account for the relatedness between the tasks. Several studies have adopted the Bayesian hierarchical modeling framework and taken the relatedness into account by placing a common prior on model parameters of the tasks (Lawrence and Platt, 2004; Xue et al., 2007). Despite the popularity of transfer learning in predictive models, relatively less work has been done on transfer learning of graphical/network models. This article intends to bridge this gap.

In this article, we propose a transfer learning approach for network modeling of multiple related tasks. We focus on one particular type of network model called a Gaussian Graphical Model (GGM). A GGM consists of nodes that are random variables that follow a multivariate normal distribution and undirected arcs that indicate non-zero partial correlations between variables. Various methods have been developed for obtaining a GGM from data, which are also known as methods for Inverse Covariance (IC) estimation, because the undirected arcs in a GGM correspond to non-zero entries in the IC matrix of the data. These methods are reviewed in the following discussion.

One class of methods for GGM learning is based on regression. For example, a variable-by-variable approach for neighborhood selection via the LASSO regression was developed by Meinshausen and Buhlmann (2006). A joint sparse regression model, which simultaneously performs neighborhood selection for all variables, was developed by Schafer and Strimmer (2005). A sparse regression technique called SPACE, which is particularly useful in identifying hubs in gene association networks, was developed by Peng et al. (2009). Another class of methods employs the maximum likelihood framework. A penalized maximum likelihood approach that performs model selection and estimation simultaneously was proposed by Yuan and Lin (2007). Furthermore, efficient algorithms were proposed by Friedman et al. (2007) and Levina et al. (2008) to implement the penalized maximum likelihood approach proposed by Yuan and Lin (2007), which are applicable to high-dimensional problems. Various other methods have been proposed, such as a method based on threshold gradient regularization developed by Li and Gui (2006) and a method for overcoming the ill-conditioned problem of the sample covariance matrix by Schafer and Strimmer (2005). In addition, there are methods that deal with the situation in which variables have a natural ordering (Bickel and Levina, 2008; Levina et al., 2008).

Different from these existing methods, the approach we propose enables the use of transfer learning in the learning of GGMs for multiple related tasks. Specifically, we adopt the Bayesian Hierarchical Modeling (BHM) framework in our problem formulation to characterize the relatedness between tasks. We further add $L_1$-regularization to our problem formulation. $L_1$-regularization is well known to be able to discourage truly zero parameters or small-valued parameters from showing up in the obtained model. This is especially advantageous when the model is high-dimensional and the sample size is limited, in which case conventional statistical estimation methods without regularization, such as Maximum Likelihood Estimation (MLE), may generate unreliable estimates. $L_1$-regularization has been utilized in regressions (Tibshirani, 1996) and graphical models without transfer learning (Friedman et al., 2007) and has demonstrated its effectiveness. Under the proposed problem formulation, we further develop a method based on the Expectation–Maximization (EM) algorithm (Dellaert, 2002) to solve the problem; i.e., to jointly obtain GGMs for multiple related tasks by utilizing transfer learning. Furthermore, we conduct simulation studies to compare the performance of the proposed transfer learning approach with single-task learning that learns a GGM for each task in isolation. Finally, we apply the proposed approach to a real-world application of brain connectivity network modeling for AD based on fMRI data. Fifteen AD patients are considered as related tasks and the purpose is to identify common patterns shared
by their brain connectivity networks, in contrast with the normal brain connectivity networks of 16 matched normal controls.

2. Introduction to GGM

A GGM consists of nodes, $X = \{X_1, \ldots, X_p\}$, and undirected arcs. The nodes are random variables following a multivariate normal distribution; i.e., $X \sim N_p(\mu, \Sigma)$. Let $\Theta$ be the IC matrix of the distribution; i.e., $\Theta = \Sigma^{-1}$. There is an arc between nodes $X_i$ and $X_j$ if and only if the entry at the $i$th row and $j$th column of $\Theta$ is non-zero. Please see Fig. 1 for an example of a GGM and the corresponding IC matrix.

Given data on the nodes, the GGM can be obtained by estimating the IC matrix. It is straightforward to derive the log-likelihood of $\Theta$, which is $\log |\Theta| - \text{tr}(S\Theta)$ where $|\cdot|$ and $\text{tr}(\cdot)$ denote the determinant and trace of a matrix, respectively, and $S$ the sample covariance matrix. By maximizing the log-likelihood, we can obtain the MLE for $\Theta$, which is $\hat{\Theta}^{\text{MLE}} = S^{-1}$. However, for large-scale GGMs with limited sample sizes, the MLE may be quite unreliable in the sense that many zero entries in $\Theta$ may be non-zero in $\hat{\Theta}^{\text{MLE}}$, leading to a densely connected GGM that is hard to interpret. In the extreme case when the sample size is less than the number of nodes, $S$ is not invertible. To tackle this deficiency of the MLE, a well-known strategy is to maximize the $L_1$-regularized log-likelihood function; that is,

$$\hat{\Theta} = \arg\max_{\Theta \succ 0} \log |\Theta| - \text{tr}(S\Theta) - \lambda \|\Theta\|_1,$$  

(1)

where $\|\Theta\|_1$ is the so-called $L_1$-norm of $\Theta$; i.e., the sum of the absolute values of the entries in $\Theta$. $\lambda$ is the regularization parameter; the larger the $\lambda$, the more zero entries in $\hat{\Theta}$; i.e., $\hat{\Theta}$ will be sparser. $\lambda$ can be specified by the user or cross-validation. Equation (1) is best known as the method of graphical LASSO (Friedman et al., 2007), which can be efficiently solved by the Block Coordinate Descent (BCD) algorithm (Sun et al., 2009).

3. Problem formulation for transfer learning in GGM

Assume that there are $m$ related tasks. To estimate the IC matrix for each task, $\Theta_i$, the graphical LASSO in Equation (1) may be used; that is,

$$\hat{\Theta}_i = \arg\max_{\Theta_i \succ 0} \log |\Theta_i| - \text{tr}(S_i\Theta_i) - \lambda_i \|\Theta_i\|_1,$$  

(2)

where $S_i$ is the sample covariance matrix for task $i$, $i = 1, \ldots, m$. This method treats the tasks as independent and does not exploit their relatedness. Alternatively, we propose to consider the relatedness by assuming that the $\Theta_i$’s are “samples” drawn from the same probability distribution; i.e., we adopt the BHM framework to characterize the relatedness between tasks (Fig. 2). This same probability distribution is chosen to be a Wishart distribution; i.e., $\Theta_i \sim \text{Wishart}_p(\Theta^h)$, where $\Theta^h$ is a $p \times p$ positive-definite matrix (called the scale matrix) and $v > p - 1$ (called the degrees of freedom). This choice is based on the following considerations: (i) the $\Theta_i$’s are symmetric, positive-definite matrices and the Wishart distribution is developed for matrices of such characteristics; (ii) in Bayesian inference, the Wishart distribution is commonly used as the prior distribution of the IC matrix of a multivariate normal distribution; (iii) the degrees of freedom, $v$, of the Wishart distribution can be nicely interpreted in our problem, as will be shown in Section 4.3; and (iv) the scale matrix, $\Theta^h$, of the Wishart distribution depicts how the tasks are related. Specifically, because

$$E(\Theta_i) = v\Theta^h,$$  

(3)

the tasks are related in the sense that their respective IC matrices share the same prior mean.

To enable transfer learning, the $\Theta_i$’s should be estimated by utilizing not only the data, i.e., the $S_i$’s, but also the
parameters of the common Wishart distribution, $\Theta^h$ and $v$. The challenge here is that $\Theta^h$ and $v$ are unknown. $v$ may be specified by the user or identified from data by cross-validation. This strategy, however, does not work for $\Theta^h$ because $\Theta^h$ is a high-dimensional matrix. Therefore, we propose to integrate out $\Theta^h$ during the estimation of the $\Theta_i$’s; i.e., we aim to find an estimate, $\hat{\Theta}_i$, for each $\Theta_i$ that maximizes the logarithm of the posterior probability of the $\Theta_i$’s given the data, marginalizing over $\Theta^h$. This can be written as

$$\{\hat{\Theta}_i\}_{i=1,2,\ldots,m} = \arg\max_{\Theta_i} \int \Theta_i \log P (\{\Theta_i\}_{i=1,2,\ldots,m}, \Theta^h|S_i) \, d\Theta^h$$

(4)

Equivalently, we can maximize the logarithm of the joint probability that is proportional to the posterior probability in Equation (4); that is,

$$\{\hat{\Theta}_i\}_{i=1,2,\ldots,m} = \arg\max_{\Theta_i} \int \Theta_i \log P (\{\Theta_i\}_{i=1,2,\ldots,m}, \Theta^h, S_i) \, d\Theta^h.$$  

(5)

In Equation (5), the $P(\{\Theta_i\}_{i=1,2,\ldots,m}, \Theta^h, S_i)$ can be written as

$$P (\{\Theta_i\}_{i=1,2,\ldots,m}, \Theta^h, S_i) = P (S_i = 1,2,\ldots,m | \{\Theta_i\}_{i=1,2,\ldots,m}, \Theta^h) \times P (\{\Theta_i\}_{i=1,2,\ldots,m} | \Theta^h) \times P (\Theta^h).$$

According to the structure in Fig. 2, the $\Theta_i$’s are conditionally independent given $\Theta^h$, so $P(\{\Theta_i\}_{i=1,2,\ldots,m} | \Theta^h) = \prod_{i=1}^{m} P (\Theta_i | \Theta^h)$. Each $S_i$ is independent of $\Theta^h$ given $\Theta_i$, so $P(S_i = 1,2,\ldots,m | \{\Theta_i\}_{i=1,2,\ldots,m}, \Theta^h) = \prod_{i=1}^{m} P (S_i | \Theta_i)$. Therefore, Equation (5) can be further written as

$$\{\hat{\Theta}_i\}_{i=1,2,\ldots,m} = \arg\max_{\Theta_i} \int \Theta_i \log \left( P(\Theta^h) \prod_{i=1}^{m} P(\Theta_i | \Theta^h) \prod_{i=1}^{m} P(S_i | \Theta_i) \right) \, d\Theta^h,$$

(6)

where

$$(\Theta_i | \Theta^h) \propto |\Theta^h|^{-v/2} |\Theta_i|^{(v-p-1)/2} e^{-tr(\Theta^{-1} \Theta_i)/2},$$

and

$$P(S_i | \Theta_i) \propto |\Theta_i|^{ni/2} e^{-n_i \cdot tr(S_i \Theta_i)/2}.$$  

Then, Equation (6) becomes

$$\{\hat{\Theta}_i\}_{i=1,2,\ldots,m} = \arg\max_{\Theta_i} \int \Theta_i \sum_{i=1}^{m} \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} tr(\Theta^{-1} \Theta_i) - \frac{n_i}{2} tr(S_i \Theta_i) \right\} \, d\Theta^h$$

(7)

Moreover, due to the same consideration as graphical LASSO, we add $L_1$-regularization to Equation (7), which gives the final objective function:

$$\{\hat{\Theta}_i\}_{i=1,2,\ldots,m} = \arg\max_{\Theta_i} \int \Theta_i \sum_{i=1}^{m} \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} tr(\Theta^{-1} \Theta_i) - \frac{n_i}{2} tr(S_i \Theta_i) - \lambda_i \cdot \Theta_i \right\} \, d\Theta^h,$$

(8)

4. Problem solution using an EM algorithm

It is difficult to solve Equation (8) directly, since it involves an integral over a high-dimensional matrix $\Theta^h$. We propose a method based on the EM algorithm. Adopting the EM algorithm is a natural choice because we want to integrate out $\Theta^h$ in Equation (8), i.e., treat $\Theta^h$ as latent, and the EM algorithm is a well-known approach for model estimation with latent variables. In this section, we first introduce the general EM algorithm and then describe the proposed method.

4.1. Introduction to the general EM algorithm

Consider a probabilistic model that depends on some observed variables, $Y$, and some unobserved latent variables, $Z$. Let $\Omega$ denote the parameters of the model that need to be estimated. One way of estimation is to find estimates for $\Omega$ that maximize the posterior probability of $\Omega$ given $Y$ and $Z$. However, as $Z$ is latent, the estimation cannot be done directly but rather through an iterative algorithm that alternates between an E (expectation) step and an M (maximization) step. Specifically, the E step is to calculate the expectation of the logarithm of the joint distribution of $\Omega$, $Y$ and $Z$ (which is proportional to the posterior probability of $\Omega$ given $Y$ and $Z$), with respect to the conditional distribution of $Z$ given $Y$ under the current (i.e., the $t$th iteration) estimates for the parameters $\Omega^t$. Denote this expectation by $Q(\Omega|\Omega^t)$.

**E step:**

$$Q(\Omega|\Omega^t) = E_{Z|Y,\Omega^t} \left[ L(\Omega, Z, Y) \right]$$

$$= \int_Z L(\Omega, Z, Y) \, P(Z|Y, \Omega^t) \, dZ,$$

where $L(\Omega, Z, Y) = \log P(\Omega, Z, Y)$. Then, the M step is to find the parameters that maximize $Q(\Omega|\Omega^t)$:

**M step:**

$$\Omega^{t+1} = \arg\max_{\Omega} Q(\Omega|\Omega^t)$$

These parameters $\Omega^{t+1}$ are then used for the next E step. Such iterations have been proven to converge (Wu, 1983).
4.2. Solving the transfer learning formulation under the EM framework

To fit the transfer learning formulation in Equation (8) into the EM framework, we consider \( \{ \Theta_i \}_{i=1,2,...,m} \), \( \Theta^k \), and \( \{ S_i \}_{i=1,2,...,m} \) to be the parameters to be estimated, the latent and the observed variables (i.e., \( \Omega, Z \), and \( Y \)), respectively. Then, the E step is to calculate \( \log L((\Theta_i)_{i=1,2,...,m}, \Theta^k, \{ S_i \}_{i=1,2,...,m}, \{ \Theta_i' \}_{i=1,2,...,m})d\Theta^k \). According to Fig. 2, \( \Theta^k \) is independent of \( \{ S_i \}_{i=1,2,...,m} \) given \( \{ \Theta_i' \}_{i=1,2,...,m} \), i.e., \( P(\Theta^k|\{ S_i \}_{i=1,2,...,m}, \{ \Theta_i' \}_{i=1,2,...,m}) = P(\Theta^k|\{ \Theta_i' \}_{i=1,2,...,m}) \). Also, according to Equation (8):

\[
L((\Theta_i)_{i=1,2,...,m}, \Theta^k, \{ S_i \}_{i=1,2,...,m}) = \sum_{i=1}^{m} \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} \text{tr} (\Theta_i^{-1} \Theta_i) - n_i \text{tr} (S_i \Theta_i) - \frac{1}{2} \left( m v - p - 1 \right) \left( \sum_{i=1}^{m} \Theta_i' \right)^{-1} \Theta_i \right\}.
\]

(9)

Then, the E and M steps are

**E step:** \( \mathbb{Q}(\{ \Theta_i \}_{i=1,2,...,m}|\{ \Theta_i' \}_{i=1,2,...,m}) = \int \mathbb{Q}(\Theta_i) L((\Theta_i)_{i=1,2,...,m}, \Theta^k, \{ S_i \}_{i=1,2,...,m}) d\Theta^k \),

\[
\mathbb{Q}(\{ \Theta_i \}_{i=1,2,...,m}|\{ \Theta_i' \}_{i=1,2,...,m}) = \frac{\mathbb{Q}(\{ \Theta_i \}_{i=1,2,...,m}) L(\{ \Theta_i \}_{i=1,2,...,m}, \Theta^k, \{ S_i \}_{i=1,2,...,m})}{\int \mathbb{Q}(\Theta_i) L((\Theta_i)_{i=1,2,...,m}, \Theta^k, \{ S_i \}_{i=1,2,...,m}) d\Theta^k},
\]

(10)

where \( L(\cdot) \) is given in Equation (9).

**M step:** \( \{ \Theta_i^{(t+1)} \}_{i=1,2,...,m} = \arg \max_{\{ \Theta_i \}_{i=1,2,...,m}} \mathbb{Q}(\{ \Theta_i \}_{i=1,2,...,m}|\{ \Theta_i' \}_{i=1,2,...,m}) \).

To conduct the E and M steps, we further decompose them into sub-steps.

4.2.1. Conducting the E step

We decompose the E step into two sub-steps: finding the parametric form of the distribution \( P(\Theta^k|\{ \Theta_i' \}_{i=1,2,...,m}) \) and then transforming the \( \mathbb{Q}(\{ \Theta_i \}_{i=1,2,...,m}|\{ \Theta_i' \}_{i=1,2,...,m}) \) into a form that facilitates the maximization in the M step. Results of these two sub-steps are summarized in Propositions 1 and 2, respectively (proofs are given in the Appendix).

**Proposition 1.** The probability distribution of \( \Theta^k|\{ \Theta_i' \}_{i=1,2,...,m} \) is an inverse Wishart distribution with scale matrix \( \sum_{i=1}^{m} \Theta_i' \) and degrees of freedom \( mv - p - 1 \).

**Proposition 2.** The \( \mathbb{Q}(\{ \Theta_i \}_{i=1,2,...,m}|\{ \Theta_i' \}_{i=1,2,...,m}) \) can be decomposed into a sum of \( m \) terms, with the \( i \)th term involving only \( \Theta_i \), not \( \Theta_j \) \( i \neq j \); that is,

\[
(\Theta_i)_{i=1,2,...,m}|\{ \Theta_i' \}_{i=1,2,...,m}) = \sum_{i=1}^{m} \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} \text{tr} (S_i \Theta_i) - \frac{1}{2} \left( m v - p - 1 \right) \left( \sum_{i=1}^{m} \Theta_i' \right)^{-1} \Theta_i \right\}.
\]

(11)

4.2.2. Conducting the M step

We also decompose the M step into two sub-steps. First, as a result of Proposition 2, the maximization in the M step can be carried out by solving \( m \) smaller-scale maximization problems.

**Corollary 1.** Finding the \( \{ \Theta_i^{(t+1)} \}_{i=1,2,...,m} \) can be achieved by solving \( m \) optimization problems; that is,

\[
\Theta_i^{(t+1)} = \arg \max_{\Theta_i} \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} \text{tr} (S_i \Theta_i) - \frac{1}{2} \left( m v - p - 1 \right) \left( \sum_{i=1}^{m} \Theta_i' \right)^{-1} \Theta_i \right\}.
\]

(12)

Next, we need to develop an efficient algorithm to solve each optimization in Equation (11). Through some algebra, it can be found that solving Equation (11) is equivalent to solving Equation (12):

\[
\Theta_i^{(t+1)} = \arg \max_{\Theta_i} \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \text{tr} (S_i' \Theta_i) - \frac{1}{2} \lambda_i \| \Theta_i \|_1 \right\}.
\]

(13)

Equation (12) has a form similar to the graphical LASSO in Equation (2). The difference is that the graphical LASSO has \( \Theta_i' \) instead of \( \Theta_i' \); so it does not enable transfer learning. To solve Equation (12), we adopt the efficient BCD algorithm (Sun et al., 2009), which was originally developed for graphical LASSO, to our problem and prove the convergence (see Appendix).

To conclude this section, we give the practical steps for solving the transfer learning formulation in learning the GGMs of related tasks in Fig. 3.

4.3. Analysis of transfer learning

Here, we would like to discuss some intuition and rationale behind the proposed transfer learning approach. Because of Equation (3), i.e., \( E(\Theta_k) = v \Theta^h \), the \( \sum_{k=1}^{m} \Theta_k' \) in Equation (13) can be considered as the estimate for \( mv \Theta^h \) obtained at iteration \( t \) of the EM algorithm; that is,

\[
\sum_{k=1}^{m} \Theta_k' = mv \Theta^h \cdot t.
\]

(14)

Inserting Equation (14) into Equation (13) leads to

\[
S_i = \frac{n_i S_i + ((mv - p - 1)/mv) (\Theta_i^{h,t})^{-1}}{v-p+1+n_i}
\]

(15)

Equation (15) indicates that \( S_i \) is a combination of two information sources: \( S_i \), which is specific to task \( i \), and \( \Theta_i^{h,t} \), which is common to all of the tasks. According to Equation (14), \( \Theta_i^{h,t} = \sum_{k=1}^{m} \Theta_k/mv \); i.e., \( \Theta_i^{h,t} \) utilizes the information
in all the tasks. Therefore, $S_i$ embraces not only the information specific to task $i$ but also information in other tasks through $\Theta_{h,t}$. Because $S_i$ is used in Equation (12) to obtain the IC matrix of task $i$, the learning process makes use of the information in all the tasks; in other words, the information in other tasks is “transferred” into the process of learning the model for of task $i$. This is the fundamental difference between the proposed approach and the graphical LASSO, in which learning the IC matrix of task $i$ only utilizes $S_i$.

An interesting observation on Equation (15) is that, for a given dataset (i.e., $n_i$, $p$, and $m$ are fixed), $v$ determines the relative weights of the two information sources. The larger the $v$, the more the $\Theta_{h,t}$ will be utilized in obtaining $S_i$ and further in learning the IC matrix of task $i$. In other words, by specifying a larger $v$, we want more information in other tasks to be transferred to the process of learning the model for of task $i$. However, the value of $v$ can be hardly known a priori in practical data analysis. We propose the use of cross-validation for selecting an appropriate $v$, which will be discussed in the next section.

Also, Equation (15) reveals how transfer learning is affected by $n_i$, $p$, and $m$. Specifically, with other parameters fixed, more information in task $i$ will be utilized when task $i$ has more samples; i.e., bigger $n_i$. Furthermore, when more tasks are considered together (bigger $m$), $\Theta_{h,t}$ will be weighted heavier relative to $S_i$. This makes sense because more tasks will help obtain a more reliable estimate for $\Theta_{h,t}$.

4.4. Selection of parameters $v$ and $\lambda_i$

To apply the proposed algorithm in Fig. 3 to solve Equation (8), $v$ and $\lambda_i (i = 1, \ldots, m)$ need to be specified. This can be accomplished by cross-validation. For example, if a $V$-fold cross-validation is used, the following steps can be performed. First, the dataset of each task is partitioned into $V$ subsamples. $V-1$ subsamples are used to create an IC matrix for each task by applying the proposed algorithm at a given $v$ and $\lambda$. Here, we assume the same $\lambda$ for all the tasks for simplicity. Next, based on the created IC matrix of each task, the likelihood of the data in the remaining one subsample in this task is computed. The likelihood values corresponding to all of the tasks are summed together and we obtain an “overall” likelihood for the given $v$ and $\lambda$. This process is repeated $V$ times for the same $v$ and $\lambda$, and each time a different subsample is used to compute the overall likelihood. Then, the average overall likelihood is computed over the $V$ repetitions. The final $v$ and $\lambda$ selected are the ones maximizing the average overall likelihood. Note that this proposed cross-validation procedure is similar to likelihood-based cross-validation, which has been commonly used in probability density function estimation.
5. Simulation study

The simulation study consists of the following steps:

**Step 1.** Construct the common prior mean matrix, $\Theta^h$. Because $E(\Theta_s) = v \Theta^h$, we first need to construct $\Theta^h$. Specifically, the initial value for $\Theta^h, \hat{\Theta}^h$, is generated by

$$
\tau_{jk}^h = \begin{cases} 
1 & j = k \\
0 & j \neq k, j \leftrightarrow k, \\
\sim \text{Uniform}(D) & j \neq k, j \leftrightarrow k 
\end{cases}
$$

where $\tau_{jk}^h$ denotes the entry at the $j$th row and $k$th column of $\Theta^h$; $j \leftrightarrow k$ means that there is an arc between nodes $j$ and $k$, and $j \neq k$ means otherwise; $D = [-1, -0.5] \cup [0.5, 1]$. $\Theta^h$ is then rescaled to ensure the positive-definite property. The rescaling includes first summing the absolute values of the off-diagonal entries for each row, then dividing each off-diagonal entry by 1.5 times the sum, and finally averaging the resulting matrix with its transpose to ensure the symmetry. This rescaling process was suggested by Peng et al. (2009). The rescaled matrix is $\tilde{\Theta}^h$. Furthermore, let $\Theta^h = \tilde{\Theta}^h / v$.

**Step 2.** Construct the true IC matrices of $m$ related tasks $\Theta_i$, $i = 1, \ldots, m$.

To generate a $\Theta_i$, the following substeps are performed.

2.1: Randomly modify $(100 - s\%)$ of the non-zero entries in $\Theta^h$ to be zero.

2.2: Randomly modify the same number of zero entries in $\Theta^h$ to be non-zero.

This is to ensure that each $\Theta_i$ has the same sparsity as $\Theta^h$. Each of these non-zero entries is sampled from Uniform($D$).

2.3: For the remaining unmodified non-zero entries in $\Theta^h$, resample their values from Uniform($D$) and add the resulting value of each non-zero entry with the entry in $\Theta^h$ at the same row and column.

This is to ensure $E(\Theta_i) = v \Theta^h$. An example for this step is given in Fig. 4.

It can be seen that $s\%$ reflects how much the IC matrix of each task, $\Theta_i$, is related to $\Theta^h$. The higher the $s\%$, the higher the level of relatedness between tasks. Considering the extreme case when $s\% = 100\%$, the $\Theta_i$ constructed following substeps (2.1) to (2.3) will have the same positions of non-zero entries as $\Theta^h$, so the GGM corresponding to $\Theta_i$ will look the same as that corresponding to $\Theta^h$. When $s\% = 0$, the $\Theta_i$ will have completely different positions of non-zero entries from $\Theta^h$, so their GGMs will not share even a single arc. Through relating to $\Theta^h$, the IC matrices of all of the tasks are related, so $s\%$ may be considered as an indirect measure of the extent of the relatedness between tasks. It makes sense to specify the relatedness between $\Theta_i$ to $\Theta^h$, rather than specifying the relatedness between the $\Theta_i$ directly, because transfer learning between tasks is through $\Theta^h$, the common prior shared by all of the tasks.

**Step 3.** Generate simulation data for each task.

A dataset consisting of $n$ independently and identically distributed observations is generated from a multivariate normal distribution with mean zero and IC matrix $\Theta_i$, for task $i$, $i = 1, \ldots, m$.

**Step 4.** Estimate the IC matrix of each task.

We apply the transfer learning approach to the data and estimate the IC matrix of each task.

**Step 5.** Measure the performance.

In order to apply the transfer learning approach, we need to choose values for $\lambda$ and $v$. To assess the overall performance of transfer learning across different choices for $\lambda$ and $v$, Receiver Operating Characteristic (ROC) curves are employed. An ROC curve plots the number of true positives versus false positives over all possible choices for $\lambda$ and $v$. Here, we count one true positive if a non-zero entry in the true IC matrix is also non-zero in the estimated IC matrix; we count one false positive if a zero entry in the true IC matrix is non-zero in the estimated IC matrix. Thus, the number of true positives

$$
\Theta_i = \begin{bmatrix}
\theta_{i1} & \theta_{i2} & 0 & 0 \\
0 & \theta_{i2} & 0 & 0 \\
0 & 0 & \theta_{i3} & 0 \\
0 & 0 & 0 & \theta_{i4}
\end{bmatrix}
$$

$$
\Theta_i = \begin{bmatrix}
0 & \theta_{i1} & 0 & 0 \\
0 & 0 & \theta_{i2} & 0 \\
0 & 0 & 0 & \theta_{i3} \\
0 & 0 & 0 & 0
\end{bmatrix}
$$

$$
\Theta_i = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & \theta_{i1} & 0 & 0 \\
0 & 0 & \theta_{i2} & 0 \\
0 & 0 & 0 & \theta_{i3}
\end{bmatrix}
$$

Fig. 4. Sub-steps for constructing the IC matrix of each task, $\Theta_i$, from $\Theta^h$ ($s\% = 50\%$).
positives (Y-axis of the ROC curve) and the number of false positives (X-axis of the ROC curve) reflects the power of the algorithm to detect non-zero entries and the false alarm error, respectively.

Note that execution of steps 3 and 4 once will generate one ROC curve for each task. To reduce sampling variation, the two steps can be repeated for $N$ times and a mean ROC curve can be generated.

In this section, we compare the performance of the proposed transfer learning approach with the single-task learning approach based on graphical LASSO; i.e., Equation (2). Following a similar procedure to steps 1 to 5, a mean ROC curve can be obtained for each task for single-task learning. Based on the definition of ROC curves, a learning approach is better if its ROC curve is closer to the upper left corner of the plot.

Figures 5 to 7 compare transfer learning and single task learning based on the mean ROC curves for the first task (other tasks show similar patterns and are not shown here due to space limits). The comparison is across various parameter settings, including the number of variables ($p = 50, 100, 200$), the number of related tasks ($m = 2, 5, 10$), and extent of the relatedness between tasks ($s\% = 90\%, 50\%, 0\%$). Small sample sizes are assumed for each task, so the sample size $n$ is set to be equal to $p$. Also, the number of non-zero entries in $\Theta^i$ is set to be equal to $p$, such that the GGM in each task is sparse.

The following observations can be made about the obtained results.

1. The advantage of transfer learning over single-task learning is more significant when the tasks are more related. When the tasks are less related (e.g., Fig. 7), the necessity of using transfer learning is not obvious.
2. The advantage of transfer learning over single-task learning is more significant when there are more related tasks.
3. The performances of both transfer learning and single-task learning degrade as the network becomes larger (i.e., larger $p$). However, the performance of transfer learning may be improved by having more related tasks to compensate for sample shortage, whereas more related tasks do not help single-task learning.

5.1. Computational efficiency

The proposed transfer learning algorithm is very fast. This is because the efficient BCD algorithm is used in the M step of the proposed EM algorithm. Also, the proposed EM algorithm usually takes only three to six iterations to converge. Table 1 shows the CPU time of the simulation studies.

Furthermore, we compare the CPU time of transfer learning with single-task learning. For fair comparison, the BCD algorithm was also used in single-task learning. Table 2 shows the CPU time for single-task learning. It can be seen that single-task learning is, in most cases, about three times faster than transfer learning. This is not surprising since transfer learning takes between three and six iterations to converge and in each iteration the BCD algorithm is applied, which takes a similar amount
of time as the BCD algorithm used in single-task learning. In other words, it is primarily the number of iterations in transfer learning that accounts for the difference between transfer learning and single-task learning in terms of CPU time. Fortunately, transfer learning takes only a few iterations to converge. Although not as fast as single-task learning, transfer learning is still computationally efficient considering that it takes less than 10 minutes to complete with 200 variables and 10 tasks (Table 1).
Table 1. CPU time (in seconds) of the simulation studies by transfer learning

<table>
<thead>
<tr>
<th>m</th>
<th>50</th>
<th>100</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>6.34</td>
<td>25.13</td>
<td>102.35</td>
</tr>
<tr>
<td>5</td>
<td>15.98</td>
<td>65.57</td>
<td>225.85</td>
</tr>
<tr>
<td>10</td>
<td>36.12</td>
<td>140.76</td>
<td>562.74</td>
</tr>
</tbody>
</table>

6. Application in brain connectivity network modeling of AD

The simulation study in Section 5 has demonstrated the effectiveness of the proposed transfer learning approach. In this section we explore its application to brain connectivity modeling of AD. AD is a fatal, neurodegenerative disorder currently affecting over 5 million Americans. The existing knowledge about AD is very limited and clinical diagnosis is imprecise. Recent studies have found that AD is closely related to alterations in the brain’s connectivity network (Stam et al., 2007; Supekar et al., 2008). Identification of brain connectivity patterns common to AD patients provides the potential for identifying AD biomarkers to help clinical diagnosis. To explore this goal, we applied the proposed transfer learning approach to the fMRI data of 15 AD patients, which were considered as 15 related tasks, and obtained a brain connectivity network for each patient. To provide a contrast for the AD connectivity networks, the networks of 16 Normal Controls (NCs), who were considered to be another set of related tasks, were also obtained using the proposed transfer learning approach.

The selection criteria for the 15 AD patients and 16 NCs were as follows. The AD patients were aged between 53 and 79, right-handed, free of other diseases such as stroke and focal pathology, and with MMSE scores between zero and 20. MMSE is a clinical instrument for cognitive assessment; the lower the score the more severe the dementia. The NCs were selected to purposely match the AD patients in terms of other selection criteria except MMSE scores; i.e., they were in the same age cohort as the AD patients, right-handed, free of other diseases, but with MMSE scores between 27 and 30 (i.e., they do not have dementia).

The fMRI data of each of the 31 subjects at his/her resting state was obtained using the 3-Tesla Siemens whole-body MRI system at Tiantan Hospital in Beijing, China. We applied the following pre-processing steps to the data. First, we applied the automated anatomical labeling technique (Tzourio-Mazoyer et al., 2002) to segment the whole brain of each subject into 116 regions. A total of 42 regions, whose names are given in Table 3, were selected since they have been proposed in the literature to be potentially related to AD. These regions are distributed in the four major neocortical lobes of the brain; i.e., the frontal, parietal, occipital, and temporal lobes. Next, within each selected region, the voxel-wise fMRI time courses were averaged into one regional average time course. Then, the first five data points for each regional average time course were discarded. Finally, each regional average time course was detrended. After the pre-processing step, the dataset of each subject had the form of a 245 × 42 matrix (245 sampling points in each regional average fMRI time course and 42 selected brain regions).

The datasets obtained through the aforementioned steps can be reasonably considered to follow normal distributions, because each brain region includes at least several hundred voxels and the measurement data for each region is a regional average over the belonging voxels and thus the central limit theorem applies. Also, the normality assumption is common for fMRI studies (Worsley et al., 1997; Valdes-Sosa et al., 2005). We applied the transfer learning approach to the datasets obtained for the 15 AD patients and produced 15 GGMs. Similarly, GGMs were obtained for the 16 NCs. Note that in applying the transfer learning approach, the values for the regularization parameters, λ_i, had to be selected. In this article, we focus on comparing AD patients and NCs in terms of the distribution/organization of the connectivity in the brain, which has been less discussed in the literature, and not in terms of the global scale of the connectivity, which has been extensively discussed in the literature. To achieve this, we had to factor out the connectivity difference between AD and NCs that is due to their difference at the global scale, so that the remaining difference reflects their difference in the connectivity distribution/organization. A common strategy is to control the total number of arcs in the AD patients and NCs connectivity networks to be the same, which has been adopted by a number of other studies (Stam et al., 2007; Supekar et al., 2008). We also adopted this strategy; specifically, we adjusted the λ_i values in estimating the connectivity networks of the 15 AD patients and 16 NC subjects so that all of these networks had the same total number of arcs. Also, by selecting different values for the total number of arcs, we could obtain connectivity networks at different strength levels. Specifically, given a small value for the total number of arcs, only strong arcs will show up in the resulting connectivity networks; when increasing the total number of arcs, mild (or even weak) arcs will also show up in the resulting connectivity networks.

Comparison of AD patients and NCs in terms of connectivity distribution/organization can be achieved by
comparing them in terms of the numbers of arcs within each lobe as well as between each pair of lobes. To be able to assess the statistical significance of the comparison, the 15 (16) GGMs were treated as “samples” of the AD patients (NCs) brain connectivity network. Based on these samples, we generated box plots for the numbers of arcs within and between lobes for AD patients and NCs as shown in Fig. 8(a). Furthermore, the statistical significance of the observed difference between AD patients and NCs in the box plots was assessed by hypothesis testing. The $p$-values of the hypothesis testing are given in Table 4.

The following conclusions can be drawn.

1. The parietal lobe of AD patients has significantly less connectivity than for the NCs. Loss of connectivity in the parietal lobe of AD patients has been previously noted in the literature (Langbaum et al., 2009; Chen et al., 2010).

2. The regions well known to be the first to be severely affected by AD are Hippocampus.L and Hippocampus.R in the temporal lobe, which play an important role in memory. We performed hypothesis testing on the difference between AD patients and NCs in terms of the number of arcs between the Hippocampus.L and Hippocampus.R and other regions in the temporal lobe.

Fig. 8. Box plots of the number of arcs within and between lobes in the connectivity networks obtained using (a) transfer learning and (b) single-task learning. (F: frontal, P: parietal, O: occipital, T: temporal; for the two box plots within a lobe or between a pair of lobes, the left box plot represents AD and the right box plot represents NCs) (color figure provided online).
and obtained $p$-values of 0.0526, 0.0111, and 0.0089 for total numbers of arcs equal to 180, 120, and 90, respectively. This indicates significant loss of connectivity between Hippocampus_L and Hippocampus_R and other regions in AD, which is consistent with several other studies reported in the literature (Wang et al., 2007; Supekar et al., 2008).

3. The frontal lobe of AD patients has significantly more connectivity than the NCs. This is consistent with reports in the literature and has been interpreted as compensatory reallocation or recruitment of cognitive resources (Gould et al., 2006; Stern, 2006). Since regions in the frontal lobe are typically less affected by AD, an increase in the levels of connectivity in the frontal lobe may help preserve some cognitive functions in AD patients. This explanation may be applied to the increase of connectivity between the frontal and temporal lobes in AD patients.

4. All of the above findings are consistent across different total numbers of arcs in the connectivity networks.

For comparison purposes we also applied the graphical LASSO approach to the same datasets. Please note that the graphical LASSO method is a single-task learning approach that models each task independently. Box plots and tabulations of the $p$-values for this case are shown in Fig. 8(b) and Table 5, respectively. It can be seen that the single-task learning approach fails to identify any significant difference between the AD patients and the NCs. The reason, as revealed by the box plots, is that there is large variability in the 15 (16) brain connectivity networks of the AD patients (NCs). This is because the single-task learning approach is not able to make use of task relatedness to compensate for the shortage of data. As a result, the obtained connectivity networks may not be reliable.

### Table 3. Names of the brain regions selected for brain connectivity network modeling (“L” means that the brain region is located in the left hemisphere; “R” means it is located in the right hemisphere)

<table>
<thead>
<tr>
<th>Prefrontal lobe</th>
<th>Parietal lobe</th>
<th>Occipital lobe</th>
<th>Temporal lobe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frontal_Sup_L</td>
<td>13 Parietal_Sup_L</td>
<td>21 Occipital_Sup_L</td>
<td>27 Temporal_Sup_L</td>
</tr>
<tr>
<td>Frontal_Sup_R</td>
<td>14 Parietal_Sup_R</td>
<td>22 Occipital_Sup_R</td>
<td>28 Temporal_Sup_R</td>
</tr>
<tr>
<td>Frontal_Mid_L</td>
<td>15 Parietal_Inf_L</td>
<td>23 Occipital_Mid_L</td>
<td>29 Temporal_Pole_Sup_L</td>
</tr>
<tr>
<td>Frontal_Mid_R</td>
<td>16 Parietal_Inf_R</td>
<td>24 Occipital_Mid_R</td>
<td>30 Temporal_Pole_Sup_R</td>
</tr>
<tr>
<td>Frontal_Sup_Medial_L</td>
<td>17 Precuneus_L</td>
<td>25 Occipital_Inf_L</td>
<td>31 Temporal_Mid_L</td>
</tr>
<tr>
<td>Frontal_Sup_Medial_R</td>
<td>18 Precuneus_R</td>
<td>26 Occipital_Inf_R</td>
<td>32 Temporal_Mid_R</td>
</tr>
<tr>
<td>Frontal_Mid_Orb_L</td>
<td>19 Cingulum_Post_L</td>
<td>33 Temporal_Pole_Mid_L</td>
<td></td>
</tr>
<tr>
<td>Frontal_Mid_Orb_R</td>
<td>20 Cingulum_Post_R</td>
<td>34 Temporal_Pole_Mid_R</td>
<td></td>
</tr>
<tr>
<td>Rectus_L</td>
<td>35 Temporal_Inf_L 8301</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rectus_R</td>
<td>36 Temporal_Inf_R 8302</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cingulum_Ant_L</td>
<td>37 Fusiform_L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cingulum_Ant_R</td>
<td>38 Fusiform_R</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cingulum_Ant_L</td>
<td>39 Hippocampus_L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cingulum_Ant_R</td>
<td>40 Hippocampus_R</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cingulum_Ant_L</td>
<td>41 ParaHippocampal_L</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cingulum_Ant_R</td>
<td>42 ParaHippocampal_R</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 4. P-values for hypothesis testing on AD patients versus NCs comparison based on connectivity networks obtained using the transfer learning approach

<table>
<thead>
<tr>
<th>Total number of arcs</th>
<th>Frontal</th>
<th>Parietal</th>
<th>Occipital</th>
<th>Temporal</th>
</tr>
</thead>
<tbody>
<tr>
<td>180</td>
<td>0.0009</td>
<td>0.7604</td>
<td>1</td>
<td>0.0001</td>
</tr>
<tr>
<td></td>
<td>0.0002</td>
<td>0.5257</td>
<td>0.0125</td>
<td>0.0829</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.2143</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>0.0001</td>
<td>0.5460</td>
<td>0.6229</td>
<td>0.0000</td>
</tr>
<tr>
<td></td>
<td>0.0003</td>
<td>0.0466</td>
<td>0.1153</td>
<td>0.2314</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.0657</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.0033</td>
<td>0.2924</td>
<td>0.3343</td>
<td>0.0005</td>
</tr>
<tr>
<td></td>
<td>0.0002</td>
<td>0.1331</td>
<td>0.1158</td>
<td>0.6210</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.1282</td>
<td></td>
</tr>
</tbody>
</table>
A transfer learning approach for network modeling

Table 5. P-values for hypothesis testing on AD patients versus NCs comparison based on connectivity networks obtained using the single-task approach

<table>
<thead>
<tr>
<th>Total number of arcs</th>
<th>Frontal</th>
<th>Parietal</th>
<th>Occipital</th>
<th>Temporal</th>
</tr>
</thead>
<tbody>
<tr>
<td>180</td>
<td>0.4264</td>
<td>0.5497</td>
<td>0.6440</td>
<td>0.9579</td>
</tr>
<tr>
<td></td>
<td>0.3953</td>
<td>0.6733</td>
<td>0.2215</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.9006</td>
<td>1</td>
<td>0.8740</td>
</tr>
<tr>
<td>120</td>
<td>0.4276</td>
<td>0.5481</td>
<td>0.7669</td>
<td>0.3731</td>
</tr>
<tr>
<td></td>
<td>0.9224</td>
<td>0.9041</td>
<td>0.9764</td>
<td>0.7984</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.6600</td>
<td>0.8111</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.3827</td>
<td>0.6949</td>
<td>0.4827</td>
<td>0.5587</td>
</tr>
<tr>
<td></td>
<td>0.6783</td>
<td>1</td>
<td>0.5018</td>
<td>0.4005</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.9207</td>
<td></td>
</tr>
</tbody>
</table>

7. Conclusions

This article proposed a transfer learning approach to obtain the GGMs of multiple related tasks. The proposed approach adopted the BHM framework in the problem formulation and considered IC matrices of the tasks to be samples drawn from the same Wishart distribution. An $L_1$-regularization was further added to the problem formulation to impose sparsity on the estimation of the GGMs. Under this problem formulation, a method based on the EM algorithm was further developed to obtain the GGMs. Simulation studies showed that the transfer learning approach performs better than the single-task approach. This advantage is more substantial when there are more related tasks or when the tasks are more related to each other. Also, the proposed transfer learning approach was applied to fMRI data of 15 AD patients and 16 NCs for brain connectivity network identification. A comparison of the connectivity networks of AD patients and NCs revealed that AD is associated with decreased connectivity in the parietal lobe and between the hippocampus and other regions in the temporal lobe and increased connectivity in the frontal lobe and between the frontal and temporal lobes. All these findings are consistent with AD pathology and existing findings reported in the literature.

Transfer learning of other types of network models is also of great interest, such as directed models (also known as Bayesian networks) and models of non-Gaussian variables. For example, Bayesian networks have been used to characterize the directional effect of one brain region on another. In addition, from a practical data analysis point of view, it is of interest to examine whether the identified patterns are still present after a perturbation on the data. Specifically, we may create some perturbed datasets out of the original dataset by resampling or adding noise to the original data measurement and then apply the proposed approach to the perturbed datasets and look for patterns that consistently occur across these datasets. This is also a way to ensure the robustness of the results against sampling variability. We will investigate these methodological and practical issues in future work.

Acknowledgements

This material is based in part on work supported by the National Science Foundation under Grant no. 1149602 and 1069246, and the National Institute of Health Grant No. IR01GM096194-01. Also, we would like to thank Yumei Zhang, M.D., and Xiaoting Guan, M.D., of the Tiantan Hospital, Beijing, China, for their assistance with the data collection.

References


**Appendix**

**Proof of Proposition 1.** Using the Bayes’ rule:

\[
P(\Theta^h | \{\Theta^i\}_{i=1,2,...,m}) = \frac{P(\{\Theta^i\}_{i=1,2,...,m}, \Theta^h)}{P(\{\Theta^i\}_{i=1,2,...,m})} = \frac{P(\{\Theta^i\}_{i=1,2,...,m}) P(\Theta^h)}{P(\{\Theta^i\}_{i=1,2,...,m})}
\]

Furthermore, because \(P(\{\Theta^i\}_{i=1,2,...,m} | \Theta^h) = \prod_{i=1}^{m} P(\Theta^i | \Theta^h)\) and \(P(\Theta^h)\) is a constant:

\[
P(\Theta^h | \{\Theta^i\}_{i=1,2,...,m}) \propto \prod_{i=1}^{m} P(\Theta^i | \Theta^h) / P(\{\Theta^i\}_{i=1,2,...,m})
\]

(A1)

Thus, the first step is to derive \(P(\{\Theta^i\}_{i=1,2,...,m})\). To do this, we condition \(P(\{\Theta^i\}_{i=1,2,...,m})\) on \(\Theta^h\); that is,

\[
P(\{\Theta^i\}_{i=1,2,...,m}) = \int_{\Theta^h} P(\{\Theta^i\}_{i=1,2,...,m} | \Theta^h) \times P(\Theta^h) d\Theta^h \propto \int_{\Theta^h} \prod_{i=1}^{m} P(\Theta^i | \Theta^h) d\Theta^h
\]

(A2)

Since \(\Theta^i \sim \text{Wishart}(\Theta^h)\)

\[
P(\Theta^i | \Theta^h) \propto |\Theta^h|^{-v/2} |\Theta^i|^{(v-p-1)/2} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{-1} \Theta^i \right) \right)
\]

(A3)
Inserting Equation (A3) into Equation (A2):

\[
P \left( \{ \Theta_i \}_{i=1,2,\ldots,m} \right) \propto \int_{\Theta^k} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \left| \Theta_i \right|^{(v-p-1)/2} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta_i \right) \right) \, \text{d}\Theta_i
\]

\[= \frac{\prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2} \int_{\Theta^k} \left| \Theta_i \right|^{-(v)(v-p-1)/2} \, \text{d}\Theta_i}{\prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2}} \times \int_{\Theta^k} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{h-1} \sum_{i=1}^{m} \Theta_i \right) \right) \, \text{d}\Theta^h
\]

(A4)

Note that the function inside the integral in Equation (A4) happens to be proportional to the density function of an inverse Wishart distribution with degrees of freedom \((mv-p-1)\) and scale matrix \(\sum_{i=1}^{m} \Theta_i\). Thus, the integral in Equation (A4) is actually a constant. Then, Equation (A4) can be simplified to

\[
P \left( \{ \Theta_i \}_{i=1,2,\ldots,m} \right) \propto \frac{\prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2} \int_{\Theta^k} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{h-1} \sum_{i=1}^{m} \Theta_i \right) \right) \, \text{d}\Theta_i}{\prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2}}.
\]

(A5)

Inserting Equations (A5) and (A3) into Equation (A1):

\[
P \left( \Theta^k \mid \{ \Theta_i \}_{i=1,2,\ldots,m} \right) \propto \frac{\prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2} \int_{\Theta^k} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{h-1} \Theta_i \right) \right) \, \text{d}\Theta_i}{\prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2}}
\]

\[= \frac{\prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2} \int_{\Theta^k} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{h-1} \Theta_i \right) \right) \, \text{d}\Theta_i}{\prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2}} \times \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{h-1} \sum_{i=1}^{m} \Theta_i \right) \right).
\]

(A6)

which happens to be proportional to the density function of an inverse Wishart distribution with degrees of freedom \((mv-p-1)\) and scale matrix \(\sum_{i=1}^{m} \Theta_i\).

Proof of Proposition 2. Inserting Equation (9) into Equation (10) results in

\[
Q \left( \{ \Theta_i \}_{i=1,2,\ldots,m} \mid \{ \Theta_i \}_{i=1,2,\ldots,m} \right)
\]

\[= \int_{\Theta^k} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{h-1} \Theta_i \right) \right) \, \text{d}\Theta_i
\]

\[-\frac{1}{2} \text{tr} (S, \Theta) - \lambda_i || \Theta_i ||_1 \]

\[= \frac{1}{2} \text{tr} (S, \Theta) - \lambda_i || \Theta_i ||_1 \]

\[\sum_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2} \int_{\Theta^k} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{h-1} \sum_{i=1}^{m} \Theta_i \right) \right) \, \text{d}\Theta_i
\]

(A7)

It can be seen from Equation (A7) that the key to obtaining \(Q(\Theta_i; \Theta_{i-1}, \ldots, \Theta_{i-m})\) is to find the integral:

\[
\int_{\Theta^k} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2} \exp \left( -\frac{1}{2} \text{tr} \left( \Theta^{h-1} \sum_{i=1}^{m} \Theta_i \right) \right) \, \text{d}\Theta_i
\]

It becomes

\[
\frac{1}{2} \text{tr} (D, \Theta)
\]

(A8)

where

\[
D = \int_{\Theta^k} \Theta^{h-1} \Theta^k \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p)/2} \prod_{i=1}^{m} \left| \Theta_i \right|^{(v-p-1)/2} \exp \left( -\text{tr} \left( \Theta^{h-1} \sum_{i=1}^{m} \Theta_i \right) \right) \, \text{d}\Theta_i
\]

Now we need to derive \(D\). Specifically, according to Proposition 1:

\[
P \left( \Theta^k \mid \{ \Theta_i \}_{i=1,2,\ldots,m} \right)
\]

\[= \frac{\prod_{i=1}^{m} \Theta_i^{(v-p)/2} \prod_{i=1}^{m} \Theta_i^{(v-p-1)/2} \exp \left( -\text{tr} \left( \Theta^{h-1} \sum_{i=1}^{m} \Theta_i \right) \right) }{2^{(mv-p-1)/2} \Gamma_p (mv-p-1)/2} \]

where \(\Gamma_p\) is the Gamma function.

Also,

\[d\Theta^h = |\Theta^h|^{p+1} \, d\Theta^{h-1}.
\]

Therefore, \(D\) becomes

\[
D = \int_{\Theta^h} \Theta^{h-1} \prod_{i=1}^{m} \Theta_i^{(v-p)/2} \prod_{i=1}^{m} \Theta_i^{(v-p-1)/2} \exp \left( -\text{tr} \left( \Theta^{h-1} \sum_{i=1}^{m} \Theta_i \right) \right) \, d\Theta^{h-1},
\]

(A9)
which happens to be the mean of a Wishart distribution for \( \Theta_i \) with degrees of freedom equal to \( mv - p + 1 \) and scale matrix \( (\sum_{i=1}^{m} \Theta_i)^{-1} \); that is,

\[
D = (mv - p - 1) \left( \sum_{i=1}^{m} \Theta_i \right)^{-1}.
\] (A10)

Furthermore, inserting Equation (A10) into Equation (A8) and then into Equation (A7):

\[
Q \left( \{\Theta_i\}_{i=1,2,...,m} \mid \{\Theta_i\}_{i=1,2,...,m} \right)
= \sum_{i=1}^{m} \left\{ \frac{v - p - 1 + n_i}{2} \log |\Theta_i| - \frac{n_i}{2} \text{tr} (S_{i} \Theta_i) - \lambda_i ||\Theta_i||_1 - \frac{1}{2} \text{tr} \left( (mv - p - 1) \left( \sum_{i=1}^{m} \Theta_i \right)^{-1} \Theta_i \right) \right\}.
\]

Next, we want to update \( \Theta_j \) and \( \theta_{jj} \) while holding other elements in \( \Theta \) constant. To do this, let \( f \) represent the objective function in Equation (A7); i.e., \( f = \log |\Theta| - \text{tr}(S \Theta) - \lambda ||\Theta||_1 \); take the partial derivatives of \( f \) with respect to \( \Theta_j \) and \( \theta_{jj} \), respectively; and then make the partial derivatives to be zero; that is,

\[
\frac{\partial f}{\partial \Theta_j} = -\frac{2}{\theta_{jj} - \Theta_j \Theta_j^{-1} \Theta_j^{\top}} = -S_j - \lambda \text{SGN} (\Theta_j) = 0,
\] (A12)

and

\[
\frac{\partial f}{\partial \theta_{jj}} = \frac{1}{\theta_{jj} - \Theta_j \Theta_j^{-1} \Theta_j^{\top}} = -s_{jj} - \lambda = 0,
\] (A13)

where \( \text{SGN}(\Theta_j) \) denotes the partial derivative of \( ||\Theta||_1 \) with respect to \( \Theta_j \). It is difficult to solve for \( \Theta_j \) and \( \theta_{jj} \) from Equation (A12) and Equation (A13) directly. Therefore, we adopt the following strategies.

Let

\[
a = -\frac{\Theta_j}{\theta_{jj} - \Theta_j \Theta_j^{-1} \Theta_j^{\top}}.
\]

Then Equations (A12) and (A13) become

\[
2\Theta_j \Theta_j^{-1} a - S_j + \lambda \text{SGN}(a) = 0 \quad \text{A14}
\]

\[
a = -(s_{jj} + \lambda) \Theta_j. \quad \text{A15}
\]

It is clear that Equation (A14) is also the result of making the partial derivative of \( g \) with respect to \( a \) be zero in the following optimization problem:

\[
\min_a g = a^{\top} \Theta_j \Theta_j^{-1} a - S_j^{\top} a + \lambda' ||a||_1,
\] (A16)

which is equivalent to the following min-max problem:

\[
\max_{\kappa} \min_a g = 2 \left( -\frac{1}{2} \kappa^{\top} \Theta_j \Theta_j^{-1} \kappa + \kappa^{\top} a \right) - S_j^{\top} a + \lambda' ||a||_1.
\] (A17)

This min–max problem can be solved by the prox method proposed by Nemirovski (2005).

After \( a \) and \( \kappa \) are obtained, Equation (A15) can be used to find \( \Theta_j \); i.e., \( \Theta_j = -(a/(s_{jj} + \lambda)) \). Furthermore, based on Equation (A13), \( \theta_{jj} \) can be obtained; i.e., \( \theta_{jj} = ((-a^{\top} \kappa + 1)/s_{jj} + \lambda) \).

**Proof of convergence.** According to Tseng (2001), the BCD algorithm converges if and only if Equation (A16) has a unique solution of \( a \) at each iteration. The unique solution is guaranteed if the optimization problem in Equation (A16) is strictly convex. The strict convexity is true if \( \Theta_j \Theta_j^{-1} \) is positive definite, denoted by \( \Theta_j \Theta_j^{-1} > 0, \Theta_j \Theta_j^{-1} > 0 \) if \( \Theta > 0 \). Therefore, the key to prove the convergence of the BCD algorithm is to prove \( \Theta > 0 \). Recall that the BCD algorithm works by iterations and Equation (A16) needs to be solved at each iteration. As a result, we need to prove \( \Theta > 0 \) at each iteration. To achieve this, we use mathematical induction, which includes a basis step and an inductive step:
Let \( j \Theta \) be the \( \Theta \) obtained at the \( j \)th iteration of the BCD algorithm.

**Basis step:** Because \( 0 \Theta \) can be chosen by the user, \( 0 \Theta \) is guaranteed to satisfy \( 0 \Theta > 0 \).

**Inductive step:** Assuming \( j^{-1} \Theta > 0 \), we need to proving \( j \Theta > 0 \), which is equivalent to proving \( |j \Theta| > 0 \). Because

\[
|j \Theta| = |j^{-1} \Theta_{jj} (j_{jj} - j \Theta_j^T j^{-1} \Theta_{jj}^{-1} j \Theta_j)|,
\]

we need to prove \( |j^{-1} \Theta_{jj}| > 0 \) and \( (j_{jj} - j \Theta_j^T j^{-1} \Theta_{jj}^{-1} j \Theta_j) > 0 \).

\(|j^{-1} \Theta_{jj}| > 0 \) is true since we have assumed \( j^{-1} \Theta > 0 \). Based on Equation (A10),

\[
j_{jj} - j \Theta_j^T j^{-1} \Theta_{jj}^{-1} j \Theta_j = 1/(s_{jj} + \lambda) > 0.
\]

As a result, \( |j \Theta| > 0 \) and \( j \Theta > 0 \).

---

**Biographies**

Shuai Huang is currently a Ph.D. candidate in Industrial Engineering at Arizona State University. He received his B.S. in Statistics from the University of Science and Technology of China in 2007. His research interests include data mining and machine learning with applications in health care and manufacturing. He is a member of IIE, INFORMS, and ASQ.

Jing Li is an Assistant Professor in Industrial Engineering at Arizona State University. She received her B.S. from Tsinghua University in China and an M.A. in Statistics and a Ph.D. in Industrial and Operations Engineering from the University of Michigan in 2005 and 2007, respectively. Her research interests are applied statistics, data mining, and quality and systems engineering, with applications in biomedical informatics and manufacturing. She is a member of IIE, INFORMS, and ASQ.

Kewei Chen is the senior scientist and the head of the computational image analysis program at Banner Alzheimer’s Institute and an adjunct professor of Beijing Normal University and Shanghai Jiaotong University. He received his Ph.D. from UCLA. His main areas of interests are neuroimaging techniques for the study of human brain functions and diseases, especially Alzheimer’s disease.

Teresa Wu is an Associate Professor of Industrial Engineering at the School of Computing, Informatics, Decision Systems Engineering, Arizona State University and an Associate Professor of Radiology at the Mayo Clinic School of Medicine. She received her Ph.D. from the University of Iowa. Her main areas of interests are in health informatics and distributed decision support.

Jieping Ye is an Associate Professor of Computer Science and Engineering at Arizona State University. He received his Ph.D. in Computer Science from the University of Minnesota, Twin Cities in 2005. His research interests include machine learning, data mining, and biomedical informatics.

Xia Wu is an Associate Professor at the College of Information Science and Technology of Beijing Normal University (BNU), China. She received her Ph.D. from BNU. Her main areas of interests are intelligent signal processing, especially neuroimaging data processing.

Li Yao is a Professor and the Vice Dean of the College of Information Science and Technology of Beijing Normal University, China. She received her Ph.D. from the Chinese Academy of Science. Her main area of interest is brain signal processing.