Solving the MEG Inverse Problem: A Robust Two-Way Regularization Method

Tian Siva Tian
Department of Psychology, University of Houston
and
Jianhua Z. Huang
Department of Statistics, Texas A&M University
and
Haipeng Shen
Department of Statistics and Operations Research
University of North Carolina at Chapel Hill
January 13, 2014

Abstract

Magnetoencephalography (MEG) is a common noninvasive imaging modality for instantaneously measuring whole brain activities. One challenge in MEG data analysis is how to minimize the impact of the outliers that commonly exist in the images. In this paper, we propose a robust two-way regularization approach to solve the important MEG inverse problem, i.e., reconstructing neuronal activities using the measured MEG signals. The proposed method is based on the distributed source model and produces a spatio-temporal solution for all the dipoles simultaneously. Unlike the traditional methods that utilize the squared error loss function, our proposal uses a robust loss function, which improves the robustness of the results against outliers. To impose desirable spatial focality and temporal smoothness, we then penalize the robust loss through appropriate spatial-temporal two-way regularization. Furthermore, an alternating reweighted least squares algorithm is developed to optimize the penalized model fitting criterion. Extensive simulation studies and a real-world MEG study clearly demonstrate the advantages of the proposed method over three non-robust methods.

Keywords: Two-way regularization; Inverse problem; Source reconstruction; Robustness; Spatio-temporal data; MEG.
1 Introduction

Magnetoencephalography (MEG) is a common imaging modality for non-invasively and instantaneously measuring whole brain activities through a set of sensors near or at the scalp. It provides direct measurements of the diminutive magnetic fields. One important problem of MEG data analysis is to reconstruct cortical or subcortical neuronal generators using the measured sensor-level signals, which is referred in the literature as either the MEG inverse problem or source reconstruction problem. This problem is challenging because there are infinitely many current source configurations that lead to the same magnetic field (Pascual-Marqui, 1999) and thus it is non-trivial to identify a practically meaningful solution from the infinitely many mathematically correct solutions. Many approaches have been developed to solve the MEG inverse problem. However, very few existing methods have addressed the issue of robustness against outliers that are common in the MEG data. In this paper, we propose a robust two-way spatial-temporal regularization method for MEG source reconstruction.

Existing source reconstruction methods can be broadly distinguished by their respective source models: the equivalent current dipole (ECD) model and the distributed source (DS) model. Methods based on the ECD model attempt to reconstruct the real current sources in the brain using a limited number (usually less than 5) of ECD with arbitrary locations and orientations. By applying prior knowledge to the number of sources, one first estimates the locations and then the orientations and amplitudes of the source signals. Some potential problems with these methods are: 1) when the number of sources are unknown, these methods are not suitable; 2) estimating the source locations usually involves a complex nonlinear optimization problem with multiple local optima (Darvas et al., 2004). However, Bayesian approaches have been recently developed to overcome these problems by assigning a prior to the number of sources and using posterior inference. Computationally expensive methods such as Markov Chain Monte Carlo are usually used to sample from the posterior distribution (Jun et al., 2005).

The DS model-based methods consider a large number (usually greater than 5000) of dipoles distributed at fixed locations over a grid over the cortical region. A forward model is first constructed using the anatomic structure of the brain which in turn is obtained from high spatial resolution Magnetic Resonance Images (MRI) (Dale and Sereno, 1993). The neural activities are then modeled as a discretized version of the continuous vector field in the cortical level. The elec-
tromagnetic fields obey Maxwell’s equations; the measured magnetic signal is a linear combination of the magnetic fields induced by each individual dipole. One advantage of the DS model is that it estimates the activities for all the dipoles simultaneously without estimating the location of a dipole beforehand. Since the problem is highly ill-posed, constraints are added in two formats: prior distribution (see, for example, Kaipio and Somersalo, 2005; Lucka et al., 2012) and regularization (see the text below for a review).

The proposed method belongs to the category of DS model-based methods. The measured magnetic signal is a linear combination of primary dipole activity:

\[ \mathbf{Y} = \mathbf{X} \mathbf{B} + \mathbf{E}, \]  

where \( \mathbf{Y} \in \mathbb{R}^{n \times s} \) is the matrix of MEG recordings from \( n \) sensors over \( s \) time points, \( \mathbf{X} \in \mathbb{R}^{n \times p} \) is the forward model matrix representing the 3-D orientation and location information of the underlying \( p/3 \) dipoles, \( \mathbf{B} \in \mathbb{R}^{p \times s} \) is the matrix containing the \( p \) unknown source dynamic time courses across \( s \) time points, and the matrix \( \mathbf{E} \in \mathbb{R}^{n \times s} \) contains noises as well as potential outliers.

Note that the methods assume that the dipole orientations are unknown; hence, the dipole activities can be decomposed into three directions (\( x, y, z \)) in the Cartesian coordinate system, which is why the number of dipoles is \( p/3 \). In many MEG studies, \( p \) can be over 10,000, but \( n \) is only a few hundred. Therefore, constraints are needed in order to obtain a unique solution, which are usually placed in the form of regularization on \( \mathbf{B} \) based on the properties of the brain anatomical structure and the nature of the neuronal signals. More specifically, most DS approaches solve the following penalized least squares problem:

\[
\min_{\mathbf{B}} \left\{ \| \mathbf{Y} - \mathbf{X} \mathbf{B} \|_F^2 + \lambda \text{pen}(\mathbf{B}) \right\},
\]

where \( \| \mathbf{A} \|_F = \sqrt{\text{tr}(\mathbf{A}^T \mathbf{A})} \) is the Frobenius norm of the matrix \( \mathbf{A} \), \( \text{pen}(\cdot) \) is a penalty function that regularizes the unknown sources to enforce certain desirable properties, and \( \lambda \) is the penalty parameter controlling the degree of penalization. By using regularization, the active dipoles are automatically selected. In addition, the constraints force the solution to meet some desirable properties.

Commonly used regularization techniques include the \( L_2 \) penalization (see, for example, Hämäläinen and Ilmoniemi, 1994; Pascual-Marqui et al., 1994; Gorodnitsky and Rao, 1997; Baillet and Garnero,
1997; Pascual-Marqui, 2002; Daunizeau et al., 2006; Nummenmaa et al., 2007; Valdés-Sosa et al., 2009), the $L_1$ penalization (Matsuura and Okabe, 1995; Uutela et al., 1999; Lin et al., 2006; Ding and He, 2008; Valdés-Sosa et al., 2009), and some $L_l$ penalizations with $0 < l < 1$ or $1 < l < 2$ (Auranen et al., 2005; Jeffs et al., 1987). More recently, several spatio-temporal methods have been proposed to incorporate the dependency between consecutive measurements (Trujillo-Barreto et al., 2008; Ou et al., 2009; Bolstad et al., 2009; Tian and Li, 2011; Tian et al., 2012).

In particular, Tian et al. (2012) developed a two-step two-way regularization (TWR) method that employs both spatial and temporal regularizations to impose desirable spatio-temporal properties of the sources, namely, spatial focality and temporal smoothness. Spatial focality means that only a small number of compact spatial regions are contributing to the spatial variation in the MEG measurements. Temporal smoothness means that the source dynamic functions are relatively smooth even though the amplitudes can change rapidly. It has been shown that TWR outperforms many DS methods with different one-way regularizations and several other spatio-temporal methods (Tian et al., 2012). Tian et al. (2013) improved TWR by integrating the two-steps of TWR into one optimization objective function and developed an efficient algorithm. However, as with other DS methods, the results of TWR can be sensitive to outliers since it uses the squared-error loss as the criterion to measure the goodness-of-fit.

In this paper, we develop a robust version of TWR, called robust two-way spatial-temporal regularization (RobTWR). RobTWR replaces the squared-error loss in TWR of Tian et al. (2013) by a robust loss function such as the Huber’s function. Similar to TWR, we employ separate spatial and temporal regularizations to achieve desirable spatial-temporal properties. To optimize the objective function, we decompose the spatial and temporal features in $B$ into two separate matrices. Then we alternate the optimization with respect to the spatial and the temporal matrices iteratively. An alternating weighted least-squares algorithm is implemented to optimize the objective function with respect to the spatial component and the temporal component element by element.

To the best of our knowledge, this paper is the first attempt to construct robust MEG inverse solver using the DS model. For the ECD model, Wipf et al. (2010) proposed an empirical Bayesian scheme that attempts to obtain robust solutions of the inverse problem. Some state-space methods using either ECD model (Sorrentino et al., 2009; Pascarella et al., 2010; Sorrentino et al., 2013) or DS model (Long et al., 2011) have been proposed to treat the source reconstruction as a dynamic filtering problem. Some of these methods (for example, Sorrentino et al., 2009, 2013) can also be
extended to be robust against outliers by using non-normal distributions to model the noise term.

The remainder of the paper is organized as follows: Section 2 first briefly reviews TWR and then introduces the proposed RobTWR method, along with the implementation algorithm and associated issues in tuning parameter selection. Sections 3 and 4 present an extensive simulation study and a real-world MEG study, respectively, to demonstrate the performance of RobTWR.

2 Methodology

In this section, we first briefly review the TWR method of Tian et al. (2012) in Section 2.1 and discuss its pros and cons. We then introduce our formulation of RobTWR in Section 2.2, which is a robust extension of TWR. An efficient computational algorithm is described in Section 2.3. We discuss data-driven tuning parameter selection in Section 2.4.

2.1 Review of TWR

Considering Model (1), the TWR method is a two-step procedure that starts with a suitable raw estimate of $B$, denoted as $\tilde{B}$, which is then decomposed into two rank-$q$ matrices:

$$\tilde{B} = AG^T,$$

where the $p$-by-$q$ matrix $A$ contains the spatial coefficients, the $s$-by-$q$ matrix $G$ contains the temporal components, and $q \leq \min\{p, s\}$ is the rank of the approximation. This spatial-temporal decomposition makes it natural to impose separate spatial and temporal penalizations, such that desirable spatial and temporal structural properties can be obtained. TWR minimizes the following function:

$$\min_{A,G}\left\{ \|\tilde{B} - AG^T\|_F^2 + \mu_1|A| + \mu_2tr(G^T\Omega G) \right\},$$

where $|A| = \sum_j \sum_k |a_{jk}|$ is the $L_1$-norm of $A$, $a_{jk}$ is the $(j, k)$th element of $A$, $\Omega$ is a non-negative definite roughness penalty matrix, and the two tuning parameters $\mu_1$ and $\mu_2$ control the degrees of spatial sparsity and temporal smoothness, respectively. The authors proposed to estimate the matrices $A$ and $G$ by alternating the optimization with respect to $A$ and $G$ separately.
2.2 Formulation of RobTWR

The two-step procedure of (Tian et al., 2012) does not directly solve the inverse problem and is sub-optimal. We target a single-step procedure that is expected to produce more accurate reconstruction. To this end, we decompose $B$ as the product of a spatial matrix $A$ and a temporal matrix $G$, i.e.

$$B = AG^T = \sum_{k=1}^{q} a_k g_k^T,$$

(5)

where $a_k$ is the $k$th column of $A$ and $g_k$ is the $k$th column of $G$. We then replace the squared loss function in (2) with a robust loss function, and with desirable spatial and temporal penalizations, we consider the following minimization problem:

$$\min_{a_k, g_k} \left\{ \rho \left( \frac{Y - X \sum_{k=1}^{q} a_k g_k^T}{\sigma} \right) + \sum_{k=1}^{q} \mu_{1k} |a_k| + \sum_{k=1}^{q} \mu_{2k} g_k^T \Omega g_k \right\}.$$

(6)

where $\rho(\cdot)$ is a robust loss function, $\sigma$ is a scale parameter measuring the variability of the residuals from the goodness-of-fit, and two sequences of tuning parameters $\mu_{1k}$ and $\mu_{2k}$ allow different components $a_k$ and $g_k$ to have different amount of sparsity and smoothness, respectively.

In addition to being more robust against outliers, the above RobTWR formulation is more flexible than the TWR formulation due to the flexibility of the tuning parameters. This flexibility does introduce additional computational burden, especially in data-driven selection of the tuning parameters, as a simultaneous selection involves a search over multiple-dimensional grids. To make the estimation less demanding, we propose to estimate the pairs $\{a_k, g_k\}$ sequentially by solving a serial of rank-one approximations. Below we start with illustrating how one can estimate the first pair $\{a_1, g_1\}$.

Consider the following generic rank-one approximation problem:

$$\min_{a, g} \left\{ \rho \left( \frac{Y - X a g^T}{\sigma} \right) + \mu_1 |a| + \mu_2 g^T \Omega g \right\}.$$

(7)

where $a$ is a $p$-vector and $g$ is an $s$-vector. We describe the algorithm for solving (7) in Section 2.3. The corresponding minimizers will be the estimates for $a_1$ and $g_1$, denoted as $\hat{a}_1$ and $\hat{g}_1$. Then, we can estimate the second pair $\{a_2, g_2\}$, by replacing $Y$ in (7) with the residual matrix from the rank-one approximation, i.e., $Y - X \hat{a}_1 \hat{g}_1^T$. Additional components can be estimated similarly with
Y being replaced by its residual matrix from the lower rank approximation.

In our implementation, we choose the robust loss function \( \rho(\cdot) \) in (7) to be the Huber’s loss function (Huber, 1973), which is defined as

\[
\rho(x) = \begin{cases} 
  x^2 & |x| \leq K \\
  2K|x| - K^2 & |x| > K 
\end{cases}
\]

where \( K \) is a thresholding parameter that controls the level of robustness: a smaller value of \( K \) usually leads to more robust estimation. In this paper we choose to use \( K = 1.345 \), the value that produces 95\% efficiency when the regression errors are normally distributed (Huber and Ronchetti, 2009), which is commonly used in the robust literature. Note that other robust loss function may be considered here as well.

2.3 Alternating Reweighted Least Squares Algorithm

In this section, we propose an alternating reweighted least squares algorithm to solve the minimization problem (7). The algorithm alternates the optimization with respect to \( a \) and \( g \) iteratively.

To begin with, we rewrite (7) as

\[
\min_{a, g} \left\{ \sum_{i=1}^{n} \sum_{j=1}^{s} \rho \left( \frac{y_{ij} - g_j}{\sigma} \sum_{l=1}^{p} x_{il} a_l \right) + \mu_1|a| + \mu_2 g^T \Omega g \right\},
\]

where \( y_{ij}, x_{il}, a_l, g_j \) are the \((i,j)\)th element of \( Y \), the \((i,l)\)th element of \( X \), the \(l\)th element of \( a \), and the \(j\)th element of \( g \), respectively.

2.3.1 Updating \( a \), Given \( g \)

When \( g \) is fixed as \( \hat{g} \), the roughness penalty in (8) vanishes, and the minimization problem becomes

\[
\min_a \left\{ \sum_{i=1}^{n} \sum_{j=1}^{s} \rho \left( \frac{y_{ij} - \hat{g}_j}{\sigma} \sum_{l=1}^{p} x_{il} a_l \right) + \mu_1|a| \right\},
\]

7
which can be reformulated as a penalized weighted least squares problem. For that purpose, we denote

\[ w_{ij} = \frac{\rho \left( y_{ij} - \hat{g}_j \sum_{l=1}^p x_{il} a_l \right)}{\left( y_{ij} - \hat{g}_j \sum_{l=1}^p x_{il} a_l \right)^2}, \]

which can be calculated by plugging in the estimates of \( a \) and \( g \) from the previous iteration. Then, (9) can be reexpressed as

\[
\min_a \left\{ \sum_{i=1}^n \sum_{j=1}^s \frac{w_{ij}}{\sigma^2} \left( y_{ij} - \hat{g}_j \sum_{l=1}^p x_{il} a_l \right)^2 + \mu_1 |a| \right\}. \tag{11}
\]

Note that the weights in (11) depend on \( a \), so the penalized weighted least squares problem should be solved iteratively, with \( a \) in the definition of the weights fixed at the value from the previous iteration. Using weighted least squares to approximate the robust loss is a standard method for computation of robust regression (see, for example Fox and Weisberg, 2011; Shen et al., 2007; Zhang et al., 2013).

Then, a coordinate descent algorithm can be applied to solve (11) for \( a \) element by element. After combining terms and excluding the constant terms, the objective function for estimating the \( k \)th element of \( a, a_k \), is

\[
R(a_k) = \sum_{i=1}^n \sum_{j=1}^s \frac{w_{ij}}{\sigma^2} \left[ \hat{g}_j^2 x_{ik}^2 a_k^2 - 2x_{ik} (y_{ij} \hat{g}_j - \hat{g}_j^2 \sum_{l \neq k} x_{il} a_l) a_k \right] + \mu_1 |a_k|.
\]

Define

\[
r_k = \frac{\sum_{i=1}^n \sum_{j=1}^s w_{ij} x_{ik} \hat{g}_j \left( y_{ij} - \hat{g}_j \sum_{l \neq k} x_{il} a_l \right)}{\sum_{i=1}^n \sum_{j=1}^s w_{ij} \hat{g}_j^2 x_{ik}^2}, \quad \text{and} \quad \lambda_k = \frac{\mu_1 \sigma^2}{\sum_{i=1}^n \sum_{j=1}^s w_{ij} \hat{g}_j^2 x_{ik}^2}.
\]

According to Lemma 2 of Shen and Huang (2008), the minimizer of \( R(a_k) \) follows the soft thresholding rule, i.e.

\[
\hat{a}_k = \text{sign}(r_k) \left( |r_k| - \lambda_k \right)_+ = \begin{cases} 
  r_k - \lambda_k, & \text{if } r_k > 0 \text{ and } \lambda_k < |r_k|, \\
  r_k + \lambda_k, & \text{if } r_k < 0 \text{ and } \lambda_k < |r_k|, \\
  0, & \text{if } \lambda_k \geq |r_k|.
\end{cases}
\]
where \( \text{sign}(\cdot) \) is the sign function, and \((x)_+ = \max(x, 0)\).

### 2.3.2 Updating \( g \), Given \( a \)

When \( a \) is fixed as \( \hat{a} \), \( g \) can be updated by a similar coordinate descent algorithm. To begin with, denote \( x^* = X\hat{a} \) and note that the problem (8) is equivalent to solving

\[
\min_g \left\{ \sum_{i=1}^{n} \sum_{j=1}^{s} \rho \left( \frac{y_{ij} - x_i^* g_j}{\sigma} \right) + \mu_2 g^T \Omega g \right\},
\]

(12)

where \( x_i^* \) is the \( i \)th element of the \( n \)-vector \( x^* \).

Again, the above minimization problem can be recasted as a penalized weighted least squares problem, by letting

\[
w_{ij} = \rho \left( \frac{y_{ij} - x_i^* g_j}{\sigma} \right)^2.
\]

(13)

Thus, the objective function for the problem (12) becomes the following penalized weighted least squares:

\[
R(g) = \sum_{i=1}^{n} \sum_{j=1}^{s} \frac{w_{ij}}{\sigma^2} (y_{ij} - x_i^* g_j)^2 + \mu_2 g^T \Omega g.
\]

(14)

This criterion is minimized iteratively while the \( g \) used for defining the weights is fixed at the value from the previous iteration. For the \( k \)th element of \( g \), eliminating all irrelevant terms, the objective function (14) becomes

\[
R(g_k) = -2 \sum_{i}^2 \frac{w_{ik}}{\sigma^2} y_{ik} x_i^* g_k + \sum_{i}^n \frac{w_{ik}}{\sigma^2} x_i^2 g_k^2 + \mu_2 [(g_{k-2} - 2g_{k-1} + g_k)^2 + (g_{k-1} - 2g_k + g_{k+1})^2 + (g_k - 2g_{k+1} + g_{k+2})^2].
\]

(15)

By calculating the first derivative with respect to \( g_k \) and setting it to zero, we obtain the minimizer as

\[
\hat{g}_k = \frac{\sum_{i}^n \frac{w_{ik}}{\sigma^2} y_{ik} x_i^* - \mu_2 (g_{k-2} - 4g_{k-1} - 4g_{k+1} + g_{k+2})}{\sum_{i}^n \frac{w_{ik}}{\sigma^2} x_i^2 + 5\mu_2},
\]

(16)

which allows one to get an updated estimate for \( g_k \) using the estimated \( g_{k-2}, g_{k-1}, g_{k+1}, g_{k+2} \) from the previous iteration. Note that \( w_{ij} \) can also be calculated by the estimated \( g_j \) from the previous iteration.
2.3.3 Estimation of $\sigma$

The scale parameter $\sigma$ needs to be estimated from the data during each updating $a$ or $g$. Specifically, consider the residual matrix $Y - X\hat{a}\hat{g}^T$, with the $(i,j)$th entry denoted as $r_{ij}$, where $\hat{a}$ and $\hat{g}^T$ are the estimates from the previous iteration. As suggested by Maronna et al. (2006), we estimate $\sigma$ by the normalized Median Absolute Deviation (MAD):

$$\hat{\sigma} = \frac{1}{0.675}\text{Med}_{ij}(|r_{ij}|, r_{ij} \neq 0).$$

2.3.4 Initialization and Convergence

We follow the suggestion of Tian et al. (2012) to obtain the initial estimates for $a_k$ or $g_k$. First, we get a rough estimate of $B$, by solving the following optimization problem:

$$\min_B \|B\|^2_F \text{ subject to } \|Y - XB\|_F = 0.$$ (17)

To solve (17), we first obtain a singular value decomposition (SVD) of $X$, i.e., $X = UDV^T$. Define $\tilde{Y} = U^T Y$ and $\tilde{C} = D^{-1}\tilde{Y}$. If $D$ has zero diagonal values, we use a reduced form of the SVD by only keeping the non-zero terms on the diagonal of $D$. It is obvious that there are infinite many $B$'s satisfying $\|Y - XB\|_F = 0$ and they all solve $\tilde{C} = V^TB$. Among them, the one with the minimum norm is the unique solution of (17) and can be computed as $\tilde{B} = V\tilde{C}$. Then we perform SVD on $\tilde{B}$ and let the initial collection of the $g_k$'s be the right singular vectors of $\tilde{B}$.

We iterate the optimization with respect to $a_1$ and $g_1$ as described above in Sections 2.3.1 and 2.3.2 until convergence. For the $i$th iteration, denote the estimates as $\hat{a}_1^{(i)}$ and $\hat{g}_1^{(i)}$. We declare that the iteration converges if $\|\hat{a}_1^{(i)}\hat{g}_1^{(i)T} - \hat{a}_1^{(i-1)}\hat{g}_1^{(i-1)T}\|_F / \|\hat{a}_1^{(i)}\hat{g}_1^{(i)T}\|_F \leq 10^{-6}$, i.e. if the Frobenius norm of the relative difference between the current estimate of $a_1g_1^T$ and its previous estimate is smaller than a pre-specified threshold value, which is set to be $10^{-6}$ in our implementation. Based on our empirical studies (Sections 3 and 4), only a few iterations (usually less 10) are needed to reach convergence. After $a_1$ and $g_1$ have been estimated, we obtain the residual matrix and iterate the optimization with respect to $a_2$ and $g_2$ until convergence. The algorithm is repeated until all $q$ pairs $\{a_k, g_k\}$ are estimated.
2.4 Tuning Parameter Selection

Our method involves $2q+1$ parameters: the rank $q$, and for each pair $\{a_k, g_k\}$, there are the focality penalty parameter $\mu_{1k}$ and the roughness penalty parameter $\mu_{2k}$. Below we describe how they can be chosen in a hierarchical data-driven fashion.

The selection of $\mu_{1k}$ and $\mu_{2k}$ is nested within the selection of $q$, and the selection of $\mu_{2k}$ is nested within the selection of $\mu_{1k}$. For the $k$th pair of $\{a_k, g_k\}$, the parameters $\mu_{1k}$ and $\mu_{2k}$ are chosen by cross-validation (CV) and generalized cross-validation (GCV), respectively, as we shall illustrate below. Note that our formulation allows $\mu_{1k}$ and $\mu_{2k}$ to be different for different $k$. The derivation below considers the generic rank-one problem (8), and denotes the penalty parameters generically as $\mu_1$ and $\mu_2$.

**Selection of the Roughness Parameter $\mu_2$.** We first discuss data-driven selection of the roughness penalty parameter $\mu_2$, given a fixed $\mu_1$. Consider the updating formula of $\hat{g}_k$ given in (16), which is a solution of a penalized regression. This suggests that we can select $\mu_2$ using the commonly used GCV criterion (Hastie and Tibshirani, 1990), although we need to replace the standard squared norm with the more robust Huber norm.

Let $W$ be the weighting matrix containing the weights $w_{ij}$ in (13) as its $(i,j)$ th element. In addition, define $w_j$ as the $j$th column of $W$, $\text{diag}(w_j)$ as the diagonal matrix with the entries of $w_j$ being the diagonal elements, and

$$W = \begin{pmatrix}
\text{diag}(w_1) & 0 & \ldots & 0 \\
0 & \text{diag}(w_2) & \ldots & 0 \\
\vdots & \vdots & \ddots & 0 \\
0 & 0 & \ldots & \text{diag}(w_s)
\end{pmatrix}_{ns \times ns}.$$

Furthermore, let $y_j$ be the $j$th column of $Y$, and denote

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_s \end{pmatrix}_{ns \times 1}, \quad X = \begin{pmatrix} x^* & 0 & \ldots & 0 \\
0 & x^* & \ldots & 0 \\
\vdots & \vdots & \ddots & 0 \\
0 & 0 & \ldots & x^* \end{pmatrix}_{ns \times s}.$$
Then, the updating formula (16) can be expressed as the matrix form below:

\[
\hat{g} = (\mathcal{X}^T \mathcal{W} \mathcal{X} + \mu_2 \sigma^2 \Omega)^{-1} \mathcal{X}^T \mathcal{W} \mathcal{Y}.
\]

Define the hat matrix \(\mathcal{H} = (\mathcal{X}^T \mathcal{W} \mathcal{X} + \mu_2 \sigma^2 \Omega)^{-1} \mathcal{X}^T \mathcal{W} \mathcal{X}\), which is a block diagonal matrix. Let \(\tilde{g} = (\mathcal{X}^T \mathcal{W} \mathcal{X})^{-1} \mathcal{X}^T \mathcal{W} \mathcal{Y}\) be the unregularized update of \(g\). Then we have \(\hat{g} = \mathcal{H} \tilde{g}\). The GCV criterion can be obtained as

\[
GCV(\mu_2) = \frac{1}{s} \sum_{j=1}^{s} \frac{\rho(\hat{g}_j - \tilde{g}_j)}{(1 - \text{trace}(\mathcal{H})/s)^2},
\]

where \(\tilde{g}_j\) is the \(j\)th element of \(\tilde{g}\).

Directly computing the hat matrix \(\mathcal{H}\) or its trace can be quite expensive computationally because of the high-dimensionality of \(\mathcal{W}\) and \(\mathcal{X}\). However, we can obtain an efficient algorithm by taking advantage of the specific structures of \(\mathcal{W}\) and \(\mathcal{X}\).

Let the matrix inversion term in the hat matrix be \(\mathcal{O} = (\mathcal{X}^T \mathcal{W} \mathcal{X} + \mu_2 \sigma^2 \Omega)^{-1}\). Then the \(j\)th diagonal block of \(\mathcal{H}\) is \(\mathcal{H}_j = \mathcal{O}_{jj} x^* x^T \mathcal{W}_j\), where \(\mathcal{O}_{jj}\) is the \(jj\)th element of matrix \(\mathcal{O}\) and \(\mathcal{W}_j\) is the \(j\)th diagonal block of matrix \(\mathcal{W}\). Note that \(\mathcal{X}^T \mathcal{W} \mathcal{X}\) is an \(s\)-by-\(s\) diagonal matrix with the \(j\)th diagonal element being \(x^T \text{diag}(\mathcal{w}_j) x^*\). Therefore, one can compute each diagonal element individually. Furthermore, we let \(\Omega = \mathbf{P} \Lambda \mathbf{P}^T\) be the eigen-decomposition of the roughness penalty matrix \(\Omega\). Then \(\mathcal{O}\) can be expressed as \(\mathcal{O} = \mathbf{P} (\mathcal{X}^T \mathcal{W} \mathcal{X} + \mu_2 \sigma^2 \Lambda)^{-1} \mathbf{P}^T\). This suggests that \(\mathcal{X}^T \mathcal{W} \mathcal{X} + \mu_2 \sigma^2 \Lambda\) is a diagonal matrix; hence its inverse can be easily obtained by inverting each diagonal element. Then the trace term in the GCV criterion (19) can be computed block-wise:

\[
\text{trace}(\mathcal{H}) = \sum_{j=1}^{s} \text{trace}(\mathcal{H}_j).
\]

In addition, it is easy to show that \(\text{trace}(\mathcal{H}_j) = \mathcal{O}_{jj} x^T \text{diag}(\mathcal{w}_j) x^*\), which suggests that \(\text{trace}(\mathcal{H})\) can be computed fairly fast.

Moreover, since the GCV criterion is a smooth function of \(\mu_2\), the optimization can be done using a combination of golden section search and successive parabolic interpolation (Brent, 1973). Note that the selection of \(\mu_2\) is done at each iteration for a fixed \(\mu_1\).

**Selection of the Focality Parameter \(\mu_1\).** We use a \(K\)-fold CV to choose \(\mu_1\). To be specific, we divide the data into \(K\) parts of similar size, denoted as \(\{\mathbf{Y}_{(k)}, \mathbf{X}_{(k)}\}, k = 1, \ldots, K\). For each \(k\), we leave out the \(k\)th part of the data for validation, and fit the model on the remaining \(K-1\) parts with a particular choice of \(\mu_1\), to obtain the estimates for \(\mathbf{a}\) and \(g\), denoted as \(\hat{\mathbf{a}}_{(-k)}\) and \(\hat{g}_{(-k)}\). We
then calculate the CV error as

$$CV(\mu_1) = \frac{1}{K} \sum_{k=1}^{K} \rho(Y^{(k)} - X^{(k)} \hat{a}^{(-k)} \hat{g}^{T^{(-k)}}).$$

(20)

The above CV error is evaluated over a set of candidate values for $\mu_1$. We then choose $\mu_1$ to minimize the CV criterion (20).

We use $K = 5$ for all the numerical examples in this paper, and the candidate pool for $\mu_1$ is from 0 to 1 in 20 equal increments.

**Selection of the Rank $q$.** To obtain a simple model, we wish to choose a low rank approximation with a small $q$, and that the $q$ can be selected efficiently. We have conducted some empirical studies and found that the results are not very sensitive to the choice of $q$ as long as $q$ is not too small. Therefore, the idea is to choose $q$ as small as possible while maintaining a satisfactory performance. For these purposes, we propose a forward algorithm to choose $q$. We first fit the rank-1 model and compute some stopping criterion. When the stopping criterion is met we stop the algorithm and return $q = 1$, otherwise we fit the rank-2 model and compute the stopping criterion, and so forth.

We make use of the CV error (20). We start with $q = 1$, and obtain the optimal $\mu_1$ and $\mu_2$, which gives us the minimized CV error, denoted as $CV_1$. We then set $err_1 = CV_1$. Then, we increase $q$ to 2 and compute $err_2 = (CV_1 + CV_2)/2$. If $|err_1 - err_2|/err_1 \leq \epsilon$, for a small $\epsilon$, then we choose the rank to be 2. Otherwise, we keep increasing $q$ until we find a rank $q_0$ such that $|err_{q_0-1} - err_{q_0}|/err_{q_0-1} \leq \epsilon$, in which case we select $q_0$ to be the chosen rank. In our implementation, we set the constant $\epsilon$ to be $10^{-4}$. From our empirical studies, $q$ tended to be fairly small in many cases. The algorithm usually stopped when $q$ is 2 or 3. This leads to a very low rank model.

### 3 Simulation Studies

We compare RobTWR with three other methods: the well-known MNE method (Hämäläinen and Ilmoniemi, 1994), the $L_1L_2$ method (Ou et al., 2009), and the TWR method (Tian et al., 2012) through a simulation study with different types of noise. The MNE method is widely used in practice, and it provides a golden standard on evaluating our method. Tian et al. (2012)
have conducted comparisons of TWR versus multiple other DS methods, and TWR has shown to dominate over other methods. Therefore, we choose to use TWR as a higher standard to beat. But for the record, we also compare with another recently proposed spatio-temporal method (the $L_1 L_2$ method).

The MNE method was coded in the statistical computing environment R (R Development Core Team, 2011). TWR and RobTWR were coded in Fortran-90 using Intel Visual Fortran Compiler XE 12.0 and are executed from R. All programs were run on a Dell desktop computer equipped with an Intel(R) Xeon(R) CPU (3.20 GHz) and 12.0 GB RAM. The tuning parameters were selected using the methods discussed in Section 2.4. We observed that the computation cost for RobTWR is comparable with that for TWR, and the number of iterations needed for convergence for RobTWR is also similar to that for TWR. The computation time for both RobTWR and TWR in our simulation studies is around 5000 seconds. MNE needs relatively less computational resources and the computation time is around 4000 seconds. The computation cost for the $L_1 L_2$ method is the most – more than 30,000 seconds.

We consider three comparison criteria. The overall mean squared error (MSE) is defined as

$$\text{MSE} = \frac{1}{p} \| B - \hat{B} \|^2_F,$$

where $p = 5124$ is the number of total dipole components, and $B$ and $\hat{B}$ are the true and estimated source matrices, respectively.

Then, we focus on each active area and calculate MSE over the dipoles in that particular active region, denoted as either $\text{MSE}_L$ or $\text{MSE}_R$, in which case $p = 30$ in the above equation. Note that $\text{MSE}_L$ stands for the active area in the left hemisphere.

Finally, we consider a location discrepancy measure, by computing the distance between the “center” of each active area and the “center” of the corresponding estimated active area. For a certain time point of interest $k$, the center of the active area $C_k$ is defined as the dipole location where the amplitude reaches the maximum. We can similarly define the center of the estimated active area $\hat{C}_k$. Then, the distance between the two centers is the Euclidean distance, defined as

$$d_k = C_k - \hat{C}_k = \sqrt{(C_{k,x} - \hat{C}_{k,x})^2 + (C_{k,y} - \hat{C}_{k,y})^2 + (C_{k,z} - \hat{C}_{k,z})^2},$$
where \( C_{k,x}, C_{k,y}, C_{k,z} \) are the three coordinates in the \( x, y, z \) directions for the center point \( C_k \), and \( \hat{C}_{k,x}, \hat{C}_{k,y}, \hat{C}_{k,z} \) are the corresponding coordinates for the estimated center point \( \hat{C}_k \).

In all the simulations studies, we generate synthetic data mimicking real MEG experiments according to Model (1). The forward operator \( X \) was obtained from a real human subject’s head MRI measurements through a head boundary element model using the MNE software (available at: \text{http://www.nmr.mgh.harvard.edu/martinos/userInfo/data/sofMNE.php} ), as described in Mosher et al. (1999). This forward operator \( X \) is a 248-by-15,372 matrix representing 5124 dipoles with the potential or field distributions on 248 MEG sensors for \( x, y, z \)-orientations.

### 3.1 Simulation 1

In this simulation, the signal sources matrix \( B \) was generated following Model 5. We considered the true rank as \( q = 2 \). We took two temporal components from some sine-exponential functions and then orthogonalized them to obtain the two temporal signals \( g_1 \) and \( g_2 \), which are shown in Figure 1 (a) and (b) across 50 sampling time points. To generate the spatial coefficients \( a_1 \) and \( a_2 \), we first defined two active areas in the left and right hemispheres, respectively, each of size 30 dipoles. The spatial coefficients corresponding to the dipoles in these areas were generated from a standard normal distribution, while the spatial coefficients outside were set to be zero. Finally, the 15,372-by-50 signal matrix \( B \) was obtained via \( B = \sum_{j=1}^{q} a_j g_j^T \) and is shown in Figure 1 (c). Each curve represents the dynamic function of a row of \( B \).

![Figure 1](image1.png)

**Figure 1:** The true temporal components and the source time courses, \( q = 2 \).

The error matrix \( E \) in (1) consists of two components: regular background noises and outliers.
The background noises were generated from a normal distribution with mean zero and variance $\sigma^2 = 0.3$ or 1. We consider 12 different simulation scenarios: 2 signal-to-noise-ratio settings, 3 outlier scenarios, and 2 levels of outlier percentage. The three types of outliers are cell outliers, sensor outliers, and time block outliers, as illustrated in Figure 2. Cell outliers are outliers randomly distributed over the cells of the data matrix. Sensor outliers occur when outliers affect all the recordings from certain MEG sensors (i.e. rows of the data matrix). Time block outliers refer to those occurring at certain time points across all rows. Under each outlier scenario, we study how the methods are affected by different amount of outliers, in particular considering either 5% or 10% outliers.

For cell outliers, we randomly selected 5% or 10% of the entries in $E$ and assigned values that are 10 times of the original values. Given that $E$ is 248-by-50, there are 620 or 1240 outliers, respectively. For sensor outliers, we randomly selected 12 or 24 rows of $E$, and for each entry in this row we assigned a value 10 times larger than its original value. For time block outliers, we took time points 11 to 12 or time points 11 to 15, and assigned large values to the entries in all the rows within these time ranges.

For each simulation scenario, we conducted 50 simulation runs with randomly generated background noise and randomly selected outliers for each run. Figure 3 shows the synthetic $Y$ from one particular simulation run for the 12 simulation scenarios.

Table 1 compares the estimation results for the three methods in all 12 scenarios. The distance between the true and estimated centers was computed at time point 7. For each comparison criterion, we report the average over the 50 simulation runs and the standard errors (in the paren-
Figure 3: Synthetic Signals under Each Simulation Scenario. Each row shows one type of outliers, for $\sigma^2 = 0.3$ and $\sigma^2 = 1$, and either 5% or 10% outliers.

ones). One can see that RobTWR outperforms $L_1L_2$, TWR, and MNE in all scenarios. The performance of RobTWR is fairly stable across all scenarios. MNE fails for all scenarios. $L_1L_2$ and TWR perform better than MNE, but their accuracy levels decrease as the noise level and the amount of outliers increase. Time block outliers seem to affect the recovery accuracy of the other methods the most, with sensor outliers being the second most difficult.

To further compare the temporal reconstruction performance, Figure 4 plots the reconstructed time courses at an arbitrary location within the active area for each method from one simulation
Table 1: Performance Comparison of Mean and Standard Error (in parentheses): MSE ($\times 10^{-5}$), $\text{MSE}_{L}^*$ ($\times 10^{-5}$), $\text{MSE}_{R}^*$ ($\times 10^{-5}$), $d_{7,L}$ ($\times 10^{-4}$), and $d_{7,R}$ ($\times 10^{-4}$).

<table>
<thead>
<tr>
<th>Outlier</th>
<th>$\sigma^2$</th>
<th>Outlier%</th>
<th>Method</th>
<th>MSE</th>
<th>$\text{MSE}_{L}^*$</th>
<th>$\text{MSE}_{R}^*$</th>
<th>$d_{7,L}$</th>
<th>$d_{7,R}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.3</td>
<td>5%</td>
<td>MNE</td>
<td>1237.7(41.4)</td>
<td>942.0(36.4)</td>
<td>922.9(37.9)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Cell</td>
<td></td>
<td></td>
<td>$L_1 L_2$</td>
<td>313.2(20.4)</td>
<td>206.2(15.6)</td>
<td>216.7(16.4)</td>
<td>17.2(1.5)</td>
<td>15.8(1.5)</td>
</tr>
<tr>
<td></td>
<td>10%</td>
<td></td>
<td>MNE</td>
<td>1299.8(41.6)</td>
<td>910.6(39.1)</td>
<td>956.7(37.0)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$L_1 L_2$</td>
<td>357.4(25.9)</td>
<td>267.6(23.8)</td>
<td>298.9(24.1)</td>
<td>17.9(1.8)</td>
<td>17.4(1.9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>MNE</td>
<td>1622.8(40.2)</td>
<td>944.5(40.0)</td>
<td>902.3(39.4)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Sensor</td>
<td>0.3</td>
<td>5%</td>
<td>MNE</td>
<td>1452.4(45.9)</td>
<td>856.6(35.9)</td>
<td>779.6(33.2)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>MNE</td>
<td>1647.9(42.8)</td>
<td>1175.2(40.9)</td>
<td>1100.0(38.5)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5%</td>
<td>$L_1 L_2$</td>
<td>438.2(28.6)</td>
<td>305.9(24.5)</td>
<td>312.0(25.6)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>MNE</td>
<td>1754.2(43.6)</td>
<td>985.2(38.8)</td>
<td>959.5(35.1)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Time</td>
<td>0.3</td>
<td>5%</td>
<td>MNE</td>
<td>1497.6(43.2)</td>
<td>850.6(35.9)</td>
<td>779.6(33.2)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Block</td>
<td></td>
<td>10%</td>
<td>MNE</td>
<td>1784.2(41.2)</td>
<td>1015.4(40.4)</td>
<td>1013.2(39.2)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5%</td>
<td>$L_1 L_2$</td>
<td>468.4(29.5)</td>
<td>328.6(24.9)</td>
<td>359.6(26.8)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>MNE</td>
<td>1876.1(43.6)</td>
<td>958.2(38.8)</td>
<td>959.5(35.1)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5%</td>
<td>$L_1 L_2$</td>
<td>523.2(30.2)</td>
<td>560.9(24.9)</td>
<td>553.9(23.1)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>MNE</td>
<td>1799.2(45.5)</td>
<td>915.7(39.2)</td>
<td>840.1(37.2)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5%</td>
<td>$L_1 L_2$</td>
<td>554.2(29.1)</td>
<td>532.6(25.8)</td>
<td>562.0(25.1)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>MNE</td>
<td>1521.0(42.5)</td>
<td>927.6(37.8)</td>
<td>869.4(38.9)</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
run. The black solid curves are the true source time courses, the light blue dashed curves are the estimated source time courses by MNE, the green dotted curves are the estimated source time courses by $L_1L_2$, the red dashed-dotted curves are the estimated source time courses by TWR, and the blue dashed curves are the estimated source time courses by RobTWR. The rows correspond to the settings with cell outliers, sensor outliers, and time block outliers, respectively. As can be seen, in all scenarios, RobTWR gives the best reconstruction of the shape and the magnitude of the time course; TWR generally recovers the pattern; $L_1L_2$ seems to miss the phase; and MNE produced fairly noisy time courses.

(a) Cell, $\sigma^2 = 0.3, 5\%$  
(b) Cell, $\sigma^2 = 0.3, 10\%$  
(c) Cell, $\sigma^2 = 1, 5\%$  
(d) Cell, $\sigma^2 = 1, 10\%$  

(e) Row, $\sigma^2 = 0.3, 5\%$  
(f) Row, $\sigma^2 = 0.3, 10\%$  
(g) Row, $\sigma^2 = 1, 5\%$  
(h) Row, $\sigma^2 = 1, 10\%$  

(i) Block, $\sigma^2 = 0.3, 5\%$  
(j) Block, $\sigma^2 = 0.3, 10\%$  
(k) Block, $\sigma^2 = 1, 5\%$  
(l) Block, $\sigma^2 = 1, 10\%$

Figure 4: Simulation 1: Comparison of Temporal Reconstruction.
To compare the spatial reconstruction, Figures 5 and 6 present the true spatial brain map at time point 7 together with the estimates obtained by the three methods from one simulation run with $\sigma^2 = 0.3$ and either 5% or 10% outliers, respectively. (The plots for $\sigma^2 = 1$ are similar and hence are omitted here.) We can make the following observations. MNE fails to identify the correct areas for all three outlier settings, and its performance gets worse as the amount of outliers increases. As for $L_1L_2$ and TWR, the time block outliers are the most challenging ones, while the sensor outliers are the second most difficult ones to deal with. Within each outlier setting, RobTWR consistently performs the best in reconstructing the spatial maps.

3.2 Simulation 2

We also considered a scenario where the noise term in the model is the pre-stimulus signal from a somatosensory task. The signal to noise ratio in this scenario is set to be 6dB. We conducted 20 simulation runs with the noise term from 20 different epochs of the somatosensory study. The signal source $B$ is generated exactly the same as in Simulation 1. Here we replot the spatial and temporal maps in Figure 7 (a) and Figure 11 (a), respectively. We apply the four methods and the results are listed in Table 2. As with the previous simulation, RobTWR obtains the smallest MSE’s and $d$’s. Figure 7 (c) shows the reconstructed time courses by the different methods. RobTWR is able to identify the major peak, while the other methods seem fail to capture the correct frequency.

Table 2: Simulation 2: Performance Comparison of Mean and Standard Error (in parentheses): MSE ($\times 10^{-5}$), MSE$_L^*$ ($\times 10^{-5}$), MSE$_R^*$ ($\times 10^{-5}$), $d_{7,L}$ ($\times 10^{-4}$), and $d_{7,R}$ ($\times 10^{-4}$).

<table>
<thead>
<tr>
<th>Method</th>
<th>MSE</th>
<th>MSE$_L^*$</th>
<th>MSE$_R^*$</th>
<th>$d_{7,L}$</th>
<th>$d_{7,R}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNE</td>
<td>1089.2(37.8)</td>
<td>802.6(31.1)</td>
<td>793.1(31.9)</td>
<td>49.2(4.5)</td>
<td>42.1(4.0)</td>
</tr>
<tr>
<td>$L_1L_2$</td>
<td>325.3(30.2)</td>
<td>312.6(22.8)</td>
<td>280.3(21.3)</td>
<td>33.2(4.1)</td>
<td>30.4(3.8)</td>
</tr>
<tr>
<td>TWR</td>
<td>215.6(12.5)</td>
<td>112.5(11.7)</td>
<td>115.6(12.3)</td>
<td>12.1(2.0)</td>
<td>12.3(1.9)</td>
</tr>
<tr>
<td>RobTWR</td>
<td>196.7(13.8)</td>
<td>106.3(11.9)</td>
<td>95.2(10.8)</td>
<td>8.9(1.2)</td>
<td>7.5(1.2)</td>
</tr>
</tbody>
</table>

Figure 11 shows the spatial reconstruction of the pre-stimulus noise scenario at time point 7. Similar to results from the previous scenarios, RobTWR is able to identify both active regions. TWR also identifies both regions but there are some scattering artifacts around the identified areas. The $L_1L_2$ method selects the rough locations of the active areas but creates a rather scattering activity effect. Solutions from MNE are too scattering.
Figure 5: Simulation 1: Comparison of Spatial Reconstruction with $\sigma^2 = 0.3$ and 5% Outliers.

### 3.3 Simulation 3

We considered a third scenario where the data were not generated from Model (5). Instead, we generated $B$ directly from some sine-exponential function shown in Figure 9 (a). This way, the source matrix may not be perfectly decomposed into a low-rank representation. We would like to
Figure 6: Simulation 1: Comparison of Spatial Reconstruction with $\sigma^2 = 0.3$ and 10% Outliers.

see how large the $q$ would be in order to ensure a good performance.

This study was also conducted 50 times. As with the previous two simulations, the two sources are still located on the left and right hemispheres, respectively. We first generated a zero matrix. Then we assigned some sine-exponential functions (shown in Figure 9 (a)) to the rows corresponding
to the active regions. The black solid curve shows the activity signal for the left source, and the red dashed curve shows the activity signal for the right source. The 12th time point is of interest, where both signals reach their peaks. In Simulation 1, we have demonstrated multiple noise setups with increasing degrees of difficulties. In this simulation, we only take the most difficult noise scenario, i.e., block noise with 10% outliers and $\sigma^2 = 1$. The synthetic measured data are shown in Figure 9 (b). We applied the four methods to this setup and the results are presented in Table 3. Similar messages are obtained as in the previous simulations. RobTWR keeps performing well while the other methods are less satisfactory.

Figure 9 (c) and (d) show the reconstructed time courses at some arbitrary locations in the left and right active regions, respectively. RobTWR is able to reconstruct both temporal signals, while the other methods can only partially reconstruct the left temporal signal but miss the one on the right hemisphere.
Table 3: Simulation 3: Performance Comparison of Mean and Standard Error (in parentheses): MSE (×10^{-5}), MSE_L^* (×10^{-5}), MSE_R^* (×10^{-5}), d_{L,L} (×10^{-4}), and d_{L,R} (×10^{-4}).

<table>
<thead>
<tr>
<th>Method</th>
<th>MSE</th>
<th>MSE_L^*</th>
<th>MSE_R^*</th>
<th>d_{L,L}</th>
<th>d_{L,R}</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNE</td>
<td>1537.2(41.8)</td>
<td>996.8(33.4)</td>
<td>1021(36.5)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>L_1 L_2</td>
<td>482.5(25.3)</td>
<td>401.0(24.1)</td>
<td>471.8(25.8)</td>
<td>39.2(4.7)</td>
<td>-</td>
</tr>
<tr>
<td>TWR</td>
<td>395.9(22.1)</td>
<td>281.3(17.1)</td>
<td>378.5(20.2)</td>
<td>30.2(4.1)</td>
<td>-</td>
</tr>
<tr>
<td>RobTWR</td>
<td>245.2(15.6)</td>
<td>132.8(13.1)</td>
<td>169.3(16.7)</td>
<td>6.5(1.5)</td>
<td>10.2(1.7)</td>
</tr>
</tbody>
</table>

(a) Truth
(b) MNE
(c) L_1 L_2
(d) TWR
(e) RobTWR

Figure 9: Simulation 3: Temporal Reconstruction on Both Hemispheres.

Figure 10 shows the reconstructed spatial maps by the different methods. RobTWR identifies both active regions, while L_1 L_2 and TWR only partially identify the left active region. MNE produces rather scattering solutions. In this simulation, the rank q that RobTWR selected is 3. This indicates that RobTWR performs well at picking a good low-rank approximation for the original signal.

(a) Truth
(b) MNE
(c) L_1 L_2
(d) TWR
(e) RobTWR

Figure 10: Simulation 3: Comparison of Spatial Reconstruction at Time Point 12.
3.4 Sensitivity of the Tuning Parameters

To examine the sensitivity of these penalty parameters, we introduced some small perturbations to the selected tuning parameters and checked the sensitivity of the results to such perturbations. We report here the results for Simulation 2. Denote by $\mu_{ij}$, $i = 1, 2, j = 1, \ldots, q$, the penalty parameter $i$ for the $j$th component. We fixed the rank number to be 2 and selected the optimal $\mu_{ij}$ using the method given in Section 2.4. The perturbations we imposed for $\mu_{1j}$ and for $\mu_{2j}$ were randomly generated from $\mathcal{N}(0, 0.01^2)$ and from $\mathcal{N}(0, 0.1^2)$, respectively. The variances of these random perturbations are chosen according to the scale of the selected $\mu_{1j}$ and $\mu_{2j}$, which is $10^{-1}$ and $10^0$ respectively.

To check the sensitivity of results to changes of $\mu_{ij}$, we imposed perturbations to $\mu_{ij}$ with the other tuning parameters fixed as the selected values. Then we ran the algorithm using the perturbed tuning parameters and recorded overall mean squared error (MSE) as defined at the beginning of Section 3. The overall MSE for the optimal tuning was 196.7 (13.8). With perturbations on $\mu_{11}$, $\mu_{21}$, $\mu_{12}$, $\mu_{22}$, the resulting MSE are 289.5 (21.5), 269.4 (20.3), 227.6 (16.8), 222.8 (15.1), respectively. The results are based on 20 simulation runs and the numbers in the parentheses are the standard errors. As we can see, perturbations do not affect the performance of the algorithm too much especially when the perturbations are on the penalty parameters for the second component. In addition, visual inspection also suggests that the reconstructed spatial maps all look similar; see figure 11.

Figure 11: Effects of Perturbation of Tuning Parameters for Simulation 2: Comparison of Spatial Maps at Time Point 7.
4 An MEG Study

In this section, we demonstrate the proposed method using a human MEG data set obtained from the Center for Clinical Neurosciences at the University of Texas Health Science Center at Houston. The study subject is a 44-year-old female patient with grade three left frontal astrocytoma who underwent the MEG test as part of the presurgical evaluation. The MEG test discussed here is a somatosensory task that is designed to non-invasively identify the somatosensory areas of the patient under evaluation, with the expectation that the primary somatosensory area contralateral to the site of activation will normally be activated.

Data collection was done with a whole-head neuromagnetometer containing 248 first-order axial gradiometers. During the MEG somatosensory session, 558 repeated epochs were delivered to the subject’s right lower lip through a pneumatically driven soft plastic diaphragm (diameter: 1 centimeter) at the pressure setting of 25 psi. Each epoch lasted 450 ms starting with a 100 ms prestimulus baseline, then a 40 ms stimulation, and finally 310 ms poststimulus period. Between two epochs there was a resting period randomized between 0.5 second and 0.6 second. The sampling frequency was 508.63 Hz and hence the MEG device recorded 228 time points in each epoch. We deleted data from one bad sensor, removed the offset, and averaged the 558 epochs to obtain the final event-related magnetic field response at each sensor. We then randomly selected 5% of the entries in $Y$, i.e., 2815 elements, and increased their values by 2 times, which introduces outliers into the data and can better demonstrate the differences among the methods. After preprocessing, the $Y$ matrix is 247-by-228 with the rows shown in Figure 12 (a). Among the 228 time points, there are two clear activity peaks at time points 85 and 99. The forward operator was obtained using the MNE software. In terms of Model (1), we have $n = 247$, $p = 15,374$, $s = 228$.

Applying the four methods to this data set, we obtained the reconstructed source time courses. Figure 12(b) shows the reconstructed time courses by the four methods for an arbitrary location in the somatosensory area where neural dynamics should be observed. Both TWR and RobTWR are able to identify the two peaks at time points 85 and 99, and the estimated time courses are smooth. MEN produces a rather noisy time course, while $L_1L_2$ only identifies activity around time point 85. Figure 12(c) plots the reconstructed time courses at one arbitrary location outside the somatosensory area. The TWR estimate clearly shows activity that is not supposed to present. In contrast, RobTWR and $L_1L_2$ give clean inactive reconstructions. MNE still provides a rather noisy
reconstruction. Figure 13 shows the sideview of the 3-D spatial maps at time point 85 (indicated by the vertical line in Figure 12(a)) by the four methods. The reason we chose the time point 85 is that this point is the first peak after electromyographic (EMG) onset and the brain spatial map should have a big contrast. As we can see, MNE and $L_1L_2$ can not find the accurate location of the somatosensory area, while TWR produces some artifacts around the somatosensory area and results in false positives, which can be confirmed by Figure 12(c). RobTWR clearly identifies the expected area. In this study, the optimal rank $q$ chosen by RobTWR and TWR were 5 and 6, respectively.

Figure 12: Comparison of Results from An MEG Study.

Figure 13: Reconstructed Brain Maps at Time Point 85.
Acknowledgement

This work is supported in part by NIDA (1 RC1 DA029425-01), NSF (DMS-09-07170, DMS-10-07618, CMMI-0800575, DMS-11-06912, DMS-12-08952, and DMS-12-08786), and King Abdullah University of Science and Technology (KUSCI-016-04).

References


