Published in final edited form as:

IIE Trans. 2012 November 1; 44(11): 915–931. doi:10.1080/0740817X.2011.649390.

A Transfer Learning Approach for Network Modeling

Shuai Huang¹, Jing Li¹, Kewei Chen³, Teresa Wu¹, Jieping Ye², Xia Wu⁴, and Li Yao^{4,5}

¹Industrial Engineering, School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ

²Computer Science, School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ

³Banner Alzheimer's Institute, Phoenix, AZ

⁴School of Information Science and Technology, Beijing Normal University, Beijing, China

⁵State Key Laboratory of Cognitive Neuroscience and Learning, Beijing Normal University, Beijing, China

Abstract

Networks models have been widely used in many domains to characterize the interacting relationship between physical entities. A typical problem faced is to identify the networks of 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. In this paper, we propose a transfer learning approach, which adopts a Bayesian hierarchical model framework to characterize task relatedness and additionally uses the 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, which 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 identification of brain connectivity networks of Alzheimer's disease (AD) from functional magnetic resonance image (fMRI) data. The findings are consistent with the AD literature.

1. Introduction

Network models have been extensively used in many domains to characterize the interacting relationship 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). With the advancement of sensing technologies, network models can be learned from the rich amounts of sensing data, such as gene micro-arrays, brain images, and production data for the aforementioned networks, respectively. Note that the network models focused on in this paper are also called graphical models. Learning graphical models from data has been a popular research area in statistics and machine learning.

Existing research in graphical models focuses on learning a network/graphical model for a single task. However, many real-world problems involve learning of 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 paper. 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 multi-time longitudinal studies. For example, it may be of interest to learn brain connectivity networks for several longitudinal time points of an AD patient to track the disease progression. Despite difference, 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 learn 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 longitudinal time points of 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 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 recognize pears; learning to play an electronic organ may help learn a piano. Transfer learning in statistics and machine learning has focused on predictive models such as regressions and neural networks (Bakker and Heskes, 2003; Baxter, 2000; Caruana, 1997; Lawrence and Platt, 2004; Zhang et al., 2006). One major difference of these existing works is that they characterize the relatedness of the tasks in different ways. Thrun and O’Sullivan (1996) proposed use of a distance metric to evaluate the relatedness between tasks. In the setting of neural networks, task relatedness was 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 which account for the relatedness of 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; Yu et al. 2005; Xue et al., 2007). Despite the popularity of transfer learning in predictive models, limited work has been done on transfer learning of graphical/network models. This paper intends to bridge this gap.

In this paper, we propose a transfer learning approach for network modeling of multiple related tasks. We focus on one particular type of network model called Gaussian Graphical Model (GGM). A GGM consists of nodes that are random variables following a multivariate normal distribution and undirected arcs that indicate non-zero partial correlations between variables. Various methods have been developed for learning a GGM from data, which are also known as methods for inverse covariance (IC) estimation, because the undirected arcs in a GGM correspond to nonzero entries in the IC matrix of the data. These methods are reviewed as follows. Note that transfer learning is not considered in these existing methods.

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 Bühlmann (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). Further, efficient algorithms were proposed by Friedman et al. (2007) and Levina et al. (2008) to implement the penalized maximum likelihood approach by Yuan and Lin (2007), which are applicable to high-dimensional problems. Some other methods were proposed, such as a method based on threshold gradient descent 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 dealing with the situations when variables have a natural ordering (Bickel and Levina, 2008; Levina et al, 2008).

Different from these existing methods, the approach we propose enables transfer learning in learning of GGMs for multiple related tasks. Specifically, we adopt the Bayesian hierarchical modeling (BHM) framework in our problem formulation to characterize the relatedness of tasks. We further add L_1 -regularization to our problem formulation. L_1 -regularization has been well-known to be able to discourage truly zero parameters or small-valued parameters from showing up in the learned 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 the Maximum Likelihood Estimation (MLE), may generate unreliable estimates. L_1 -regularization has been adopted in regressions (Tibshirani, 1996) and graphical models without transfer learning (Friedman et al., 2007), and have demonstrated 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 learn GGMs for multiple related tasks with transfer learning enabled. Furthermore, we conduct simulation studies to compare performance of the proposed transfer learning approach with single task learning that learns GGMs 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. 15 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 (NCs).

2. Introduction to GGM

A GGM consists of nodes, $\mathbf{X} = \{X_1, \dots, X_p\}$, and undirected arcs. The nodes are random variables following a multivariate normal distribution, i.e., $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Let $\boldsymbol{\Theta}$ be the inverse covariance (IC) matrix of the distribution, i.e., $\boldsymbol{\Theta} = \boldsymbol{\Sigma}^{-1}$. 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 $\boldsymbol{\Theta}$ is nonzero. Please see Fig. 1 for an example of a GGM and the corresponding IC matrix.

Given data on the nodes, the GGM can be learned by estimating the IC matrix. It is straightforward to derive the log-likelihood of $\boldsymbol{\Theta}$, which is $\log |\boldsymbol{\Theta}| - \text{tr}(\mathbf{S}\boldsymbol{\Theta})$, where $|\cdot|$ and $\text{tr}(\cdot)$ denote the determinant and trace of a matrix, respectively, and $\mathbf{S}$ is the sample covariance matrix. By maximizing this log-likelihood, we can obtain the MLE for $\boldsymbol{\Theta}$, which is $\boldsymbol{\Theta}^{\text{MLE}} = \mathbf{S}^{-1}$. However, for large-scale GGMs with limited sample sizes, the MLE may be quite unreliable in the sense that many zero entries in $\boldsymbol{\Theta}$ may be nonzero in $\boldsymbol{\Theta}^{\text{MLE}}$, 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, $\mathbf{S}$ is not invertible. To tackle this deficiency of the MLE, a well-known strategy is to maximize the L_1 -regularized log-likelihood function, i.e.,

$$\hat{\Theta} = \arg\max_{\Theta \succ 0} \log|\Theta| - \text{tr}(\mathbf{S}\Theta) - \lambda \|\Theta\|_1, \quad (1)$$

where $\|\Theta\|_1$ is the so-called L_1 -norm of Θ , i.e., the sum of the absolute values of the entries in Θ . λ is the regularization parameter; the larger the λ , the more zero entries the $\hat{\Theta}$ will have, i.e., the Θ will be sparser. λ can be specified by the user or cross-validation. (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, Θ_i , the graphical LASSO in (1) may be used, i.e.,

$$\hat{\Theta}_i = \arg\max_{\Theta_i \succ 0} \log|\Theta_i| - \text{tr}(\mathbf{S}_i \Theta_i) - \lambda_i \|\Theta_i\|_1, \quad (2)$$

where $\mathbf{S}_i$ is the sample covariance matrix for task i , $i = 1, \dots, m$. This method treats the tasks as independent and does not exploit their relatedness. Alternatively, we propose to consider the relatedness by assuming that the Θ_i 's are "samples" drawn from the same probability distribution, i.e., we adopt the BHM framework to characterize the task relatedness (Fig. 2). This same probability distribution is chosen to be a Wishart distribution, i.e., $\Theta_i \sim \text{Wishart}_v(\Theta^h)$, where Θ^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 Θ_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 has been 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 is nicely interpretable in our problem, as will be shown in Section 4.3. (iv) The scale matrix, Θ^h , of the Wishart distribution depicts how the tasks are related. Specifically, because

$$E(\Theta_i) = v\Theta^h, \quad (3)$$

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

To enable transfer learning, the Θ_i 's should be estimated by utilizing not only the data, i.e., the $\mathbf{S}_i$'s, but also the parameters of the common Wishart distribution, Θ^h and v . The challenge here is that Θ^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 Θ^h because Θ^h is a high-dimensional matrix. Therefore, we propose to integrate out Θ^h during the estimation of the Θ_i 's, i.e., we aim to find an estimate, $\hat{\Theta}_i$, for each Θ_i , that maximizes the logarithm of the posterior probability of Θ_i 's given the data, marginalizing over Θ^h . This can be written as:

$$\{\hat{\Theta}_i\}_{i=1,2,\dots,m} = \arg\max_{\{\Theta_i\}_{i=1,2,\dots,m}} \int_{\Theta^h} \log P(\{\Theta_i\}_{i=1,2,\dots,m}, \Theta^h | \{\mathbf{S}_i\}_{i=1,2,\dots,m}) d\Theta^h. \quad (4)$$

Equivalently, we can maximize the logarithm of the joint probability which is proportional to the posterior probability in (4), i.e.,

$$\{\hat{\Theta}_i\}_{i=1,2,\dots,m} = \operatorname{argmax}_{\{\Theta_i\}_{i=1,2,\dots,m}} \int_{\Theta^h} \log P(\{\Theta_i\}_{i=1,2,\dots,m}, \Theta^h, \{\mathbf{S}_i\}_{i=1,2,\dots,m}) d\Theta^h. \quad (5)$$

In (5), the $P(\{\Theta_i\}_{i=1,2,\dots,m}, \Theta^h, \{\mathbf{S}_i\}_{i=1,2,\dots,m})$ can be written as $P(\{\Theta_i\}_{i=1,2,\dots,m}, \Theta^h, \{\mathbf{S}_i\}_{i=1,2,\dots,m}) = P(\{\mathbf{S}_i\}_{i=1,2,\dots,m} | \{\Theta_i\}_{i=1,2,\dots,m}, \Theta^h) P(\{\Theta_i\}_{i=1,2,\dots,m} | \Theta^h) P(\Theta^h)$.

According to the structure in Fig. 2, the Θ_i 's are conditionally independent given Θ^h , so $P(\{\Theta_i\}_{i=1,2,\dots,m} | \Theta^h) = \prod_{i=1}^m P(\Theta_i | \Theta^h)$. Each $\mathbf{S}_i$ is independent of Θ^h given Θ_i , so $P(\{\mathbf{S}_i\}_{i=1,2,\dots,m} | \{\Theta_i\}_{i=1,2,\dots,m}, \Theta^h) = \prod_{i=1}^m P(\mathbf{S}_i | \Theta_i)$. Therefore, (5) can be further written as:

$$\{\hat{\Theta}_i\}_{i=1,2,\dots,m} = \operatorname{argmax}_{\{\Theta_i\}_{i=1,2,\dots,m}} \int_{\Theta^h} \log(P(\Theta^h) \prod_{i=1}^m P(\Theta_i | \Theta^h) \prod_{i=1}^m P(\mathbf{S}_i | \Theta_i)) d\Theta^h, \quad (6)$$

where $P(\Theta_i | \Theta^h) \propto |\Theta^h|^{-v/2} |\Theta_i|^{(v-p-1)/2} e^{-tr(\Theta^{h-1} \Theta_i)/2}$ and $P(\mathbf{S}_i | \Theta_i) \propto |\Theta_i|^{n_i/2} e^{-n_i tr(\mathbf{S}_i \Theta_i)/2}$. Then, (6) becomes:

$$\{\hat{\Theta}_i\}_{i=1,2,\dots,m} = \operatorname{argmax}_{\{\Theta_i\}_{i=1,2,\dots,m}} \int_{\Theta^h} \sum_{i=1}^m \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} tr(\Theta^{h-1} \Theta_i) - \frac{n_i}{2} tr(\mathbf{S}_i \Theta_i) \right\} d\Theta^h \quad (7)$$

Moreover, due to the same consideration as graphical LASSO, we add L_1 -regularization to (7), which gives the final objective function, i.e.,

$$\{\hat{\Theta}_i\}_{i=1,2,\dots,m} = \operatorname{argmax}_{\{\Theta_i\}_{i=1,2,\dots,m}} \int_{\Theta^h} \sum_{i=1}^m \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} tr(\Theta^{h-1} \Theta_i) - \frac{n_i}{2} tr(\mathbf{S}_i \Theta_i) - \lambda_i \|\Theta_i\|_1 \right\} d\Theta^h, \quad (8)$$

4. Problem Solving by an EM Algorithm

It is difficult to solve (8) directly, since it involves an integral over a high-dimensional matrix Θ^h . We propose a method based on the EM algorithm. Adopting the EM algorithm is a natural choice because we want to integrate out Θ_h in (8), i.e., treat Θ_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 general EM Algorithm

Consider a probabilistic model that depends on some observed variables, $\mathbf{Y}$, and some unobserved latent variables, $\mathbf{Z}$. Let Ω denote the parameters of the model that need to be estimated. One way of estimation is to find estimates for Ω that maximize the posterior probability of Ω given $\mathbf{Y}$ and $\mathbf{Z}$. However, as $\mathbf{Z}$ is latent, the estimation cannot be done directly, but 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 Ω , $\mathbf{Z}$, and $\mathbf{Y}$ (which is proportional to the posterior probability of Ω given $\mathbf{Y}$ and $\mathbf{Z}$), with respect to the conditional distribution of $\mathbf{Z}$ given $\mathbf{Y}$ under the current (i.e., the t -th iteration) estimates for the parameters Ω^t . Denote this expectation by $Q(\Omega | \Omega^t)$, i.e.,

$$\text{Estep: } Q(\Omega | \Omega^t) = E_{\mathbf{Z} | \mathbf{Y}, \Omega^t} \{L(\Omega, \mathbf{Z}, \mathbf{Y})\} = \int_{\mathbf{Z}} L(\Omega, \mathbf{Z}, \mathbf{Y}) P(\mathbf{Z} | \mathbf{Y}, \Omega^t) d\mathbf{Z}$$

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

$$\text{M step: } \Omega^{t+1} = \underset{\Omega}{\operatorname{argmax}} Q(\Omega|\Omega^t)$$

These parameters Ω^{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 EM framework

To fit the transfer learning formulation in (8) into the EM framework, we consider $\{\Theta_i\}_{i=1, 2, \dots, m}$, Θ^h , and $\{\mathbf{S}_i\}_{i=1, 2, \dots, m}$ to be the parameters to be estimated, the latent and the observed variables (i.e., Ω , $\mathbf{Z}$, and $\mathbf{Y}$), respectively. Then, the E step is to calculate

$$\int_{\Theta^h} L(\{\Theta_i\}_{i=1, 2, \dots, m}, \Theta^h, \{\mathbf{S}_i\}_{i=1, 2, \dots, m}) P(\Theta^h | \{\mathbf{S}_i\}_{i=1, 2, \dots, m}, \{\Theta_i^t\}_{i=1, 2, \dots, m}) d\Theta^h.$$

According to Fig. 2, Θ^h is independent of $\{\mathbf{S}_i\}_{i=1, 2, \dots, m}$ given $\{\Theta_i^t\}_{i=1, 2, \dots, m}$, i.e.,

$$P(\Theta^h | \{\mathbf{S}_i\}_{i=1, 2, \dots, m}, \{\Theta_i^t\}_{i=1, 2, \dots, m}) = P(\Theta^h | \{\Theta_i^t\}_{i=1, 2, \dots, m}).$$
 Also, according to (8),

$$L(\{\Theta_i\}_{i=1, 2, \dots, m}, \Theta^h, \{\mathbf{S}_i\}_{i=1, 2, \dots, m}) = \sum_{i=1}^m \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} \operatorname{tr}(\Theta^{h-1} \Theta_i) - \frac{n_i}{2} \operatorname{tr}(\mathbf{S}_i \Theta_i) - \lambda_i \|\Theta_i\|_1 \right\}. \quad (9)$$

Then, the E and M steps are:

$$\text{E step: } Q(\{\Theta_i\}_{i=1, 2, \dots, m} | \{\Theta_i^t\}_{i=1, 2, \dots, m}) = \int_{\Theta^h} L(\{\Theta_i\}_{i=1, 2, \dots, m}, \Theta^h, \{\mathbf{S}_i\}_{i=1, 2, \dots, m}) P(\Theta^h | \{\Theta_i^t\}_{i=1, 2, \dots, m}) d\Theta^h \quad (10)$$

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

$$\text{M step: } \{\Theta_i^{t+1}\}_{i=1, 2, \dots, m} = \underset{\{\Theta_i\}_{i=1, 2, \dots, m}}{\operatorname{argmax}} Q(\{\Theta_i\}_{i=1, 2, \dots, m} | \{\Theta_i^t\}_{i=1, 2, \dots, m})$$

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

Conducting the E step—We decompose the E step into two sub-steps: finding the parametric form of the distribution $P(\Theta^h | \{\Theta_i^t\}_{i=1, 2, \dots, m})$, and then transforming the $Q(\{\Theta_i\}_{i=1, 2, \dots, m} | \{\Theta_i^t\}_{i=1, 2, \dots, m})$ 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 $\Theta^h | \{\Theta_i^t\}_{i=1, 2, \dots, m}$ is an Inverse-Wishart distribution with scale matrix $\sum_{i=1}^m \Theta_i^t$ and degrees of freedom $mv - p - 1$.

Proposition 2: The $Q(\{\Theta_i\}_{i=1, 2, \dots, m} | \{\Theta_i^t\}_{i=1, 2, \dots, m})$ can be decomposed into a sum of m terms, with the i^{th} term involving only Θ_i not Θ_j ($j \neq i$), i.e.,

$$(\{\Theta_i\}_{i=1,2,\dots,m}|\{\Theta_i^t\}_{i=1,2,\dots,m})$$

$$= \sum_{i=1}^m \left\{ \frac{v-p-1+n_i}{2} \log|\Theta_i| - \frac{n_i}{2} \text{tr}(\mathbf{S}_i \Theta_i) - \lambda_i \|\Theta_i\|_1 - \frac{1}{2} \text{tr} \left((mv-p-1) \left(\sum_{i=1}^m \Theta_i^t \right)^{-1} \Theta_i \right) \right\}$$

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, i.e.,

Corollary 2.1: To find the $\{\Theta_i^{t+1}\}_{i=1,2,\dots,m}$ can be achieved by solving m optimization problems, i.e.,

$$\Theta_i^{t+1} = \underset{\Theta_i}{\text{argmax}} \left\{ \frac{v-p-1+n_i}{2} \log|\Theta_i| - \frac{n_i}{2} \text{tr}(\mathbf{S}_i \Theta_i) - \lambda_i \|\Theta_i\|_1 - \frac{1}{2} \text{tr} \left((mv-p-1) \left(\sum_{i=1}^m \Theta_i^t \right)^{-1} \Theta_i \right) \right\}, \quad (11)$$

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

$$\Theta_i^{t+1} = \underset{\Theta_i}{\text{argmax}} \{ \log|\Theta_i| - \text{tr}(\mathbf{S}_i^t \Theta_i) - \lambda'_i \|\Theta_i\|_1 \}, \quad (12)$$

where $\lambda'_i = \frac{2}{v-p-1+n_i} \lambda_i$, and

$$\mathbf{S}_i^t = \frac{n_i \mathbf{S}_i + (mv-p-1) \left(\sum_{k=1}^m \Theta_k^t \right)^{-1}}{v-p-1+n_i}. \quad (13)$$

(12) has a similar form to the graphical LASSO in (2). The difference is that the graphical LASSO has $\mathbf{S}_i$ instead of $\mathbf{S}_i^t$, so it does not enable transfer learning. To solve (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 GGM learning 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 (3), i.e., $\mathbf{E}(\Theta_k) = v\Theta^h$, the $\sum_{k=1}^m \Theta_k^t$ in (13) can be considered as the estimate for $mv\Theta^h$ obtained at iteration t of the EM algorithm, i.e.,

$$\sum_{k=1}^m \Theta_k^t = mv\Theta^{h,t}. \quad (14)$$

Inserting (14) into (13),

$$\mathbf{S}_i^t = \frac{n_i \mathbf{S}_i + \frac{mv-p-1}{mv} (\boldsymbol{\Theta}^{h,t})^{-1}}{v-p-1+n_i}. \quad (15)$$

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

An interesting observation on (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 $\boldsymbol{\Theta}^{h,t}$ will be utilized in obtaining $\mathbf{S}_i^t$ 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 learning of task i . However, the value of v can be hardly known as *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, (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., smaller n_i . Furthermore, when more tasks are learned together (bigger m), $\boldsymbol{\Theta}^{h,t}$ will be weighted heavier relative to $\mathbf{S}_i$. This makes sense because more tasks will help obtain a more reliable estimate for $\boldsymbol{\Theta}^{h,t}$. In addition, when the dimensionality of each task is higher (bigger p), $\boldsymbol{\Theta}^{h,t}$ will be weighted less. This also makes sense because $\boldsymbol{\Theta}^{h,t} = \sum_{k=1}^m \boldsymbol{\Theta}_k^t / mv$ and $\boldsymbol{\Theta}_k^t$'s are learned from data; in general, statistical learning suffers more as the dimensionality of problems increases. By weighting $\boldsymbol{\Theta}^{h,t}$ less, the proposed transfer learning approach provides a robust mechanism to tackle high dimensionality.

4.4 Selection of parameters v and λ_i

To apply the proposed algorithm in Fig. 3 to solve (8), v and λ_i ($i = 1, \dots, m$) 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 for learning an IC matrix for each task by applying the proposed algorithm at a given v and λ . Here, we assume the same λ for all the tasks for simplicity; making the λ_i 's different is just a straightforward extension. Next, based on the learned 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 the tasks are summed together and we obtain an “overall” likelihood for the given v and λ . This process is repeated V times for the same v and λ , and each time a different subsample is used for computing the overall likelihood. Then, the average overall likelihood is computed over the V repetitions. This whole procedure is performed on different combinations of values for v and λ (i.e., a grid search on v and λ). The final v and λ selected are the ones maximizing the average overall likelihood. Note that this proposed cross-validation procedure is similar to the 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:

- i. *Construct the common prior mean matrix, Θ_h .* Because $E(\Theta_i) = v\Theta^h$, we first need to construct Θ^h . Specifically, the initial value for Θ^h , $\tilde{\Theta}^h$, is generated by

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

where $\tilde{\tau}_{jk}^h$ denotes the entry at the j -th row and k -th column of $\tilde{\Theta}^h$; $j \leftrightarrow k$ means that there is an arc between nodes j and k , and $j \nleftrightarrow k$ means otherwise; $D = [-1, -0.5] \cup [0.5, 1]$. $\tilde{\Theta}^h$ is then rescaled to ensure the positive definiteness. 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 fold of 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 Θ^h . Furthermore, let $\Theta^h = \Theta^h/v$

- ii. *Construct the true IC matrices of m related tasks, $\Theta_i, i = 1, \dots, m$.* To generate a Θ_i , the following substeps are performed: (ii.1) Randomly modify $(100 - s)\%$ of the nonzero entries in Θ^h to be zero. (ii.2) Randomly modify the same number of zero entries in Θ^h to be nonzero. This is to ensure that each Θ_i has the same sparsity as Θ^h . Each of these nonzero entries is sampled from $\text{Uniform}(D)$. (ii.3) For the remaining unmodified nonzero entries in Θ^h , resample their values from $\text{Uniform}(D)$ and add the resulting value of each nonzero entry with the entry in Θ^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, Θ_i , is related to Θ^h . The higher the $s\%$, the more the relatedness. Considering the extreme case when $s\% = 100\%$, the Θ_i constructed following substeps (ii.1)–(ii.3) will have the same positions of nonzero entries as Θ^h , so the GGM corresponding to Θ_i will look the same as that corresponding to Θ^h . When $s\% = 0$, the Θ_i will have completely different positions of nonzero entries from Θ^h , so their GGMs will not share even a single arc. Through relating to Θ^h , the IC matrices of all the tasks are related, so $s\%$ may be considered as an indirect measure of task relatedness. It makes sense to specify the relatedness of Θ_i to Θ^h , rather than specifying the relatedness between the Θ_i 's directly, because transfer learning between tasks is through Θ^h , the common prior shared by all the tasks.

- iii. *Generate simulation data for each task.* A dataset consisting of n independently and identically distributed (IID) observations is generated from a multivariate normal distribution with mean zero and IC matrix Θ_i for task $i, i = 1, \dots, m$.
- iv. *Estimate the IC matrix of each task.* We apply the transfer learning approach to the data and estimate the IC matrix of each task.
- v. *Measure the performance.* In order to apply the transfer learning approach, we need to choose values for λ and v . To assess the overall performance of transfer learning across different choices for λ and v , ROC curves are employed. An ROC curve plots the numbers of true positives vs. false positives over all possible choices for λ and v . Here, we count one true positive if a nonzero entry in the true IC matrix is

also nonzero in the estimated IC matrix; we count one false positive if a zero entry in the true IC matrix is nonzero in the estimated IC matrix. Thus, the number of true positives (Y axis of the ROC curve) and the number of false positives (X axis of the ROC curve) reflects the power of algorithm in detecting non-zero entries and the false alarm error, respectively.

Note that execution of steps (iii)–(iv) 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., (2). Following a similar procedure to steps (i)–(v), 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.

Figs. 5–7 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 task relatedness ($s\% = 90\%, 50\%, 0\%$). 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 Θ^h is set to be equal to p , such that the GGM in each task is sparse.

The following observations can be obtained:

- 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.
- The advantage of transfer learning over single task learning is more significant when there are more related tasks.
- 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.

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 3–6 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 is also used in single task learning. Table 2 shows the CPU time of 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 because transfer learning takes 3–6 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 only takes less than 10 minutes to complete with 200 variables and 10 tasks (Table 1).

6. Application in brain connectivity network modeling of AD

The simulation study in Section 5 has demonstrated effectiveness of the proposed transfer learning approach. In this section we explore the application to brain connectivity modeling of AD. AD is a fatal, neurodegenerative disorder currently affecting over five 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 alteration in the brain 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 apply the proposed transfer learning approach to the fMRI data of 15 AD patients, which are considered as 15 related tasks, and learn a brain connectivity network for each patient. To provide a contrast for the AD connectivity networks, the networks of 16 NCs, who are considered as another set of related tasks, are also learned by the proposed transfer learning approach.

The selection criteria for the 15 AD patients and 16 NCs are as follows. The AD patients are aged between 53~79, right-handed, free of other diseases such as stroke and focal pathology, and with MMSE scores between 0–20. MMSE is a clinical instrument for cognitive assessment; the lower the score the more severe the dementia. The NCs are selected to purposely match the AD patients in terms of other selection criteria except MMSE scores, i.e., they are in the same age cohort as the AD patients, right-handed, free of other diseases, but with MMSE scores between 27–30 (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 apply the following preprocessing steps to the data. First, we apply the Automated Anatomical Labeling (AAL) technique (Tzourio-Mazoyer, 2002) to segment the whole brain of each subject into 116 regions. 42 regions, whose names are given in Table 3, are selected as they have been considered to be potentially related to AD in the literature. These regions distribute 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 are averaged into one regional average time course. Then, the first five data points for each regional average time course are discarded. Finally, each regional average time course is detrended. After the preprocessing, the dataset of each subject takes 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 hundreds of 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 (Valdes-Sosa et al., 2005 and Worsley, et al., 1997). We apply the transfer learning approach to the datasets of the 15 AD patients, which produces 15 GGMs. Similarly, GGMs are learned for the 16 NCs. Note that in applying the transfer learning approach, the values for the regularization parameters, λ_i 's, need to be selected. In this paper, we focus on comparing AD and NCs in terms of the distribution/organization of the connectivity in the brain, which has been less studied in the literature, but not in terms of the global scale of the connectivity, which has been studied substantially. To achieve this, we must factor out the connectivity difference between AD and NCs that is due to their difference at the global scale, so that the remaining difference will reflect their difference in the connectivity distribution/organization. A common strategy is to control the total number of arcs in the AD and NCs connectivity networks to be the same, which has been adopted by a number of other studies (Supekar et al., 2008; Stam et al., 2007). We also adopt this

strategy; specifically, we adjust the λ_i 's in estimating the 15 AD connectivity networks and those in estimating the 16 NCs connectivity networks, such that all these networks have the same total number of arcs. Also, by selecting different values for the total number of arcs, we can 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 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 are treated as “samples” of the AD (NCs) brain connectivity network. Based on these samples, we generate boxplots for the numbers of arcs within and between lobes for AD and NCs in Fig. 8(a). Furthermore, statistical significance of the observed difference between AD and NCs in the boxplots is assessed by hypothesis testing. The P-values of the hypothesis testing are given in Table 4.

The following interesting findings are obtained:

- The parietal lobe of AD has significantly less connectivity than NCs. Loss of connectivity in the parietal lobe of AD has been reported by a number of other studies in the literature (Langbaum, et al., 2009; Chen, et al., 2010).
- The regions well-known to be attacked by AD the first and severely are “Hippocampus_L&R” in the temporal lobe, which play an important role in memory. We perform hypothesis testing on the difference between AD and NCs in terms of the number of arcs between “Hippocampus_L&R” and other regions in the temporal lobe, and find the P-values to be 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&R” and other regions in AD, which is consistent with quite a few other studies in the literature (Supekar et al., 2008; Wang et al., 2007).
- The frontal lobe of AD has significantly more connectivity than NCs. This is consistent with the previous literature and has been interpreted as compensatory reallocation or recruitment of cognitive resources (Gould et al., 2006; Stern, 2006). Because regions in the frontal lobe are typically less affected by AD, increase of connectivity in the frontal lobe may help preserve some cognitive functions in AD patients. Same explanation may be applied to the increase of connectivity between the frontal and temporal lobes in AD.
- All the above findings are consistent across different total numbers of arcs in the connectivity networks.

For comparison purposes, we also apply graphical LASSO to the same datasets. Please note that graphical LASSO is a single task learning approach which learns each task independently. Boxplots and P-value tables similar to those in Fig. 8(a) and Table 4 are generated, as shown in Fig. 8(b) and Table 5. It can be seen that the single task learning fails to identify any significant difference between AD and NCs. The reason, as revealed by the boxplots, is that there is large variability in the 15 (16) brain connectivity networks of AD (NCs). This is because the single task learning is not able to make use of task relatedness to compensate for sample size shortage. As a result, the learned connectivity networks may not be reliable.

7. Conclusion

This paper proposed a transfer learning approach for jointly learning 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 GGMs estimation. Under this problem formulation, a method based on the EM algorithm was further developed to learn GGMs from data. Simulation studies showed that the transfer learning approach performs better than single task learning. 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 the fMRI data of 15 AD patients and 16 NCs for brain connectivity network identification. Comparison between the connectivity networks of AD and NCs revealed that AD is associated with decreased connectivity in the parietal lobe and between 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 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 practical data analysis point of view, it is of interest to examine if the identified patterns will still be present given small perturbation to the data. Specifically, we may create some perturbed datasets out of the original dataset by resampling or adding noisy 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 of providing assurance for the robustness of the results against sampling variability. We will investigate these methodological and practical issues in future work.

Acknowledgments

This material is based in part on work supported by the National Science Foundation under Grant No. 1149602 and No. 1069246, and the National Institute of Health under Grant No. 1R01GM096194-01. Also, we would like to thank Yumei Zhang, MD, and Xiaoting Guan, MD, in Tiantan Hospital of Beijing, China, for their assistance with the data collection.

References

1. Bakker B, Heskes T. Task Clustering and Gating for Bayesian Multitask Learning. *Journal of Machine Learning Research*. 2003; 4:83–99.
2. Baxter J. A Model of Inductive Bias Learning. *Journal of Artificial Intelligence Research*. 2000
3. Bickel PJ, Levina E. Regularized Estimation of Large Covariance Matrices. *Annals of Statistics*. 2008; 36:199–227.
4. Caruana R. Multitask learning. *Machine Learning*. 1997; 28:41–75.
5. Chen K, Langbaum JBS, Fleisher AS, Ayutyanont N, Reschke C, Lee W, Liu X, Bandy D, Alexander GE, Thompson PM, Foster NL, Harvey DJ, de Leon MJ, Koeppe RA, Jagust WJ, Weimer MW, Reiman EM. the ADNI. Twelve-month metabolic declines in probable Alzheimer's disease and amnesic mild cognitive impairment assessed using an empirically pre-defined statistical region-of-interest: Findings from the Alzheimer's Disease Neuroimaging Initiative. *NeuroImage*. 2010; 51:645–664.
6. Dellaert, F. Technical Report. GVV Center; College of Computing; Georgia Tech; 2002. The Expectation Maximization Algorithm. GIT-GVV-02–20

7. Friedman N, Linial M, Nachman I, Pe'er D. Using Bayesian Networks to Analyze Expression Data. *Journal of Computational Biology*. 2000; 7:601–620. [PubMed: 11108481]
8. Friedman J, Hastie T, Tibshirani R. Sparse Inverse Covariance Estimation with the Graphical Lasso. *Biostatistics*. 2007; 8(1):1–10.
9. Gould RL, Arroyo B, Brown RG, Owen AM, Bullmore ET, Howard RJ. Brain Mechanisms of Successful Compensation during Learning in Alzheimer Disease. *Neurology*. 2006; 67:1011–1017. [PubMed: 17000970]
10. Huang S, Li J, et al. Learning Brain Connectivity of Alzheimer's Disease by Sparse Inverse Covariance Estimation. *NeuroImage*. 2010; 50:935–949. [PubMed: 20079441]
11. Lawrence, ND.; Platt, JC. Learning to Learn with the Informative Vector Machine. *Proceedings of the 21st International Conference on Machine Learning*; 2004.
12. Langbaum JBS, Chen K, Lee W, Reschke C, Bandy D, Fleisher AS, Alexander GE, Foster NL, Weiner MW, Koeppe RA, Jagust WJ, Reiman EM. the ADNI. Categorical and correlational analyses of baseline fluorodeoxyglucose positron emission tomography images from the Alzheimer's Disease Neuroimaging Initiative (ADNI). *NeuroImage*. 2009; 45:1107–1116. [PubMed: 19349228]
13. Levina E, Rothman AJ, Zhu J. Sparse Estimation of Large Covariance Matrices via a Nested Lasso Penalty. *Annals of Applied Statistics*. 2008; 2:245–263.
14. Li H, Gui J. Gradient Directed Regularization for Sparse Gaussian Concentration Graphs, with Applications to Inference of Genetic Networks. *Biostatistics*. 2006; 7:302–317. [PubMed: 16326758]
15. Li J, Shi J. Knowledge Discovery from Observational Data for Process Control through Causal Bayesian Networks. *IIE Transactions*. 2007; 39(6):681–690.
16. Meinshausen N, Bühlmann P. High Dimensional Graphs and Variable Selection with the Lasso. *Annals of Statistics*. 2006; 34:1436–1462.
17. Nemirovski A. Prox-method with Rate of Convergence $o(1/t)$ for Variational Inequalities with Lipschitz Continuous Monotone Operators and Smooth Convex-Concave Saddle Point Problems. *SIAM Journal on Optimization*. 2005; 15(1):229–251.
18. Peng J, Wang P, Zhou NF, Zhu J. Partial Correlation Estimation by Joint Sparse Regression Models. *Journal of the American Statistical Association*. 2009; 104(486):735–746. [PubMed: 19881892]
19. Ravikumar P, Raskutti G, Wainwright MJ, Yu B. Model selection in Gaussian graphical models: High-dimensional consistency of ℓ_1 -regularized MLE. *Advances in Neural Information Processing Systems (NIPS)*. 2008; 21
20. Schafer J, Strimmer K. A Shrinkage Approach to Large-Scale Covariance Matrix Estimation and Implications for Functional Genomics. *Statistical Applications in Genetics and Molecular Biology*. 2005; 4(1):Article 32.
21. Stam CJ, Jones BF, Nolte G, Breakspear M, Scheltens P. Small-World Networks and Functional Connectivity in Alzheimer's Disease. *Cerebral Cortex*. 2007; 17:92–99.
22. Stern Y. Cognitive Reserve and Alzheimer Disease. *Alzheimer Disease Associated Disorder*. 2006; 20:69–74.
23. Supekar K, Menon V, Rubin D, Musen M, Greicius MD. Network Analysis of Intrinsic Functional Brain Connectivity in Alzheimer's Disease. *PLoS Comput Biol*. 2008; 4(6):1–11.
24. Sun, L.; Patel, R.; Liu, J.; Chen, K.; Wu, T.; Li, J.; Reiman, E.; Ye, J. Mining Brain Region Connectivity for Alzheimer's Disease Study via Sparse Inverse Covariance Estimation. *Proceedings of Knowledge Discovery and Data Mining Conference (KDD)*; 2009.
25. Thrun, S.; O'Sullivan, J. Discovering structure in multiple learning tasks: The TC algorithm. *International Conference on Machine Learning*; 1996. p. 489–497.
26. Tibshirani R. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society Series B*. 1996; 58(1):267–288.
27. Tseng P. Convergence of Block Coordinate Descent Method for Nondifferentiable Maximization. *J Opt Theory and Applications*. 2001; 109(3):474–494.

28. Tzourio-Mazoyer N, et al. Automated Anatomical Labelling of Activations in SPM using a Macroscopic Anatomical Parcellation of the MNI MRI Single Subject Brain. *Neuroimage*. 2002; 15:273–289. [PubMed: 11771995]
29. Valdés-Sosa P, Sánchez-Bornot J, Lage-Castellanos A, Vega-Hernández M, Bosch-Bayard J, Melie-García L, Canales-Rodríguez E. Estimating brain functional connectivity with sparse multivariate autoregression. *Philos Trans Roy Soc B: Biological Sciences*. 2005; 360:969–981.
30. Wang K, Liang M, Wang L, Tian L, Zhang X, Li K, Jiang T. Altered Functional Connectivity in Early Alzheimer's Disease: A Resting-State fMRI Study. *Human Brain Mapping*. 2007; 28:967–978. [PubMed: 17133390]
31. Worsley KJ, Poline JB, Friston KJ, Evans AC. Characterizing the response of PET and fMRI data using multivariate linear models. *NeuroImage*. 1997; 6(4):305–319. [PubMed: 9417973]
32. Wu CFJ. On the Convergence Properties of the EM Algorithm. *The Annals of Statistics*. 1983; 11:95–103.
33. Xue Y, Liao X, Carin L, Krishnapuram B. Multi-task learning for classification with Dirichlet process priors. *Journal of Machine Learning Research*. 2007; 8:35–63.
34. Yuan M, Lin Y. Model Selection and Estimation in the Gaussian Graphical Model. *Biometrika*. 2007; 94 (1):19–35.
35. Zhang, J.; Ghahramani, Z.; Yang, Y. Learning Multiple Related Tasks using Latent Independent Component Analysis. In: Weiss, Y.; Schölkopf, B.; Platt, J., editors. *Advances in Neural Information Processing Systems*. Vol. 18. MIT Press; Cambridge, MA: 2006.

Biographies

Shuai Huang is currently a PhD candidate in Industrial Engineering in 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 in 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. She 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 the Alzheimer's disease.

Teresa Wu is an associate professor of Industrial Engineering at School of Computing, Informatics, Decision Systems Engineering, Arizona State University and an associate professor of Radiology at 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 University of Minnesota, Twin Cities in 2005. His research interests include machine learning, data mining, and biomedical informatics.

Xia Wu is an associate professor of College of Information Science and Technology of Beijing Normal University, 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 College of Information Science and Technology of Beijing Normal University, China. She received her Ph.D. from Chinese Academy of Science. Her main areas of interests are brain signal processing.

Appendix

I. Proof of Proposition 1

Using the Bayes' rule,

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

$$P(\Theta^h | \{\Theta_i^t\}_{i=1,2,\dots,m}) \propto \prod_{i=1}^m P(\Theta_i^t | \Theta^h) / P(\{\Theta_i^t\}_{i=1,2,\dots,m}) \quad (\text{A-1})$$

So, the first step is to derive $P(\{\Theta_i^t\}_{i=1,2,\dots,m})$. To do this, we condition $P(\{\Theta_i^t\}_{i=1,2,\dots,m})$ on Θ^h , i.e.,

$$P(\{\Theta_i^t\}_{i=1,2,\dots,m}) = \int_{\Theta^h} P(\{\Theta_i^t\}_{i=1,2,\dots,m} | \Theta^h) P(\Theta^h) d\Theta^h \propto \int_{\Theta^h} \prod_{i=1}^m P(\Theta_i^t | \Theta^h) d\Theta^h \quad (\text{A-2})$$

Because $\Theta_i^t \sim \text{Wishart}_v(\Theta^h)$,

$$P(\Theta_i^t | \Theta^h) \propto |\Theta^h|^{-v/2} |\Theta_i^t|^{(v-p-1)/2} e^{-\text{tr}(\Theta^{h-1} \Theta_i^t)/2} \quad (\text{A-3})$$

Inserting (A-3) into (A-2),

$$\begin{aligned} P(\{\Theta_i^t\}_{i=1,2,\dots,m}) &\propto \int_{\Theta^h} \prod_{i=1}^m \left\{ |\Theta^h|^{-v/2} |\Theta_i^t|^{(v-p-1)/2} e^{-\text{tr}(\Theta^{h-1} \Theta_i^t)/2} \right\} d\Theta^h \\ &= \frac{\prod_{i=1}^m |\Theta_i^t|^{\frac{v-p-1}{2}}}{|\sum_{i=1}^m \Theta_i^t|^{\frac{(mv-p-1)}{2}}} \int_{\Theta^h} |\Theta^h|^{-\frac{(mv-p-1)+p+1}{2}} \left| \sum_{i=1}^m \Theta_i^t \right|^{\frac{(mv-p-1)}{2}} e^{-\frac{1}{2} \text{tr}(\Theta^{h-1} \sum_{i=1}^m \Theta_i^t)} d\Theta^h. \end{aligned} \quad (\text{A-4})$$

Note that the function inside the integral in (A-4) happens to be proportional to the density function of an inverse-Wishart distribution with degrees of freedom $(mv - p - 1)$ and scale matrix $\sum_{i=1}^m \Theta_i^t$. So, the integral in (A-4) is actually a constant. Then, (A-4) can be simplified into

$$P(\{\Theta_i^t\}_{i=1,2,\dots,m}) \propto \frac{\prod_{i=1}^m |\Theta_i^t|^{\frac{v-p-1}{2}}}{|\sum_{i=1}^m \Theta_i^t|^{\frac{(mv-p-1)}{2}}} \quad (\text{A-5})$$

Inserting (A-5) and (A-3) into (A-1),

$$P(\Theta^h | \{\Theta_i^t\}_{i=1,2,\dots,m}) \propto \frac{|\sum_{i=1}^m \Theta_i^t|^{\frac{(mv-p-1)}{2}}}{\prod_{i=1}^m |\Theta_i^t|^{\frac{v-p-1}{2}}} \prod_{i=1}^m \left\{ |\Theta^h|^{-v/2} |\Theta_i^t|^{(v-p-1)/2} e^{-\text{tr}(\Theta^{h-1} \Theta_i^t)/2} \right\} \quad (\text{A-6})$$

$$= |\sum_{i=1}^m \Theta_i^t|^{\frac{(mv-p-1)}{2}} |\Theta^h|^{-\frac{mv}{2}} e^{-\frac{1}{2} \text{tr}(\Theta^{h-1} \sum_{i=1}^m \Theta_i^t)},$$

which happens to be proportional to the density function of an inverse-Wishart distribution

with degrees of freedom $(mv - p - 1)$ and scale matrix $\sum_{i=1}^m \Theta_i^t$

II. Proof of Proposition 2

Inserting (9) into (10),

$$Q(\{\Theta_i\}_{i=1,2,\dots,m} | \{\Theta_i^t\}_{i=1,2,\dots,m})$$

$$= \int_{\Theta^h} \sum_{i=1}^m \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{1}{2} \text{tr}(\Theta^{h-1} \Theta_i) - \frac{n_i}{2} \text{tr}(\mathbf{S}_i \Theta_i) - \lambda_i \|\Theta_i\|_1 \right\} P(\Theta^h | \{\Theta_i^t\}_{i=1,2,\dots,m}) d\Theta^h$$

. Some terms on the right-hand side of this equation do not include Θ^h , so they can be moved outside the integral, i.e.,

$$Q(\{\Theta_i\}_{i=1,2,\dots,m} | \{\Theta_i^t\}_{i=1,2,\dots,m})$$

$$= \sum_{i=1}^m \left\{ \frac{v-p-1+n_i}{2} \log |\Theta_i| - \frac{n_i}{2} \text{tr}(\mathbf{S}_i \Theta_i) - \lambda_i \|\Theta_i\|_1 \right\} \quad (\text{A-7})$$

$$- \int_{\Theta^h} \sum_{i=1}^m \frac{1}{2} \text{tr}(\Theta^{h-1} \Theta_i) P(\Theta^h | \{\Theta_i^t\}_{i=1,2,\dots,m}) d\Theta^h$$

It can be seen from (A-7) that the key in getting $Q(\{\Theta_i\}_{i=1,2,\dots,m} | \{\Theta_i^t\}_{i=1,2,\dots,m})$ is the find the integral $\int_{\Theta^h} \sum_{i=1}^m \frac{1}{2} \text{tr}(\Theta^{h-1} \Theta_i) P(\Theta^h | \{\Theta_i^t\}_{i=1,2,\dots,m}) d\Theta^h$. Because $\text{tr}(\cdot)$ is a linear operator, this integral becomes

$$\int_{\Theta^h} \sum_{i=1}^m \frac{1}{2} \text{tr}(\Theta^{h-1} \Theta_i) P(\Theta^h | \{\Theta_i^t\}_{i=1,2,\dots,m}) d\Theta^h = \sum_{i=1}^m \frac{1}{2} \text{tr}(\mathbf{D} \Theta_i), \quad (\text{A-8})$$

where $\mathbf{D} = \int_{\Theta^h} \Theta^{h-1} P(\Theta^h | \{\Theta_i^t\}_{i=1,2,\dots,m}) d\Theta^h$. Now we need to derive $\mathbf{D}$. Specifically, according to Proposition 1,

$$P(\Theta^h | \{\Theta_i^t\}_{i=1,2,\dots,m}) = \frac{|\sum_{i=1}^m \Theta_i^t|^{\frac{(mv-p-1)}{2}} |\Theta^h|^{-\frac{mv}{2}} e^{-\frac{1}{2} \text{tr}(\Theta^{h-1} \sum_{i=1}^m \Theta_i^t)}}{2^{\frac{(mv-p-1)p}{2}} \Gamma_p(\frac{mv-p-1}{2})} \quad \text{Also, } d\Theta^h = |\Theta^h|^{p+1} d\Theta^{h-1}.$$

Therefore, $\mathbf{D}$ becomes

$$\mathbf{D} = \int_{\Theta^{h-1}} \Theta^{h-1} \frac{|\sum_{i=1}^m \Theta_i^t|^{\frac{(mv-p-1)}{2}} |\Theta^{h-1}|^{\frac{mv-p-1-p-1}{2}} e^{-\frac{1}{2} \text{tr}(\Theta^{h-1} \sum_{i=1}^m \Theta_i^t)}}{2^{\frac{(mv-p-1)p}{2}} \Gamma_p(\frac{mv-p-1}{2})} d\Theta^{h-1}, \quad (\text{A-9})$$

which happens to be the mean of a Wishart distribution for Θ^{h-1} with degrees of freedom equal to $mv - p - 1$ and scale matrix $(\sum_{i=1}^m \Theta_i^t)^{-1}$, i.e.,

$$\mathbf{D}=(mv-p-1)(\sum_{i=1}^m \mathbf{\Theta}_i^t)^{-1}. \quad (\text{A-10})$$

Furthermore, inserting (A-10) into (A-8) and then into (A-7),

$$\begin{aligned} Q(\{\mathbf{\Theta}_i\}_{i=1,2,\dots,m}|\{\mathbf{\Theta}_i^t\}_{i=1,2,\dots,m}) \\ = \sum_{i=1}^m \left\{ \frac{v-p-1+n_i}{2} \log|\mathbf{\Theta}_i| - \frac{n_i}{2} \text{tr}(\mathbf{S}_i \mathbf{\Theta}_i) - \lambda_i \|\mathbf{\Theta}_i\|_1 - \frac{1}{2} \text{tr} \left((mv-p-1) (\sum_{i=1}^m \mathbf{\Theta}_i^t)^{-1} \mathbf{\Theta}_i \right) \right\} \end{aligned}$$

III. BCD algorithm for solving the optimization in (12) and proof of its convergence

The basic idea of the BCD algorithm is to update each column (or row) of $\mathbf{\Theta}_i$ iteratively while fixing all other columns (or rows), until convergence. For notation simplicity, we change the notations of (12) in the following way: We drop the subscript “ i ” which is the index for the tasks, since we will apply the BCD algorithm to all the tasks in the same way. Also, we drop the superscripts “ t ” and “ $t+1$ ” which represent the t -th iteration of the EM algorithm, because we will apply the BCD algorithm in every iteration of the EM algorithm. Therefore, (12) becomes

$$\mathbf{\Theta}^* = \arg\max_{\mathbf{\Theta}} \{ \log|\mathbf{\Theta}| - \text{tr}(\mathbf{S}\mathbf{\Theta}) - \lambda' \|\mathbf{\Theta}\|_1 \}, \quad (\text{A-11})$$

where $\mathbf{\Theta}^*$ denotes the optimal solution to $\mathbf{\Theta}$. In what follows, we will show how to apply the BCD algorithm to solve for (A-11).

Specifically, because the BCD algorithm works by iterations, we will only illustrate the steps in one iteration and other iterations work in a similar way. At a certain iteration, we first need to partition the current $\mathbf{\Theta}$ as follows. Let $\mathbf{\Theta}_{\setminus j}$ be the matrix produced by removing row j and column j from $\mathbf{\Theta}$, θ_{jj} be the element at row j and column j of $\mathbf{\Theta}$, and $\mathbf{\Theta}_j$ be the

column j of $\mathbf{\Theta}$ with θ_{jj} removed. Then, $\mathbf{\Theta}$ can be partitioned as $\mathbf{\Theta} = \begin{bmatrix} \mathbf{\Theta}_{\setminus j} & \mathbf{\Theta}_j \\ \mathbf{\Theta}_j^T & \theta_{jj} \end{bmatrix}$, and

correspondingly $\mathbf{S}$ can be partitioned as $\mathbf{S} = \begin{bmatrix} \mathbf{S}_{\setminus j} & \mathbf{S}_j \\ \mathbf{S}_j^T & s_{jj} \end{bmatrix}$. Next, we want to update $\mathbf{\Theta}_j$ and θ_{jj} while holding other elements in $\mathbf{\Theta}$ constant. To do this, let f represent the objective function in (A-7), i.e., $f = \log|\mathbf{\Theta}| - \text{tr}(\mathbf{S}\mathbf{\Theta}) - \lambda' \|\mathbf{\Theta}\|_1$; take the partial derivatives of f with respect to $\mathbf{\Theta}_j$ and θ_{jj} , respectively; and then make the partial derivatives to be zero, i.e.,

$$\frac{\partial f}{\partial \mathbf{\Theta}_j} = -\frac{2}{\theta_{jj} - \mathbf{\Theta}_j^T \mathbf{\Theta}_{\setminus j}^{-1} \mathbf{\Theta}_j} \mathbf{\Theta}_{\setminus j}^{-1} \mathbf{\Theta}_j - \mathbf{S}_j - \lambda' \text{SGN}(\mathbf{\Theta}_j) = 0, \text{ and} \quad (\text{A-12})$$

$$\frac{\partial f}{\partial \theta_{jj}} = \frac{1}{\theta_{jj} - \mathbf{\Theta}_j^T \mathbf{\Theta}_{\setminus j}^{-1} \mathbf{\Theta}_j} - s_{jj} - \lambda' = 0, \quad (\text{A-13})$$

where $\text{SGN}(\mathbf{\Theta}_j)$ denotes the partial derivative of $\|\mathbf{\Theta}\|_1$ with respect to $\mathbf{\Theta}_j$. It is difficult to solve for $\mathbf{\Theta}_j$ and θ_{jj} from (A-12) and (A-13) directly. Therefore, we adopt the following strategies.

Letting $\mathbf{a} = -\frac{\Theta_j}{\theta_{jj} - \Theta_j^T \Theta_{\setminus j \setminus j}^{-1} \Theta_j}$, then (A-12) and (A-13) become

$$2\Theta_{\setminus j \setminus j}^{-1} \mathbf{a} - \mathbf{S}_j + \lambda' \text{SGN}(\mathbf{a}) = 0 \quad (\text{A-14})$$

$$\mathbf{a} = -(s_{jj} + \lambda') \Theta_j. \quad (\text{A-15})$$

It is clear that (A-14) is also the result of making the partial derivative of g with respect to $\mathbf{a}$ to be zero in the following optimization problem:

$$\min_{\mathbf{a}} g = \mathbf{a}^T \Theta_{\setminus j \setminus j}^{-1} \mathbf{a} - \mathbf{S}_j^T \mathbf{a} + \lambda' \|\mathbf{a}\|_1, \quad (\text{A-16})$$

which is equivalent to the following min-max problem:

$$\max_{\kappa} \min_{\mathbf{a}} g = 2 \left(-\frac{1}{2} \kappa^T \Theta_{\setminus j \setminus j} \kappa + \kappa^T \mathbf{a} \right) - \mathbf{S}_j^T \mathbf{a} + \lambda' \|\mathbf{a}\|_1. \quad (\text{A-17})$$

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

After $\mathbf{a}$ and κ are obtained, (A-15) can be used to find Θ_j , i.e., $\Theta_j = -\frac{\mathbf{a}}{s_{jj} + \lambda'}$. Furthermore, based on (A-13), θ_{jj} can be obtained, i.e., $\theta_{jj} = \frac{(-\mathbf{a}^T \kappa + 1)}{s_{jj} + \lambda'}$.

Proof of convergence

According to (Tseng, 2001), the BCD algorithm converges if and only if (A-16) has a unique solution of $\mathbf{a}$ at each iteration. The unique solution is guaranteed if the optimization problem in (A-16) is strictly convex. The strict convexity is true if $\Theta_{\setminus j \setminus j}$ is positive definite, denoted by $\Theta_{\setminus j \setminus j} \succ 0$ if $\Theta \succ 0$. Therefore, the key to prove the convergence of the BCD algorithm is to prove $\Theta \succ 0$. Recall that the BCD algorithm works by iterations and (A-16) needs to be solved at each iteration. As a result, we need to prove $\Theta \succ 0$ at each iteration. To achieve this, we use mathematical induction, which includes a basis step and an inductive step:

Let $^j\Theta$ be the Θ 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 $^0\Theta \succ 0$.

Inductive step

Assuming $^{j-1}\Theta \succ 0$, we need to prove $^j\Theta \succ 0$, which is equivalent to prove $|^j\Theta| > 0$. Because $|^j\Theta| = |^{j-1}\Theta_{\setminus j \setminus j}| ({}^j\theta_{jj} - {}^j\Theta_j^T {}^{j-1}\Theta_{\setminus j \setminus j}^{-1} {}^j\Theta_j)$, we need to prove $|^{j-1}\Theta_{\setminus j \setminus j}| > 0$ and $({}^j\theta_{jj} - {}^j\Theta_j^T {}^{j-1}\Theta_{\setminus j \setminus j}^{-1} {}^j\Theta_j) > 0$. $|^{j-1}\Theta_{\setminus j \setminus j}| > 0$ is true since we have assumed $^{j-1}\Theta \succ 0$. Based on (A-10), ${}^j\theta_{jj} - {}^j\Theta_j^T {}^{j-1}\Theta_{\setminus j \setminus j}^{-1} {}^j\Theta_j = 1/(s_{jj} + \lambda') > 0$. As a result, $|^j\Theta| > 0$ and $^j\Theta \succ 0$.

Fig. 1.

A GGM and the corresponding IC matrix ($\theta_{i,j} = 0$; only entries at the upper triangle are shown because the matrix is symmetric and the diagonal entries are not used in the GGM)

Fig. 2.
A BHM framework for characterizing task relatedness

Input: sample covariance matrixes, $\{\mathbf{S}_i\}_{i=1,2,\dots,m}$; sample size for each task, n_i ; number of variable, p ; number of tasks, m ; regularization parameters, $\{\lambda_i\}_{i=1,2,\dots,m}$; degrees of freedom in the common prior Wishart distribution, v ; initial $\{\boldsymbol{\Theta}_i^0 > 0\}_{i=1,2,\dots,m}$; stopping criterion, ϵ .

Initialize:
 Compute $\{\mathbf{S}_i^0\}_{i=1,2,\dots,m}$ by (13);
 Let *converge* = *false*;
 Let $t = 0$;

Repeat
 Use the BCD algorithm to solve (12) and get $\{\boldsymbol{\Theta}_i^{t+1}\}_{i=1,2,\dots,m}$;
 Compute $\{\mathbf{S}_i^{t+1}\}_{i=1,2,\dots,m}$ by (13);
If $\sum_{i=1}^m \|\boldsymbol{\Theta}_i^{t+1} - \boldsymbol{\Theta}_i^t\|_2 \leq \epsilon$ **then**
 converge = *true*;
Else
 converge = *false*;
End if
 Let $t = t + 1$;

Until *converge* = *true*

Output: $\{\boldsymbol{\Theta}_i^{t+1}\}_{i=1,2,\dots,m}$ as the estimated IC matrix (i.e., GGM) for each task.

Fig. 3.

Steps for solving the transfer learning formulation in GGM learning

$$\begin{aligned}
 \Theta_h = & \begin{bmatrix} 0 & \theta_{13} & \theta_{14} & 0 & 0 \\ & \theta_{23} & 0 & 0 & 0 \\ & 0 & \theta_{35} & 0 & \\ & & \theta_{45} & \theta_{46} & \\ & & & 0 & \end{bmatrix} \xrightarrow{\text{(ii.1)}} \begin{bmatrix} 0 & \theta_{13} & 0 & 0 & 0 \\ & \theta_{23} & 0 & 0 & 0 \\ & 0 & 0 & 0 & \\ & & \theta_{45} & 0 & \\ & & & 0 & \end{bmatrix} \xrightarrow{\text{(ii.2)}} \begin{bmatrix} 0 & \theta_{13} & 0 & 0 & u_{16} \\ & \theta_{23} & 0 & u_{23} & 0 \\ & & u_{34} & 0 & \\ & & & \theta_{45} & 0 \\ & & & & 0 \end{bmatrix} \xrightarrow{\text{(ii.3)}} \begin{bmatrix} 0 & \theta_{13}+u_{13} & 0 & 0 & u_{16} \\ & \theta_{23}+u_{23} & 0 & u_{23} & 0 \\ & & u_{34} & 0 & \\ & & & \theta_{45}+u_{45} & 0 \\ & & & & 0 \end{bmatrix} \triangleq \Theta_i
 \end{aligned}$$

(ii.1) Randomly modify 3 (6x50%) nonzero entries in Θ_h to zero
 (ii.2) Randomly modify 3 zero entries in Θ_h to nonzero $\mu_{ij} \sim \text{Uniform}(D)$
 (ii.3) Re-sample remaining nonzero entries in Θ_h from $\text{Uniform}(D)$ and add to the θ_{ij}

Fig. 4. Substeps for constructing the IC matrix of each task, Θ_i , from Θ^h (s%=50%).

Fig. 5. Mean ROC curves for transfer learning (solid curve) and single task learning (dash curve) with task relatedness $s\% = 90\%$

Fig. 6. Mean ROC curves for transfer learning (solid curve) and single task learning (dash curve) with task relatedness $s\% = 50\%$

Fig. 7.
Mean ROC curves for transfer learning (solid curve) and single task learning (dash curve)
with task relatedness $s\% = 0\%$

Fig. 8.

Boxplots for numbers of arcs within and between lobes in learned connectivity networks by (a) transfer learning (left panel), and (b) single task learning (right panel). (F: frontal, P: parietal, O: occipital, T: temporal; for the two boxplots within a lobe or between a pair of lobes, the left boxplot represents AD and right boxplot represents NC.)

Table 1

CPU time (in seconds) of the simulation studies by transfer learning

	p=50	p=100	p=200
m=2	6.34	25.13	102.35
m=5	15.98	65.57	225.85
m=10	36.12	140.76	562.74

Table 2

CPU time (in seconds) of the simulation studies by single task learning

	p=50	p=100	p=200
m=2	2.87	9.34	24.98
m=5	5.21	29.88	73.47
m=10	11.35	52.19	101.53

Table 3

Names of the brain regions selected for brain connectivity network modeling (‘‘L’’ means that the brain region is located at the left hemisphere, ‘‘R’’ means right hemisphere.)

	Prefrontal lobe	Parietal lobe	Occipital lobe	Temporal lobe
1	Frontal_Sup_L	Parietal_Sup_L	Occipital_Sup_L	Temporal_Sup_L
2	Frontal_Sup_R	Parietal_Sup_R	Occipital_Sup_R	Temporal_Sup_R
3	Frontal_Mid_L	Parietal_Inf_L	Occipital_Mid_L	Temporal_Pole_Sup_L
4	Frontal_Mid_R	Parietal_Inf_R	Occipital_Mid_R	Temporal_Pole_Sup_R
5	Frontal_Sup_Medial_L	Precuneus_L	Occipital_Inf_L	Temporal_Mid_L
6	Frontal_Sup_Medial_R	Precuneus_R	Occipital_Inf_R	Temporal_Mid_R
7	Frontal_Mid_Orb_L	Cingulum_Post_L		Temporal_Pole_Mid_L
8	Frontal_Mid_Orb_R	Cingulum_Post_R		Temporal_Pole_Mid_R
9	Rectus_L			Temporal_Inf_L 8301
10	Rectus_R			Temporal_Inf_R 8302
11	Cingulum_Ant_L			Fusiform_L
12	Cingulum_Ant_R			Fusiform_R
				Hippocampus_L
				Hippocampus_R
				ParaHippocampal_L
				ParaHippocampal_R

Table 4

P-values for hypothesis testing on AD vs. NCs comparison based on learned connectivity networks by transfer learning

(a) Total num. of arcs= 180				
	Frontal	Parietal	Occipital	Temporal
Frontal	0.0009	0.7604	1	0.0001
Parietal		0.0002	0.5257	0.4753
Occipital			0.0125	0.0829
Temporal				0.2143

(b) Total num. of arcs= 120				
	Frontal	Parietal	Occipital	Temporal
Frontal	0.0001	0.546	0.6229	0.0000
Parietal		0.0003	0.0466	0.9005
Occipital			0.1153	0.2314
Temporal				0.0657

(b) Total num. of arcs= 90				
	Frontal	Parietal	Occipital	Temporal
Frontal	0.0033	0.2924	0.3343	0.0005
Parietal		0.0002	0.1331	0.6983
Occipital			0.1158	0.621
Temporal				0.1282

Table 5

P-values for hypothesis testing on AD vs. NCs comparison based on learned connectivity networks by single task learning

(a) Total num. of arcs= 180				
	Frontal	Parietal	Occipital	Temporal
Frontal	0.4264	0.5497	0.644	0.9579
Parietal		0.3953	0.6733	0.9006
Occipital			0.2215	1
Temporal				0.874

(b) Total num. of arcs= 120				
	Frontal	Parietal	Occipital	Temporal
Frontal	0.4276	0.5481	0.7669	0.3731
Parietal		0.9224	0.9041	0.8764
Occipital			0.66	0.7984
Temporal				0.8111

(c) Total num. of arcs=90				
	Frontal	Parietal	Occipital	Temporal
Frontal	Frontal	0.6949	0.4827	0.5587
Parietal	Parietal		1	0.5018
Occipital	Occipital			0.4005
Temporal	Temporal			