A STATISTICAL APPROACH TO THE INVERSE PROBLEM IN MAGNETOENCEPHALOGRAPHY

by

Zhigang Yao

B.S. Zhejiang University City College, China 2006

M.A. University of Pittsburgh, Pittsburgh, PA 2008

Submitted to the Graduate Faculty of
the Arts & Sciences in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2011

UNIVERSITY OF PITTSBURGH

ARTS & SCIENCES

This dissertation was presented

by

Zhigang Yao

It was defended on

June 1, 2011

and approved by

Leon J. Gleser, Professor, Statistics, Chair

William F. Eddy, Professor, Statistics, Co-Chair (Carnegie Mellon University)

Satish Iyengar, Professor, Statistics

Robert T. Krafty, Assistant Professor, Statistics

Dissertation Advisors: **Leon J. Gleser**, Professor, Statistics, Chair,

William F. Eddy, Professor, Statistics, Co-Chair (Carnegie Mellon University)

A STATISTICAL APPROACH TO THE INVERSE PROBLEM IN MAGNETOENCEPHALOGRAPHY

Zhigang Yao, PhD

University of Pittsburgh, 2011

Magnetoencephalography (MEG) is an imaging technique used to measure the magnetic field outside the human head produced by the electrical activity inside the brain. The MEG inverse problem, identifying the location of the electric sources from the magnetic signal measurements, is ill-posed; that is, there is an infinite number of mathematically correct solutions. Common source localization methods assume the source does not vary with time and do not provide estimates of the variability of the fitted model. We reformulate the MEG inverse problem by considering time-varying sources and we model their time evolution using a state space model. Based on our model, we investigate the inverse problem by finding the posterior source distribution given the multiple channels of observations at each time rather than fitting fixed source estimates. A computational challenge arises because the data likelihood is nonlinear, where Markov chain Monte Carlo (MCMC) methods including conventional Gibbs sampling are difficult to implement. We propose two new Monte Carlo methods based on sequential importance sampling. Unlike the usual MCMC sampling scheme, our new methods work in this situation without needing to tune a high-dimensional transition kernel which has a very high-cost. We have created a set of C programs under LINUX and use Parallel Virtual Machine (PVM) software to speed up the computation.

Common methods used to estimate the number of sources in the MEG data include principal component analysis and factor analysis, both of which make use of the eigenvalue distribution of the data. Other methods involve the information criterion and minimum description length. Unfortunately, all these methods are very sensitive to the signal-to-noise ratio (SNR). First, we consider a wavelet approach, a residual analysis approach and a Fourier approach to estimate the noise variance. Second, a Neyman-Pearson detection theory-based eigenthresholding method is used to decide the number of signal sources. We apply our methods to simulated data where we know the truth. A real MEG dataset without a human subject is also tested. Our methods allow us to estimate the noise more accurately and are robust in deciding the number of signal sources.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

PREFACE

Though my name is printed on the cover of this thesis, the word "I" does not appear within its chapters. I do this to pay tribute to the myriad contributions of my advisors and collaborators, and the support of my family and friends.

Bill, your dedication, encouragement, and support constantly inspires me to better myself and aim higher. For the past four years working with you, I have been interested in many different kinds of problems, and I have learned how to choose problems to work. My dissertation would not have happened without your supervision. Thank you for giving me the freedom to work on my thesis, and this at the same time makes me an independent researcher.

Leon, thank you for believing in me and creating for me quite a flexible research environment which enables me to pursue my own interests. I have always remembered the words, "Faint heart never won fair maiden," which you told me when I was a fresh graduate, and this somehow proved itself encouraging both in my academic pursuit and social life.

Rob, I would like to thank you for giving me suggestions not only on research but also on a career path.

Jiashun, thanks for giving me an opportunity to work with you on a new research field, which was valuable in my early career.

CMU "remarks", thank you for providing assistance on all sorts of computing problems that I had in my programming.

Thanks to funding agency: this research was supported in part by the Multi-Modal Neuro-Imaging Program Fellowship at the Center for the Neural Basis of Cognition at University of Pittsburgh and Carnegie Mellon under grants NIH/NIDA R90 DA023420 and Doctoral Dissertation Improvement Grants under NSF SES-1061387 from the National Sci-

ence Foundation.

Many of my fellow students and collaborators took time away from their own overextended lives to provide feedback on this thesis, and for this I am extremely grateful. I hesitate to include such a list, as I will surely omit some who deserve mention.

Finally, I want to thank all of my friends who made life in Pittsburgh enjoyable and sociable.

All errors and limitations remaining in this thesis are mine alone.

1.0 INTRODUCTION

1.1 THE BASICS OF MEG

Exploration of the human brain is of fundamental interest. Although the anatomy of the brain has been studied intensively for millennia, how the brain functions is still not well understood; in particular, how the physical functioning of the brain as an organ gives rise to the thinking of the mind remains a complete mystery. The neurons in the brain produce macroscopic electric currents when the brain functions, and those synchronized neuronal currents in the gray matter of the brain induce extremely weak magnetic fields ($10 - 100$ femtoTesla) outside the head. The comparatively recent development of Superconducting Quantum Interference Devices (SQUIDs) makes it possible to detect those magnetic signals. MEG is an imaging technique using SQUIDs to measure the magnetic signals outside of the head produced by the electrical activity inside the brain [24]. Due to its noninvasiveness (it is a completely passive measurement method) and its impressive temporal resolution (better than 1 millisecond, compared to 1 second for functional magnetic resonance imaging, or to 1 minute for positron emission tomography) (see Figure 1) and due to the fact that the signal it measures is a direct consequence of neural activity, MEG is a near optimal tool for studying brain activity in both the research and the clinical setting. The computation associated with estimating the electric source from the magnetic measurement is a challenging problem that needs to be solved to allow high temporal and spatial resolution imaging of the dynamic activity of the human brain.

Because of its ability in revealing the precise dynamic of neuronal activities, MEG has started to move toward clinical applications such as presurgical planning for epileptic patients [42]. However, the full potentiality has not been exploited due to the difficulties of the MEG data analysis. The main problem is the ill-posed neuromagnetic inverse problem; that is, estimating neuronal current flow from magnetic field measurements has no unique solution [47]. The ill-posedness directly results in the instability of the solution. Second, the magnetic signals from the brain are extremely weak, i.e., nine orders less than Earth's magnetic field; this means the MEG recording contain not only a magnetic field associated with the signal sources of interest but also interference magnetic fields generated from non-target activities. Such non-target activities include spontaneous brain activities or some evoked activities that are not the focus of the current investigation. Third, the temporal analysis of the data has not been extensively investigated due to computational ineffectiveness. The MEG data is characterized by a very low signal-to-noise ratio (SNR); the commonly used method to improve the SNR is simply to average over the data in time. However, this ignores the high temporal resolution that MEG offers and prevents the possibility of discovering the dynamics of the underlying neuronal current. The complexity of studying the inverse problem still exists and the computional challenge associated with it still needs to be solved.

There are three key steps to any source localization algorithm in MEG. First, define the solution space and the parameter space of the signal source in MEG. Second, calculate the magnetic field given the information about the head model. Third, according to what criterion the solution must satisfy, perform a search for the solution iteratively which automatically requires the same amount of forward model calculation. Methods of finding the solution of the neuronal current from the observed MEG signal have been extensively exploited during the past two decades. Rather than working with continuous neuronal current, one type of method assumes that the current can be thought of an electric dipole; this model is called equivalent current dipole (ECD). From the perspective of ECD, a dipole has its location, orientation, and magnitude and the magnetic field generated by this dipole can explain the MEG measurement. In addition, there is a version of ECD assuming multiple dipoles [52]. Such an ECD models a large number of dipoles located at fixed places over the cortical surface. In neuroscience, it is believed that the MEG data should be explained by only a few dipoles (less than 10), and different criteria or algorithms are made to shrink the number of dipoles in various ECD models. These criteria include L_1-norm [74, 93, 96] , L_2-norm [42], or L_1L_2-norm [70]. Other algorithms from spatial filters or array signal processing field are used with the application to the MEG inverse problem. There algorithms invlove multiple signal classification [68] and beamforers [94, 95, 98, 39, 85]. Fitting ECDs requires solving nonconvex optimization problems which often leed to a nonstable solution. The other type of method used to solve the MEG inverse problems is called distributed model or Bayesian model, where high-level knowledge of the dipole is considered when doing the dipole fitting. Such high level information can be the anatomy of a subject, physiological or functional information and other prior information concerning the source [27, 73, 81]. The existing Bayesian model [10, 83, 12] plays a role in furnishing the unique solution by imposing extra constraints or inverse criteria. However, the "distributed" term has not been fully utilized in the sense of the meaning itself. Finding the distribution of the source in space and (particularly) in time is still a problem requiring investigation.

1.2 A TIME-VARYING SOURCE MODEL FOR THE MEG INVERSE PROBLEM WITH PARALLEL COMPUTING

We motivate the development of a time-varying source model for the MEG inverse problem. Rather than attempting to "solve" the inverse problem, we try to develop estimates of the dipole parameters using an inherently spatio-temporal model. Throughout this thesis, we are not interested in developing an algorithm for finding an unique solution for any dipole. Instead, we present a statistical framework to the inverse problem in MEG; that is, solving the model allows us to provide the distribution of the dipoles' parameters. This naturally comes up with a time-varying dipole model, where the dipole at each time point is assumed not fixed . Although we use the term "time-varying" which seems to only refer to the temporal resolution of the dipole, we consider the spatial resolution as well. We parameterize each dipole using both the electric moments and spatial location for each time point; the time dependencies for the parameters are modelled by a state-space model. The goal of interest is to find the joint distribution of these parameters at each time. Based on our predictive model, we investigate the inverse problem by finding the posterior source distribution given the multiple channels of observations at each time rather than fitting fixed source estimates.

This new model faces the following statistical challenge: the parameter spaces is greatly expanded by the new parameterization. The dipole parameters increase with the number of time points included. The joint distribution of interest inevitably becomes very high dimensional as the number of time points increases without bound. In addition, the data likelihood is nonlinear (the model is nonlinear). The regular MCMC methods, including conventional Gibbs sampling, suffer from being difficult to implement and extremely slow to converge. This makes it difficult to find the joint distribution, even if for a single dipole. We propose two new Monte Carlo methods based on sequential importance sampling. Unlike the usual MCMC sampling scheme, our new methods work in this situation without a very high cost tuning of the high-dimensional transition kernel. The benefit of sequential methods is that we do not attempt to estimate the entire target distribution at once, but rather attempt to estimate samples for each time point sequentially. To assess the performance of our proposed method, we also do simulation studies. In particular, we study our method's ability to sample from a high dimensional distribution and compare our method with other methods such as MCMC+Gibbs sampling or Hybrid MCMC+Gibbs. The simulations help give credibility to the use of sequential methods to investigate the time-varying model. Guided by the simulation study, we implement our proposed methods on the real MEG data sets.

Our interest is also in the context of implementing our proposed algorithms for long time. Because of the expanded parameter space, there is an obvious need for parallel computational methods. Our initial attempt utilized PVM and provided the expected reduction in running time. We have software that runs our sequential methods for data of up to 5000 milliseconds. The common MEG dataset we have from an experiment is very large (e.g., hundreds of thousands of milliseconds or more), and no regular computing facility that can help. A natural extension of running our algorithms is through a highly parallel computing scheme. We are exploring the use of more advanced forms of parallelism such as CUDA and OPENCL to further reduce the running time.

1.3 STATISTICAL APPROACHES TO ESTIMATING THE NUMBER OF SIGNAL SOURCES IN MEG

The source localization method to the MEG inverse problem that was mentioned in the previous section assume the number of sources is known. In most cases, the number of signal sources in MEG is predefined or chosen from some prior distribution. However, in practice, the number of sources is often not known. Estimating the number of electric sources in the MEG data is not easy. Common methods include use of principal component analysis (PCA) [75], independent component analysis (ICA) [37] and factor analysis [62, 63, 22], all of which make use of the eigenvalue distribution of the data covariance matrix to estimate the number of sources. Other methods use the information criterion such as Akaike information criterion (AIC) [99, 100, 7, 8, 56, 101] and minimum description length (MDL) [40] as criterion for choosing a solution.

The development of hyperspectral imaging in remote sensing and geographic information suggests an alternative way to decide the number of signal sources in hyperdimensional data. Hyperdimensional data (or spectra) can be thought of as points in n-dimensional space. The data for a given pixel corresponds to a spectral reflectance for that given pixel. The distribution of the hyperspectral data in n-space can be used to estimate the number of spectral endmembers and their pure spectral signatures and to help understand the spectral characteristics of the materials which make up that signature. The MEG data can be thought as an analog of hyperspectral data where each channel corresponds to a frequency band in the spectrum. Unfortunately, all of the methods above are equivalent to identifying the intrinsic data dimensionality rather than the number of clusters constituted by distinct sources, and therefore they become quite sensitive if the signal-to-noise ratio is relatively small. Furthermore, they are not very useful for hyperdimensional image datasets with hundreds of channels, or more.

We consider the virtual dimensionality concept for MEG data and consider a wavelet approach, a residual analysis approach, and a Fourier approach to estimate the noise at each sensor of the data. A Neyman-Pearson detection theory-based eigenthresholding method is used to decide the number of signal sources in the data. To assess performance of our methods, we apply them to simulated data where we vary the number of sources and SNRs and also compare our methods with other methods. A real MEG dataset collected in a special room without a human subject is also tested. Our methods allow us to estimate the noise more accurately for MEG data and are robust in deciding the number of signal sources.

1.4 OTHER ISSUES

In the application of the proposed time-varying source model to the inverse problem in MEG, some other important statistical issues are of concern in both theory and computation.

First, our results so far were mainly based on a one-source model where we assume there was only one dipole in the MEG data. We are still developing a multiple-source model for the MEG inverse problem. The extension from one source to having multiple sources is natural and only the computational complexity increases. Our algorithms will still work in this case.

Second, in both the time-varying source model for the MEG inverse problem and the approaches to estimating the number of signal sources, we assume the noise from each sensor is normally distributed. We also assume the distinct sources in the brain act independently. Such assumptions statistically simplify our analysis. Unfortunately in practice the data is always far from normal and the noise is correlated.

Third, while implementing our algorithms, in order to focus on the source parameters we fixed several parameters (source noise parameters, measurement noise parameters, etc.) in the model. In fact, those parameters could be estimated along with the source distribution. The natural way of implementing this is to iterate estimates of those parameters and of the source distribution until all of them converge. Furthermore, the skewness of weights that arise in the sequential importance sampling could be a tradeoff between the efficiency of the program and the quality of the source distribution. Residual sampling can be used to replace regular weight sampling.

Fourth, previous results show that PVM did improve the speed of calculating the source distribution by computing in parallel. Since our PVM program involves randomness and a resampling scheme, several issues from our PVM implementation still need to be resolved: 1) If our algorithm were implemented in a single program without parallelism, all samples generated before resampling from this program should be simply related to the random number generator. However, when there were several worker programs with each of them doing the same thing as a single program but in parallel, the unique randomness within each worker program will eventually come up with different but similar samples before resampling. 2) In a single program without parallelism, we would only have one resampling procedure. The samples would be generated from the resampling procudure. However, there was one resampling procedure within each of our worker programs in PVM. The samples were generated from each of these workers and should eventually be pooled together. 3) There is always a tradeoff between resampling in parallel or not. We will address these issues in the PVM section.

1.5 ORGANIZATION OF THE DISSERTATION

The dissertation mainly consists of three parts:

1. A time-varying source model for the MEG inverse problem with parallel computing;
2. Statistical approaches to estimating the number of signal sources in MEG;
3. Future work: real-time analysis of the MEG data.

In Chapter 2, we introduce the forward and inverse problem in MEG (Section 2.1), the general statistical framework of the MEG inverse problem (Section 2.2), and the related source localization methods (Section 2.3). In Chapter 3, we describe a probabilistic time-varying source model for the MEG inverse problem (Section 3.1). Due to the difficulty of using Markov chain Monte Carlo (MCMC) methods for generating samples from the time-varying model, our algorithms based on sequential importance sampling will be proposed (Section 3.2). Simulation studies of comparing our methods with other methods are presented in Section 3.3. A PVM application to speed up the computations follows in Section 3.4. At the end of Chapter 3 are the real data analysis (Section 3.5) and discussion (Section 3.6).

In Chapter 4, we describe spectral signatures in hyperspectral data and summarize previous work on estimating the number of signal sources in the EEG/MEG field based on intrinsic dimensionality (Section 4.1). Three methods (a wavelet, a residual analysis and a Fourier framework) for noise estimation in multi-channel data are presented (Section 4.2) followed by the introduction of the virtual dimensionality concept for hyperspectral imagery. A simulation study (Section 4.3) and a real data analysis (Section 4.4) are described. Discussion is in Section 4.5. Finally, in Chapter 5, promised future work about the real-time MEG imaging (Section 5.1), an ongoing NSF project (Section 5.2) and the dissemination of our research (Section 5.3) are briefly sketched.

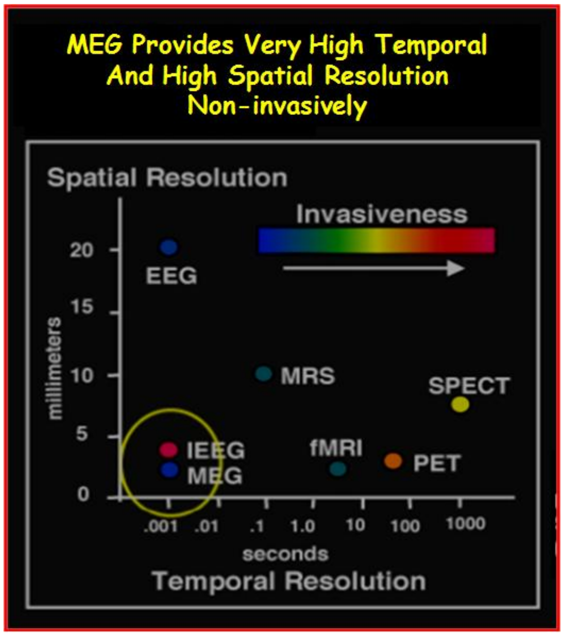


Figure 1: Temporal and spatial resolution of each brain imaging technique. The color bar displays the invasiveness of each imaging technique.

2.0 BACKGROUND ON MEG AND RELATED WORK

2.1 FORWARD AND INVERSE MEG PROBLEM

The MEG signals derive from the *primary* current (the net effect of ionic currents flowing in the dendrites of neurons) and the *volume* current (that is, the additive ohmic current set up in the surrounding medium to complete the electric circuit) (see Figure 2). If the electric source is known and the head model [57] is specified (e.g., a sphere with homogeneous conductivity), then the “forward problem” is to compute the electric field **E** and the magnetic field **B** from the source current **J**. The calculation uses Maxwell's equations, see, e.g., [38],

$$\begin{aligned}
\nabla \cdot \mathbf{E} &= \rho/\epsilon_0 \\
\nabla \times \mathbf{E} &= -\partial \mathbf{B}/\partial t \\
\nabla \cdot \mathbf{B} &= 0 \\
\nabla \times \mathbf{B} &= \mu_0(\mathbf{J} + \epsilon_0 \partial \mathbf{E}/\partial t)
\end{aligned}$$

where ϵ_0 and μ_0 are the permittivity and permeability of a vacuum, respectively, and ρ is the charge density. The total current **J** consists of the primary current $\mathbf{J}^P$ plus the volume current $\mathbf{J}^V$. The source activity in the brain corresponds to the primary current. Under reasonable assumptions, see [42], the volume current $\mathbf{J}^V$ is not included in the analysis because of its diffuse nature. The terms $\partial\mathbf{B}/\partial t$ and $\partial\mathbf{E}/\partial t$ in Maxwell's equations can be ignored by assuming that the magnetic field varies relatively slowly in time. We assume that **E** is generated by $\mathbf{J}^P$ which comes from the sum of N localized current dipoles at locations $\mathbf{r}_n$

$$\mathbf{J}_n^P(\mathbf{r}) = Q_n\delta(\mathbf{r} - \mathbf{r}_n), \;\; n = 1, \ldots, N \tag{2.1}$$

where $\delta(\cdot)$ is the Dirac delta function. The Q_n is a charged dipole at the point $\mathbf{r}_n$ in the brain volume Ω. Using the quasi-static approximation to Maxwell's equations (that is, ignoring the partial derivatives with respect to time) in [79], the magnetic field $\mathbf{B}$ at location $\mathbf{r}$ of a current dipole at $\mathbf{r}_n$ can be calculated by the Biot-Savart equation,

$$\mathbf{B}(\mathbf{r}) = \frac{\mu_0}{4\pi} \int_{\Omega} \frac{\mathbf{J}^P(\mathbf{r}_n) \times (\mathbf{r} - \mathbf{r}_n)}{|\mathbf{r} - \mathbf{r}_n|^3} d\mathbf{r}_n. \tag{2.2}$$

In the case of multiple current dipoles, the induced magnetic fields simply add.

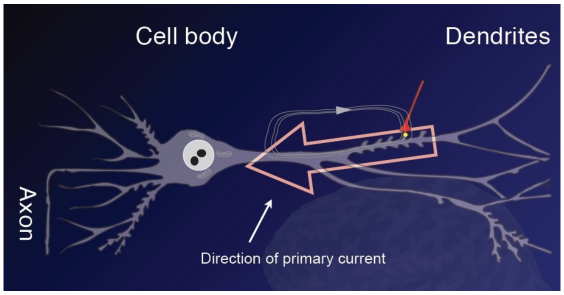


Figure 2: Primary current and volumn current.

The "inverse problem" comes from the forward model; we want to estimate the dipole parameters from the observed magnetic signal. The difficulty is that there is not a unique solution; there are infinitely many different sources within the skull that produce the same observed data (see [47]). The goal is to find a meaningful solution among the many mathematically correct solutions.

2.2 GENERAL FRAMEWORK OF THE MEG INVERSE PROBLEM

In a typical MEG scanner, the magnetic field $\mathbf{B}$ is sampled on a finite number L of sensors, each one measuring one component (z direction) of the magnetic field, namely $\mathbf{B}_z$ (see the arrow in Figure 3); if $\mathbf{e} = (0, 0, 1)$, a unit vector, is used to find $\mathbf{B}_z$, the z component of $\mathbf{B}$ can be obained by $\mathbf{B}_z = \mathbf{B} \cdot \mathbf{e}$. Nevertheless, for simplicity, we will ignore the superscript z in

$\mathbf{B}_z$ from now on. With the Biot-Savart equation, mathematically the MEG forward model based on (lead-field) can be written as,

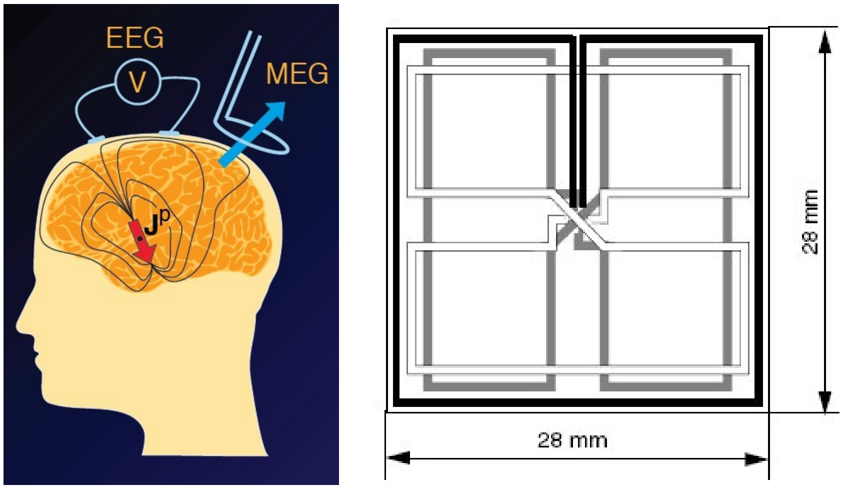
$$\mathbf{B}(\mathbf{r}) = \sum_{n=1}^{N} g(\mathbf{r}, \mathbf{r}_n) \cdot Q_n \tag{2.3}$$

where

$$g(\mathbf{r}, \mathbf{r}_n) = \frac{\mu_0}{4\pi F^2(\mathbf{r})} \mathbf{r}_n \times [F(\mathbf{r})\mathbf{e} - (\nabla F(\mathbf{r}) \cdot \mathbf{e})\mathbf{r}]$$

is the lead-field vector and

$$F(\mathbf{r}) = |\mathbf{r} - \mathbf{r}_n| \left(|\mathbf{r}| \, |\mathbf{r} - \mathbf{r}_n| + |\mathbf{r}|^2 - \mathbf{r}_n \cdot \mathbf{r} \right).$$



(a)

(b)

Figure 3: Left: perpendicular direction (blue arrow) of the magnetic field is observed. Right: black loop (magnetometer); grey and white loops (two gradiometers)

Therefore, the general framework of the MEG inverse problem has the follwing form

$$\mathbf{Y} = \mathbf{G}\mathbf{Q} + \mathbf{U} \tag{2.4}$$

where $\mathbf{Y}$ is the $L \times T$ matrix with each entry $Y_{k,t}$ representing the observed magnetic field at the k^{th} sensor at time t, $1 \leq k \leq L; 1 \leq t \leq T$. $\mathbf{G}$ is the $L \times 3N$ matrix constructed by the

N $L \times 3$ sub-block matrices corresponding to the three components of the above lead-field vector. The columns of $\mathbf{G}$ describe the measurements observed across sensors, induced by a parciluar dipole. The k^{th} row of $\mathbf{G}$ describe the flow of current for a given sensor through N dipoles with each one at location $\mathbf{r}_n$, $1 \leq n \leq N$. The $\mathbf{Q}$ is the $3N \times T$ matrix constructed by the N $3 \times L$ sub-block matrices associated to the three components (moments) of the current dipole $\mathbf{J}_n^P$. The $\mathbf{U}$ is the $L \times T$ matrix with each entry $(U_{k,t})$ associated the additive observation noise.

2.3 EXISTING SOURCE LOCALIZATION METHODS

2.3.1 Classical Approach: Minimum Norm Estimates

A well-known approach to the MEG inverse problem is the minimum norm estimate (MNE) [42] which recovers source parameters with minimum overall energy (or minimum L_2-norm). This method minimizes the quadratic energy function

$$\underset{\mathbf{Q}}{\operatorname{argmin}} \left\|\mathbf{Y} - \mathbf{GQ}\right\|^2 + \lambda \left\|\mathbf{Q}\right\|^2 \tag{2.5}$$

where $\|\cdot\|$ denotes the Frobenius norm of a matrix. To be specific, let $\mathbf{A} = [a_{ij}]_{m \times n}$ be a matrix with m rows and n columns, then $\|\mathbf{A}\| = \left(\sum_{i=1}^m \sum_{j=1}^n a_{ij}^2\right)^{1/2} = \sqrt{\operatorname{tr}(\mathbf{A}^T\mathbf{A})}$. The tuning parameter λ controls the regularization strength. An extended framework for the MNE is the weighted minimum norm estimate (WMNE) [91, 65]. It minimizes

$$\underset{\mathbf{Q}}{\operatorname{argmin}} \left\|\mathbf{Y} - \mathbf{GQ}\right\|^2_{\mathbf{W}_1} + \lambda \left\|\mathbf{Q}\right\|^2_{\mathbf{W}_2} \tag{2.6}$$

where $\|\cdot\|_{\mathbf{W}}$ indicates the Frobenius norm of a matrix associated with metric $\mathbf{W}$. Specifically, $\|\mathbf{A}\|_{\mathbf{W}} = \sqrt{\operatorname{tr}(\mathbf{A}^T\mathbf{W}\mathbf{A})}$; $\mathbf{W}_1$ and $\mathbf{W}_2$ are the two weight matrices (commonly diagonal matrices) that are associated with the two terms in the minimization, respectively.

The solution of the minimization problem above can be expressed as

$$\begin{aligned} \hat{\mathbf{Q}} &= \left[\mathbf{G}^T(\mathbf{W}_1^T\mathbf{W}_1)\mathbf{G} + \lambda(\mathbf{W}_2^T\mathbf{W}_2)\right]^{-1}\mathbf{G}^T(\mathbf{W}_1^T\mathbf{W}_1)\mathbf{Y} \\ &= (\mathbf{W}_2^T\mathbf{W}_2)^{-1}\mathbf{G}^T\left[\mathbf{G}(\mathbf{W}_2^T\mathbf{W}_2)^{-1}\mathbf{G}^T + \lambda(\mathbf{W}_1^T\mathbf{W}_1)^{-1}\right]^{-1}\mathbf{Y}. \end{aligned} \tag{2.7}$$

This formulation shows that the MNE solution is the special case of the WMNE where the diagonal weight matrix $\mathbf{W}_1$ is an identity $\mathbf{I}_1$ (Frobenius norm) and the diagonal weight matrix $\mathbf{W}_2$ satisfies $\lambda \mathbf{W}_2^{-1} \times \mathbf{W}_2 = \mathbf{I}_2$. The LORETA approach [71] is another special case of the WMNE in which the $\mathbf{W}_2$ is equal to a spatial Laplacian operator [82]. Although the L_2-norm method leads to an efficient linear solution to the MEG inverse problem, it is often too diffuse; in other words, the MNE (or WMNE) estimates are typically spread out spatially, which turns out a large number of dipoles are active. This drawback makes the solutions of the method contradictive in some circumstances where there are only a few well-localized dipoles appearing in the brain.

2.3.2 Minimum Current Estimates

An alternative solution that can provide sparse estimates of the dipole is to minimize the same quadratic energy function but penalize on L_1-norm

$$\underset{\mathbf{Q}}{\operatorname{argmin}} \|\mathbf{Y} - \mathbf{G}\mathbf{Q}\|^2 + \lambda |\mathbf{Q}| \tag{2.8}$$

where $|\cdot|$ denotes the a matrix version of L_1-norm. Following the notation used in Section 2.3.1, we now have $|\mathbf{A}| = \sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|$. The L_1 estimates are called the minimum current estimates (MCE) [74, 93, 96]. Unlike the solutions based on L_2-norm regularization, the L_1-norm solutions cannot be computed in closed form; instead, they need to be obtained by a nonlinear minimization procedure. The conventional method is to search for the solution through linear programming (LP). From the view of computation and accuracy, there exist two types of methods in the context of finding L_1 solution over the past decades: 1) gradient-based methods; 2) methods based on path algorithms. The gradient-based methods, like Newton-Raphson method, cannot be applied directly. On numerical optimization, Tibshirani [90] offered an algorithm where the regularization term was seen as a combination of linear constraints; however, it was proven to be computationally inefficient, because the L_1 term implies a large number linear constraints. Methods based on path algorithms (e.g., [30]) improved the computation time and the accuracy of the estimates. The coordinate descent method (e.g., [31]) is one of the methods that computes the estimate efficiently and largely improves the accuracy of the estimates. The MCE leads to more focal source estimates than estimates using MNE and can represent well the relatively compact source areas typically activated in the sensory projection areas.

There is also a weighted version of MCE:

$$\underset{\mathbf{Q}}{\operatorname{argmin}} \|\mathbf{Y} - \mathbf{GQ}\|^2_{\mathbf{W}_1} + \lambda|\mathbf{Q}|_{\mathbf{W}_2} \tag{2.9}$$

where $|\cdot|_{\mathbf{W}}$ is weighted version of $|\cdot|$; to be specific, $|\mathbf{A}|_{\mathbf{W}} = |\mathbf{WA}|$. $\mathbf{W}_1$ and $\mathbf{W}_2$ are the two weight matrices defined in the Section 2.3.1. Similarly, this can be solved by the weighted L_1 method where the coordinate descent algorithm can be embedded within each iteration of fitting weighted linear regression problems [104]. MCE could be thought of as the maximum of a posterior distribution corresponding to an exponential prior distribution. Because of the columnar organization of the cortex, the observable sources are typically perpendicular to the cortical surface. Magnetic resonance imaging (MRI) can be used to determine the normal of the cortex at given points and thus also the most probable current orientation [27, 10]. Since the cortex is heavily convoluted, a large number of points are required to represent its geometry accurately. However, the use of a dense point set may be unnecessary because of the relatively poor spatial resolution of MEG.

2.3.3 Multiple Signal Classification

The multiple signal classification (MUSIC) was first developed in the array signal processing community [84]. With the application of this method to the MEG field, one of the earliest works can be found in [68] where the MUSIC formulation is considered in the nonlinear framework of the MEG inverse problem. The motivation of MUSIC is that: due to the organization of the brain and the neuroscience perspective of activity in the brain, the primary current $\mathbf{J}^p$ usually concentrates in one or a few regions; in other words, it is reasonable to believe that the magnetic fields that are observed are produced by a very small number of dipoles (< 10). The MUSIC approach does not require testing all possible dipole orientations at each location; instead, it needs to solve a generalized eigenvalue problem whose solution gives us the estimates of the dipole parameters. This eases the difficulty of optimizing the energy function with respect to the locations of the dipoles by either MNE or MCE, which often become trapped in local minima, yielding significant localization errors. We reconsider the model (see Section 2.2) by assuming there are only N unique dipoles

$$\begin{aligned}\mathbf{Y} &= \begin{bmatrix} \mathbf{Y}_1, \cdots, \mathbf{Y}_T \end{bmatrix} = \mathbf{GQ} + \mathbf{U} \\ &= \begin{bmatrix} g(\mathbf{r}_1^{'}, \mathbf{r}_1) & \cdots & g(\mathbf{r}_1^{'}, \mathbf{r}_N) \\ \vdots & \ddots & \vdots \\ g(\mathbf{r}_L^{'}, \mathbf{r}_1) & \cdots & g(\mathbf{r}_L^{'}, \mathbf{r}_N) \end{bmatrix} \begin{bmatrix} \mathbf{q}_1(1) & \cdots & \mathbf{q}_1(T) \\ \vdots & \ddots & \vdots \\ \mathbf{q}_N(1) & \cdots & \mathbf{q}_N(T) \end{bmatrix} + \mathbf{U} \end{aligned} \tag{2.10}$$

where $\mathbf{Y}_t = (Y_{1,t}, \cdots, Y_{L,t})^T$, $1 \leq t \leq T$. To avoid confusion, we rename the sensor location $\mathbf{r}_i$ by $\mathbf{r}_i^{'}$ $(1 \leq i \leq L)$; we still use $\mathbf{r}_j$ $(1 \leq j \leq N)$ as dipole location, and rewrite a new matrix $\mathbf{Q}$ as

$$\mathbf{Q} = \begin{bmatrix} \mathbf{q}_1(1) & \cdots & \mathbf{q}_1(T) \\ \vdots & \ddots & \vdots \\ \mathbf{q}_N(1) & \cdots & \mathbf{q}_N(T) \end{bmatrix} = \begin{bmatrix} \mathbf{o}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{o}_N \end{bmatrix} \begin{bmatrix} s_1(1) & \cdots & s_1(T) \\ \vdots & \ddots & \vdots \\ s_N(1) & \cdots & s_N(T) \end{bmatrix},$$

where each entry $\mathbf{q}_j(t) = \mathbf{o}_j s_j(t)$ and $\mathbf{o}_j$ is a unit norm orientation vector of dimension 3. The $s_j(t)$ scalar time series is a linear parameter of the j^{th} dipole at time t, $1 \leq t \leq T$. Notice the corresponding dipole locations $\mathbf{r}_j$ $(1 \leq j \leq N)$ are the nonlinear parameters, and the dipole orientations $\mathbf{o}_j$ $(1 \leq i \leq N)$ are the quasilinear parameters. For convenience, we rewrite the model in terms of $\mathbf{A}$ and $\mathbf{S}$ as

$$\mathbf{Y} = \mathbf{AS} + \mathbf{U} \tag{2.11}$$

where

$$\mathbf{A} = \mathbf{G} \begin{bmatrix} \mathbf{o}_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{o}_N \end{bmatrix} \quad \text{and} \quad \mathbf{S} = \begin{bmatrix} s_1(1) & \cdots & s_1(T) \\ \vdots & \ddots & \vdots \\ s_N(1) & \cdots & s_N(T) \end{bmatrix}.$$

If we assume $\mathbf{ASS}^T\mathbf{A}^T$ is rank r, then from the singular value decomposition (SVD) point of view, we have

$$\begin{aligned}
\mathbf{ASS}^T\mathbf{A}^T &= \mathbf{\Phi\Lambda}_L\mathbf{\Phi}^T \\
&= [\mathbf{\Phi}_s \quad \mathbf{\Phi}_n]\begin{bmatrix} \mathbf{\Lambda}_s & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda}_n \end{bmatrix}[\mathbf{\Phi}_s \quad \mathbf{\Phi}_n]^T \\
&= \mathbf{\Phi}_s\mathbf{\Lambda}_s\mathbf{\Phi}_s^T
\end{aligned} \tag{2.12}$$

where $\mathbf{\Lambda}_L$ is the $L \times L$ diagonal matrix of the eigenvalues of $\mathbf{ASS}^T\mathbf{A}^T$ and $\mathbf{\Phi}$ is the corresponding matrix of eigenvectors; specifically, there are r non-zero and $L-r$ zero eigenvalues in the $\mathbf{\Lambda}_L$. Furthermore, we can write $\mathbf{\Phi} = (\mathbf{\Phi}_s, \mathbf{\Phi}_n)$ where $\mathbf{\Lambda}_s$ is defined as the diagonal matrix containing the r nonzero eigenvalues and $\mathbf{\Phi}_s$ as the matrix containing the corresponding eigenvectors (signal space); similarly, $\mathbf{\Lambda}_n$ ($\mathbf{0}$ matrix) is defined as the diagonal matrix containing the $L-r$ zero eigenvalues and $\mathbf{\Phi}_n$ as the matrix containing the corresponding eigenvectors (noise space). We also have the other diagonalization

$$\mathbf{SS}^T = \mathbf{\Gamma\Lambda}_r\mathbf{\Gamma}^T \tag{2.13}$$

where $\mathbf{\Lambda}_r$ is the $r \times r$ diagonal matrix of the r non-zero eigenvalues of $\mathbf{SS}^T$ and $\mathbf{\Gamma}$ is the $N \times r$ matrix containing the corresponding eigenvectors. Based on the two diagonalizations, the result of MUSIC for MEG is that

$$r(\mathbf{A\Gamma}) = r(\mathbf{\Phi}_s) \tag{2.14}$$

where $r(\cdot)$ is the rank of a matrix. We observe under the white noise assumption (i.e., $E(\mathbf{u}_t\mathbf{u}_t^T) = \sigma^2\mathbf{I}$),

$$\begin{aligned}
E(\mathbf{YY}^T) &= \mathbf{ASS}^T\mathbf{A}^T + \sum_{t=1}^{T} E(\mathbf{u}_t\mathbf{u}_t^T) \\
&= \mathbf{ASS}^T\mathbf{A}^T + T\sigma^2\mathbf{I} \\
&= [\mathbf{\Phi}_s \quad \mathbf{\Phi}_n]\begin{bmatrix} \mathbf{\Lambda}_s + T\sigma^2\mathbf{I} & \mathbf{0} \\ \mathbf{0} & T\sigma^2\mathbf{I} \end{bmatrix}[\mathbf{\Phi}_s \quad \mathbf{\Phi}_n]^T
\end{aligned} \tag{2.15}$$

holds. In practice, we observe $\mathbf{YY}^T = \mathbf{ASS}^T\mathbf{A}^T + \sigma^2\mathbf{I}$, the MUSIC algorithm is performed as follows: 1) After $\mathbf{YY}^T$ is diagonalized, the number of eigenvalues bigger than σ^2 is calculated as an estimate of r; together with the corresponding eigenvectors, estimate the signal subspace . 2) All the points in the brain and, for each point, all the orientations, are spanned to find the ones that satisfy $r(\mathbf{A\Gamma}) = r(\mathbf{\Phi}_s)$. 3) Given $\mathbf{A}$ determined from the previous two steps, the $\mathbf{S}$ is fitted by solving least square solutions from $\mathbf{Y} = \mathbf{AS} + \mathbf{U}$.

The drawback of the MUSIC approach include the following: determining the number of useful eigenvalues (comparing with σ^2) might be a difficult task in real application, where the common SVD often overestimates the number of useful eigenvalues. The other problem that arises with the use of MUSIC is based on the assumptions that the data are produced by a set of asynchronous dipolar sources and that the data are corrupted by additive spatially white noise. Often both of these assumptions are incorrect in clinical or experimental data. Different versions of MUSIC, such as Recursive-MUSIC [66], RAP-MUSIC [67] have been proposed to improve the performance.

2.3.4 Beamformers

The beamformer apprach was originally developed in the radar and sonar signal processing community [17, 46]. Beamformers are spatial filters discriminating the signals on the basis of their spatial location. Recently, there has been a variety of beamformer approaches that have been introduced to study brain activity, particularly with application to the inverse problem in MEG [94, 98, 39]. The basic idea of beamformer design is to allow the source signal of interest to pass through in certain source location(s) and orientation(s), called pass-band(s), while suppressing noise or unwanted signal in other source location(s) or orientation(s), called stop-band(s). The beamformer output is a weighted linear combination of the measurements, reflecting the dipole activity in a specified location over time. The conventional beamformer approaches in MEG assume that the dipole orientations known, and therefore that there are only N beamformers, with one for each known dipole location [86, 98]. This assumption is not realistic; a more reasonable version of the beamformer approach is to design separate beamformers for each individual principal dipole orientation using the vectorized beamformer approaches [95, 85].

The general framework of beamformer approaches is: we decompose the $L \times 3N$ matrix $\mathbf{G}$ into N $L \times 3$ matrices $\mathbf{G}_i$, $i = 1, \dots N$. Each $\mathbf{G}_i = (g(\mathbf{r}'_1, \mathbf{r}_i), \cdots, g(\mathbf{r}'_L, \mathbf{r}_i))^T$ is a sub-matrix, representing the lead-fields for a dipole at a particular location across all sensors (see Section 2.3.3 for notation). The key idea in beamforming is that: the N $L \times 3$ unknown weight matrics $\mathbf{W}_i$, $i = 1, \dots N$ are introduced, and they can be determined by solving the following minimization problem (see e.g., [25])

$$\underset{\mathbf{W}_i}{\operatorname{argmin}} \ \operatorname{Var}(\hat{\mathbf{Q}}_i) \ \text{ subject to } \mathbf{W}_i^T \mathbf{G}_i = \mathbf{I}_3 \tag{2.16}$$

where $\hat{\mathbf{Q}}_i = \mathbf{W}_i^T \mathbf{Y}$. The variance term $\operatorname{Var}(\hat{\mathbf{Q}}_i)$

$$\operatorname{Var}(\hat{\mathbf{Q}}_i) = \operatorname{tr}\left[(\hat{\mathbf{Q}}_i - E(\hat{\mathbf{Q}}_i))(\hat{\mathbf{Q}}_i - E(\hat{\mathbf{Q}}_i))^T\right]$$

is used to measure the strength of the vectorial process $\hat{\mathbf{Q}}_i$. To simplify calculating $\operatorname{Var}(\hat{\mathbf{Q}}_i)$, we assume the dipoles are uncoorelated in time; that is, if we divide the $3N \times T$ matrix $\mathbf{Q}$ into N $3 \times T$ matrices $\mathbf{Q}_i$, $i = 1, \dots N$, then we will have

$$E(\mathbf{Q}_i - E(\mathbf{Q}_i))(\mathbf{Q}_j - E(\mathbf{Q}_j))^T = 0 \quad \text{if } i \neq j. \tag{2.17}$$

The problem of finding the weight matrix $\mathbf{W}_i$ that minimizes $\operatorname{Var}(\hat{\mathbf{Q}}_i)$ is equivalent to finding the matrix $\hat{\mathbf{G}}_i$ with the strength closest to the strength of $\mathbf{G}_i$ at $\mathbf{r}_n$. This statement can be easily noticed from the relation between $\hat{\mathbf{G}}_i$ and $\mathbf{G}_i$

$$\operatorname{Var}(\hat{\mathbf{Q}}_i) = \operatorname{Var}(\mathbf{Q}_i) + \operatorname{Var}\left(\mathbf{W}_i^T \left[\sum_{k \neq i}^{N} \mathbf{G}_k (\mathbf{Q}_k - E(\mathbf{Q}_k))(\mathbf{Q}_k - E(\mathbf{Q}_k))^T \mathbf{G}_k^T\right] \mathbf{W}_i\right).$$

The solution can be easily obtained using the Lagrange multiplier method (see [95])

$$\mathbf{W}_i = (\mathbf{Y}\mathbf{Y}^T)^{-1} \mathbf{G}_i (\mathbf{G}_i^T (\mathbf{Y}\mathbf{Y}^T)^{-1} \mathbf{G}_i)^{-1}. \tag{2.18}$$

There are several advantages of the beamformer over L_1-norm or (L_2-norm) dipole fitting: 1) It requires no prior assumptions about the number of dipoles; 2) Beamformers can easily handle both superficial and deep sources; 3) Statistical tests are usually difficult for both MNE and MCE solutions, whereas a variety of statistical analysis can be easily implemented using beamformer approaches. However, beamformers are very sensitive to noise in calculating the $\mathbf{W}_i$, and the inversion of $\mathbf{YY}^T$ needs to be regularized, i.e., $(\mathbf{YY}^T)^{-1}$ can be partially solved by $(\mathbf{YY}^T + \lambda\mathbf{I})^{-1}$, where the parameter λ is chosen on the basis of the noise level [98]. The assumption of uncorrelated dipole is often unrealistic from a neurophysiological viewpoint. $\mathbf{YY}^T$ is treated as stationary in beamformer analysis. A large sample size is needed to estimate $\mathbf{YY}^T$ and the local time series models for estimating $\mathbf{YY}^T$ with temporal adjustment are necessary.

2.3.5 Bayesian Methods

Recently, there have been some studies where the Bayesian formalism is used to find a solution to the MEG inverse problem. By introducing some prior information into the regularization processes discussed (i.e., L_1-norm, L_2-norm), the Bayeisan methods yield a maximum a posteriori (MAP) estimator of dipole parameters. So far, the attempts to insert physiological and anatomical criteria into the *prior* are still preliminary due to the complexity of optimization [27, 73, 10, 83, 12]. In general, the goal of the Bayesian framework is to maximize the posterior probability

$$\hat{\mathbf{Q}}_t = \underset{\mathbf{Q}_t}{\text{argmax}}\ p(\mathbf{Q}_t|\mathbf{Y}_t) \tag{2.19}$$

where $\mathbf{Q}_t = (\mathbf{q}_1(t), \cdots, \mathbf{q}_P(t))^T$ (not the $\mathbf{Q}_i$ defined in Section 2.3.4) and $\mathbf{Y}_t = (Y_{1,t}, \cdots, Y_{L,t})^T$ are corresponding columns (at time t, $1 \leq t \leq T$) of the matrices $\mathbf{Q}$ and $\mathbf{Y}$ (see Section 2.3.3). Given the distribution of $p(\mathbf{Q}_t)$, according to the Bayes' law, we have

$$p(\mathbf{Q}_t|\mathbf{Y}_t) \propto p(\mathbf{Y}_t|\mathbf{Q}_t)p(\mathbf{Q}_t) \tag{2.20}$$

where $p(\mathbf{Y}_t|\mathbf{Q}_t)$ is the likelihood function of $\mathbf{Q}_t$, which in this case is the forward model calculation. As the regularization of the $\mathbf{Q}_t$ (e.g., L_1-norm or L_2-norm) discussed in Sections 2.3.2 and 2.3.1, the $p(\mathbf{Q}_t|\mathbf{Y}_t)$ can be written owing to an energy function in terms of $\mathbf{Q}_t$ that can be associated to the probability distributions above. Specifically, let U and λ denote the energy function, we have

$$p(\mathbf{Q}_t|\mathbf{Y}_t) = \frac{1}{Z}\exp(-U(\mathbf{Q}_t)) \tag{2.21}$$

where Z is a normalization constant called partition function; then the MAP estimator of dipole becomes

$$\hat{\mathbf{Q}}_t = \underset{\mathbf{Q}_t}{\operatorname{argmin}} \; U(\mathbf{Q}_t) \tag{2.22}$$

where the $U(\mathbf{Q}_t) = U_1(\mathbf{Q}_t) + \lambda U_2(\mathbf{Q}_t)$, and $U_1(\cdot)$ and $U_2(\cdot)$ are energy functions associated with $p(\mathbf{Y}_t|\mathbf{Q}_t)$ and $p(\mathbf{Q}_t)$, respectively; λ is the tuning parameter. The MAP scheme can be related to the MNE or MCE; the $U_1(\mathbf{Q}_t)$ is simply the Frobenius norm such that

$$U_1(\mathbf{Q}_t) = \|\mathbf{Y}_t - \mathbf{G}_t\mathbf{Q}_t\|^2$$

and as a prior term, $U_2(\mathbf{Q}_t)$ is $\|\mathbf{Q_t}\|^2$ in MNE; $U_2(\mathbf{Q}_t)$ is $\|\mathbf{Q_t}\|^1$ in MCE. Forthermore, the $U_2(\mathbf{Q}_t)$ can be written as a combination of spatial constraints $U_{2,s}(\cdot)$ and temporal constraints $U_{2,t}(\cdot)$ such that

$$U_2(\mathbf{Q}_t) = U_{2,s}(\mathbf{Q}_t) + U_{2,t}(\mathbf{Q}_t).$$

In the literature, several choices of the spatial and temporal constraints are used [91, 35, 16, 69]. Although it is convenient to use a Bayesian framework to build an estimator of dipole, one of the main drawbacks of regularization techniques is that they need some well chosen tuning parameters in order to be effective. The statistical method suffers from being very time consuming and practically its convergence is not guaranteed.

2.3.6 Other Methods

Independent component analysis (ICA) has been used to identify and remove the artifacts such as blinking, eye muscle movement, facial muscle artifacts, cardiac artifacts, etc. from the MEG data [97, 51, 48]. ICA has also been studied to separate different brain sources [61, 11]. There has been some work on using other modalities of imaging methods (i.e., MRI, fMRI) in combination with the MEG data. The MRI image can give information on the position and orientation of the cortical dipoles, while fMRI provides topographical information on active dipoles. In general, the information from other imaging methods is used as prior information about the source [26, 72, 83]. The problem with this approach is that although fMRI has high spatial resolution, it has been pointed out that the hemodynamic signals of fMRI may not precisely correspond to neural activity due to various factors such as the effects of noise and artifacts. Thus, this is still an open question and a variety of hierarchical models [80, 81, 2] have been introduced attempting to investigate the inverse problem in MEG.

3.0 A TIME-VARYING SOURCE MODEL FOR THE MEG INVERSE PROBLEM WITH PARALLEL COMPUTING

3.1 A PROBABILISTIC TIME-VARYING SOURCE MODEL

3.1.1 Motivation

The methods mentioned briefly in Chapter 2 (MNE, MCE, MUSIC, etc.) have been widely used and produce meaningful solutions of dipole estimates; however, they have overly restricted model assumptions and lack estimates of variability and sensitivity of source estimates. By assuming a static localized dipole, these methods are limited in their ability to incorporate problem-specific anatomical or physiological information. It is quite reasonable to consider that the source is time-varying rather than fixed, in which case the noise reduction obtained by averaging over consecutive observations in time is problematic. By utilizing a time-varying source model, we will be able to investigate the distribution of the source at each time point and provide estimates of its variability. Following this idea, the time evolution of the source is modelled by a state space model and our goal is to find the posterior distribution of the source parameters. Our reformulaton of the inverse problem is to present a predictive model for the location and moments of each dipole. Such an approach automatically uses Bayes' rule. It turns out that the posterior source distribution from our predictive model can be interpreted as a statistical solution to the MEG inverse problem.

3.1.2 The State-space Model Formulation

Assume that the magnetic field data from the k^{th} sensor, $k = (1, \ldots, L)$ is measured respectively at time $t, t = (1, \ldots, T)$ as $Y_{k,t}$. We model $Y_{k,t}$ as

$$Y_{k,t} = \mathbf{B}_k(\mathbf{J}_t^P) + U_{k,t},\ \ 1 \leq t \leq T, 1 \leq k \leq L, \tag{3.1}$$

where $U_{k,t} \sim N(0, \sigma_1^2)$ denotes the observation noise that is assumed, for simplicity, to be Gaussian, additive, and homogeneous for all the sensors. Therefore, we can write

$$\mathbf{Y}_t = \mathbf{B}(\mathbf{J}_t^P) + \mathbf{U}_t,\ \ 1 \leq t \leq T, \tag{3.2}$$

where $\mathbf{Y}_t = (Y_{1,t}, \cdots, Y_{L,t})^T$, $\mathbf{B}(\mathbf{J}_t^P) = (\mathbf{B}_1(\mathbf{J}_t^P), \cdots, \mathbf{B}_L(\mathbf{J}_t^P))^T$ and $\mathbf{U}_t = (U_{1,t}, \cdots, U_{L,t})^T$. Here, $\mathbf{U}_t \sim MVN(\mathbf{0}, \mathbf{\Sigma}_1)$. For simplicity we assume $\mathbf{\Sigma}_1$ is a known L by L diagonal matrix with the following form $\mathbf{\Sigma}_1 = \text{diag}[\sigma_1^2, \sigma_1^2, ..., \sigma_1^2]$.

The $\mathbf{B}_k(\mathbf{J}_t^P)$, a function of the dipole with parameter vector $\mathbf{J}_t^P$, is the physical approximation of the Biot-Savart law in Section 2.2. The noiseless magnetic field, $\mathbf{B}_k$, is computed from the source $\mathbf{J}_t^P = (\mathbf{p}_t, \mathbf{q}_t)$ at time t. The vector $\mathbf{p}_t = (p_{1t}, p_{2t}, p_{3t})$ contains the location parameters of the source and the vector $\mathbf{q}_t = (q_{1t}, q_{2t}, q_{3t})$ contains the moments of the source. Thus,

$$\mathbf{B}_k(\mathbf{J}_t^P) = \frac{\mu_0}{4\pi} \frac{\mathbf{q}_t \times (\mathbf{r}_k - \mathbf{p}_t) \cdot \mathbf{e}}{|\mathbf{r}_k - \mathbf{p}_t|^3}. \tag{3.3}$$

Here, $\mathbf{r}_k$ is the location of the k^{th} sensor, $\mathbf{p}_t$ and $\mathbf{q}_t$ are parameters associated with the source defined above at time t. Because the magnetometers measure only the z direction of the magnetic field, $\mathbf{B}$, $\mathbf{e} = (0, 0, 1)$, a unit vector, is used to find $\mathbf{B}_z$, the z component of $\mathbf{B}$. Conventionally, z is perpendicular to the surface of the skull.

To specify the *prior*: first, the time evolution of the current density $\mathbf{J}_t^P$ is specified by a state space model; we note that one could choose any state space model one might wish, but for simplicity, we have chosen a six-dimensional first-order autoregression:

$$\mathbf{J}_t^P = \mathbf{m}_{com} + \rho(\mathbf{J}_{t-1}^P - \mathbf{m}_{com}) + \mathbf{V}_t,\ \ 1 \leq t \leq T, \tag{3.4}$$

where $\mathbf{V}_t \sim MVN(\mathbf{0}, \mathbf{\Sigma}_2)$ denotes the state evolution noise. We assume for simplicity that $\mathbf{\Sigma}_2 = \text{diag}[\sigma_{11}^2, \sigma_{22}^2, ..., \sigma_{66}^2]$ is a known 6 by 6 diagonal matrix and σ_{ii}^2 is the variance of the

i^{th} source parameter. The parameter vector $\mathbf{m}_{com}$ is a constant (over time) associated with the source $\mathbf{J}_t^P$ for $1 \leq t \leq T$. The initial state $\mathbf{J}_0^P$ has distribution $MVN(\mathbf{m}_{ini}, \mathbf{\Sigma}_2)$ and $\mathbf{m}_{ini}$ is also a constant (over time) parameter vector for $\mathbf{J}_0^P$. Both $\mathbf{m}_{ini}$ and $\mathbf{m}_{com}$ are specified in advance. The diagonal matrix $\rho = \text{diag}[\rho_1, \rho_2, ..., \rho_6]$ is 6 by 6 with the diagonal representing the autoregressive coefficients. Hence, $\mathbf{J}_t^P$ or $((\mathbf{p}_t, \mathbf{q}_t))$ is the random vector containing the parameters of the current at time t and $\mathbf{Y}_t = (Y_{1,t}, \dots, Y_{L,t})$ is the (very noisy) data at time t from all L sensors. Both $\{\mathbf{J}_t^P\}_{t=0}^T$ and $\{Y_{k,t}\}_{t=1}^T$ are assumed to have the following Markov properties:

(i) The $\mathbf{J}^P$ is a first order Markov process. The distribution of each state $\mathbf{J}_t^P$ only depends on its own previous state $\mathbf{J}_{t-1}^P$,

$$p(\mathbf{J}_t^P | \mathbf{J}_0^P, \mathbf{J}_1^P, \dots, \mathbf{J}_{t-1}^P) = p(\mathbf{J}_t^P | \mathbf{J}_{t-1}^P)$$

(we are using p as a generic symbol for a probability distribution; the two p's in this equation are not the same function).

(ii) The process $Y_{k,t}$ (for any $1 \leq k \leq L$) is also a Markov process with respect to the history of $\mathbf{J}^P$. The density of $Y_{k,t}$ conditioned on $\{\mathbf{J}_t^P\}_0^t$ satisfies,

$$f(Y_{k,t} | \mathbf{J}_0^P, \mathbf{J}_1^P, \dots, \mathbf{J}_t^P) = f(Y_{k,t} | \mathbf{J}_t^P)$$

(again f is a generic symbol, in this case, for a likelihood function).

(iii) When conditioned on its own history, the unknown $\mathbf{J}_t^P$ does not depend on past measurements. The distribution of $\mathbf{J}_t^P$ based on $\mathbf{Y}^k = (Y_{k,1}, \cdots, Y_{k,t-1})$ and $\mathbf{J}_{t-1}^P$ is,

$$g(\mathbf{J}_t^P | \mathbf{J}_{t-1}^P, \mathbf{Y}^k) = p(\mathbf{J}_t^P | \mathbf{J}_{t-1}^P), t > 0$$

(the right-hand side in (iii) is the same as the right-hand side in (i)). The transition kernel, $p(\mathbf{J}_t^P | \mathbf{J}_{t-1}^P)$, is defined here as a first order Markov process in the state space model above. For a more complex state space model it could be also be a higher order Markov process. The choice of more realistic models for this process (e.g., in the situation where the magnetic signal is a response to a stimulus, the source variance might change much more rapidly immediately after the stimulus than before it; the likelihood $f(Y_{k,t} | \mathbf{J}_t^P)$ for any $1 \leq k \leq L$

may also vary in time since not all the measurements can be carried out simultaneously) is not our aim for this thesis.

By taking all the previous *prior* information and the three assumptions ((i), (ii), (iii)) above into account, our problem can be stated as finding the posterior distribution, $p(\mathcal{J}_t^P|\mathcal{Y}_{obs}^t)$, given the magnetic measurements $\mathcal{Y}_{obs}^t$. By Bayes' Theorem, we have

$$\begin{aligned} p(\mathcal{J}_t^P|\mathcal{Y}_{obs}^t) &\propto f(\mathcal{Y}_{obs}^t|\mathcal{J}_t^P)p(\mathcal{J}_t^P) \\ &= \left[\prod_{s=1}^{t}\prod_{k=1}^{L} f(Y_{k,s}|\mathbf{J}_t^P)\right]\left[\prod_{s=1}^{t} p(\mathbf{J}_{s+1}^P|\mathbf{J}_s^P)\right] p(\mathbf{J}_0^P) \end{aligned} \tag{3.5}$$

where $\mathcal{Y}_{obs}^t = (\mathbf{Y}_1, \cdots, \mathbf{Y}_L) = (Y_{1,1}, \cdots, Y_{1,t}, \ldots, Y_{L,1}, \cdots, Y_{L,t})$ and $\mathcal{J}_t^P = (\mathbf{J}_0^P, \ldots, \mathbf{J}_t^P)$. Our framework is based on a one-source model (N=1). This framework can be extended to a multiple-source model which includes several $\mathbf{J}^P$ because the fields generated by distinct sources simply add. This framework is used here to find the joint source distribution given all the measurements we have. Because this distribution is high-dimensional ($1 \leq t \leq T$, T is very large), MCMC methods or conventional Gibbs sampling are very hard to implement, as we will show in Section 3.2.1. Obtainng $p(\mathcal{J}_t^P|\mathcal{Y}_{obs}^t)$ can be achieved dynamically by computing the $p(\mathbf{J}_u^P|\mathcal{Y}_{obs}^u)$ at each time point $1 \leq u \leq t$; the details can be found in Section 3.2.2.

3.2 SOLVING THE MEG INVERSE PROBLEM

3.2.1 The Difficulty of Solving the Time-varying Model

A major problem with MCMC methods (e.g., Metropolis-Hastings) for getting joint posterior samples from $p(\mathcal{J}_t^P|\mathcal{Y}_{obs}^t)$ when there are a large number of states is the difficulty of finding a joint transition kernel which can be used in an MCMC sampler. However, the goal of getting $p(\mathcal{J}_t^P|\mathcal{Y}_{obs}^t)$ can be achieved by sampling from the distribution $p(\mathbf{J}_s^P|\mathcal{Y}_{obs}^s)$ for each state s ($1 \leq s \leq t$) separately and the entire outcome can be regarded as the sample from the joint distribution. Classical Gibbs sampling can be used for this alternative goal, but because the likelihood term $f(\mathcal{Y}_{obs}^t|\mathcal{J}_t^P)$ is not linear in $\mathbf{J}_t^P$, it is not easy to sample from

$p(\mathbf{J}_t^P|\mathbf{J}_{s\neq t}^P, \mathcal{Y}_{obs}^t)$ because we do not know the form of $p(\cdot|\cdot)$. One natural way to address this is to insert some kind of Metropolis MCMC sampler for $p(\cdot|\cdot)$ into a Gibbs sampling scheme. When we insert a random-walk Metropolis algorithm into the Gibbs sampler we call it a **random-walk MCMC+Gibbs** sampler and when we insert a hybrid Metropolis algorithm into the Gibbs sampler we call it a **hybrid MCMC+Gibbs** sampler.

The key to **random-walk MCMC+Gibbs** is to propose a $\mathbf{J}_t^{P^*} \sim MVN(\mathbf{J}_t^P, \mathbf{\Sigma}_3)$ for each $t = (1, \ldots, T)$ where $\mathbf{\Sigma}_3 = \text{diag}[\tau_1^2, \tau_2^2, ..., \tau_6^2]$ is a 6 by 6 diagonal matrix and accept $\mathbf{J}_t^{P^*}$ through the Metropolis-Hasting ratio if

$$\alpha_t = \frac{\prod_{k=1}^L f(Y_{k,t}|\mathbf{B}_k(\mathbf{J}_t^{P^*}))p(\mathbf{J}_t^{P^*}|\mathbf{J}_{t-1}^P)p(\mathbf{J}_{t+1}^P|\mathbf{J}_t^{P^*})}{\prod_{k=1}^L f(Y_{k,t}|\mathbf{B}_k(\mathbf{J}_t^P))p(\mathbf{J}_t^P|\mathbf{J}_{t-1}^P)p(\mathbf{J}_{t+1}^P|\mathbf{J}_t^P)}$$

is large enough. The problem is that $MVN(\mathbf{J}_t^P, \mathbf{\Sigma}_3)$ is not a good proposal for $\mathbf{J}_t^{P^*}$ (that is, we almost always reject the proposal) and this cannot be solved by extensively tuning $\mathbf{\Sigma}_3 = \text{diag}[\tau_1^2, \tau_2^2, ..., \tau_6^2]$ in most practical cases if the dimension of the states is very high. The Taylor expansion in [87] is worth attention if we could linearize the term $f(Y_{k,t}|\mathbf{J}_t^P)$ and incorporate it into the proposal distribution. However, the extra work of a Taylor expansion might be unnecessary if we only need an efficient sampling scheme in high dimensional analysis.

The **hybrid MCMC+Gibbs** improves upon the random-walk MCMC+Gibbs when the target distribution is difficult to capture by a simple random-walk MCMC+Gibbs. In [19, 14, 87], a full conditional prior (hybrid MCMC) was proposed. Similar work can also be found in [20] where a single move blocking strategy was developed but bad convergence behavior was discovered. Gamerman [33] suggested using a reparameterization of the model to a prior independent of the system disturbance and built a proposal by a weighted least squares algorithm; however, that reparameterization resulted in quadratic computational time. Knorr-Held [55] suggested an autoregressive prior which does not approximate the full conditionals; instead of depending on the observation, the proposal is only dependent on other states. As a comparison, our hybrid MCMC+Gibbs is built on a single move proposal; that is, $\mathbf{J}_t^{P^*}$ is proposed from the distribution of $p(\mathbf{J}_t^P|\mathbf{J}_{s\neq t}^P)$ which could be further reduced to $p(\mathbf{J}_t^P|\mathbf{J}_{t-1}^P, \mathbf{J}_{t+1}^P)$ due to the Markov property. Careful computation leads to

$$\mathbf{J}_t^{P^*} \sim MVN(\rho(\mathbf{J}_{t-1}^P - \mathbf{J}_{t+1}^P) + (\mathbf{I} - \rho\rho^{'})\mathbf{m}_{com}(\mathbf{I} + \rho\rho^{'})^{-1}, \mathbf{\Sigma}_2(\mathbf{I} + \rho\rho^{'})^{-1}). \qquad (3.6)$$

The Metropolis-Hasting ratio therefore reduces to

$$\alpha_t = \frac{\prod_{k=1}^{L} f(Y_{k,t}|\mathbf{B}_k(\mathbf{J}_t^{P^*}))}{\prod_{k=1}^{L} f(Y_{k,t}|\mathbf{B}_k(\mathbf{J}_t^{P}))}.$$

The performance of a single move could be improved by extending to a block move by sampling a block of states at the same time based on other states. As an intermediate strategy, the block move method updates a block of $\mathbf{J}_t^P$s at once rather than one at a time. Naturally the $\mathbf{J}_r^{P^*}, \ldots, \mathbf{J}_s^{P^*}$ comes from the conditional proposal

$$p(\mathbf{J}_r^P, \ldots, \mathbf{J}_s^P | \mathbf{J}_{1,\ldots,T}^P / (\mathbf{J}_r^P, \ldots, \mathbf{J}_s^P))$$

where $r < s$ and $\mathbf{J}_{1,\ldots,T}^P / (\mathbf{J}_r^P, \ldots, \mathbf{J}_s^P)$ means a collection of $\mathbf{J}_1^P, \ldots, \mathbf{J}_{r-1}^P, \mathbf{J}_{s+1}^P, \ldots, \mathbf{J}_T^P$. Thus, the Metropolis-Hasting ratio becomes

$$\alpha_t = \frac{\prod_{k=1}^{L} \prod_{t=r}^{s} p(Y_{k,t}|\mathbf{J}_t^{P^*})}{\prod_{k=1}^{L} \prod_{t=r}^{s} p(Y_{k,t}|\mathbf{J}_t^{P})}.$$

Although the block move provides a considerable improvement in the situation where a single move has poor mixing behavior, Carter and Kohn [20] showed that both of these two methods will cause convergence problems.

Recently developed adaptive samplers [102, 41, 5, 4, 6, 9, 78] might help find the transition kernel within a Gibbs sampler, but these methods do not seem to work for MEG data. In addition, although parallel tempering [88] seems reasonable, finding the temperature is not straighforward and significantly increases the computational cost. Again, the MEG data set is extremely large; in particular, we collect hundreds of channels of data at each time and we collect data for hundreds of thousands of time points. It is quite difficult to implement these methods even in a simple model which has an extremely large number of states. The computational burden is even more substantial in the multiple-dipole case. We need a simple and efficient sampling scheme for our dynamic system.

3.2.2 Sequential Importance Sampling (SIS)

Sequential importance sampling (SIS) (see [59]) is advocated as a more practical tool for a dynamic system. As we mentioned briefly in Section 3.1.2, computing $p(\mathbf{J}_u^P|\mathcal{Y}_{obs}^u)$ sequentially in u for $1 \leq u \leq t$ can lead to $p(\mathcal{J}_t^P|\mathcal{Y}_{obs}^t)$. Suppose $\pi_t(\mathbf{J}_t^P) = p(\mathbf{J}_t^P|\mathcal{Y}_{obs}^t)$, calculating $p(\mathcal{J}_t^P|\mathcal{Y}_{obs}^t)$ can be achieved by performing the following two processes in sequential order

$$\pi_t(\mathbf{J}_t^P) = \frac{f(\mathbf{Y}_t|\mathbf{J}_t^P)\pi_{t-1}(\mathbf{J}_t^P)}{\pi_{t-1}(\mathbf{Y}_t)}, \tag{3.7}$$

$$\pi_t(\mathbf{J}_{t+1}^P) = \int p(\mathbf{J}_{t+1}^P|\mathbf{J}_t^P)\pi_t(\mathbf{J}_t^P)d\mathbf{J}_t^P, \tag{3.8}$$

where $f(\mathbf{Y}_t|\mathbf{J}_t^P) = \prod_{k=1}^L f(Y_{k,t}|\mathbf{B}_k)$ and $\mathbf{Y}_t$ is defined in Section 3.1.2. The denominator $\pi_{t-1}(\mathbf{Y}_t)$ is a constant $\int f(\mathbf{Y}_t|\mathbf{J}_t^P)\pi_{t-1}(\mathbf{J}_t^P)d\mathbf{J}_t^P$. The first equation computes the posterior density $\pi_t(\mathbf{J}_t^P)$ and the second equation is the well-known Chapman-Kolmogorov equation, which allows computing of the next prior density based on $p(\mathbf{J}_{t+1}^P|\mathbf{J}_t^P)$ (the initial $p(\mathbf{J}_0^P)$ is also known). For each t, most of the MCMC samples are either obtained from sampling the joint $\pi_t(\mathcal{J}_t^P)$ or some other distribution $g_t(\mathcal{J}_t^P)$ and use an acceptance criterion [45]. However, the random draws of $\pi_t(\mathcal{J}_t^P)$ are never used again when the system proceeds from π_t to π_{t+1} [19]. In high dimensions, the posterior samples for each state will have larger variation between iterations and hence both convergence and computation problems arise. In contrast, the SIS is able to reuse the current samples and help create the samples for the next iteration; that improves the computational efficiency and reduces the variations between iteratons. For non-linear problems or non-Gaussian densities, SIS requires the use of numerical approximation techniques where the key idea is to represent an approximation to the target posterior distribution by a set of samples and their associated weights.

In practice, suppose a stream $S_t = \{(\mathcal{J}_t^P)^{(j)}, j = 1, \ldots, m\}$ (m by t) is a set of random samples properly weighted by the the set of weights $\{w_t^{(j)}, j = 1, \ldots, m\}$ (m by 1) with respect to $\pi_t(\mathcal{J}_t^P)$ (this can be viewed as approximated posterior samples from $\mathcal{J}_t^P = (\mathbf{J}_1^P, \ldots, \mathbf{J}_t^P)$). Define $g_{t+1}(\mathbf{J}_{t+1}^P|(\mathcal{J}_t^P)^{(j)})$ a trial function for $\mathbf{J}_{t+1}^P$; the recursive SIS procedure produces a new stream S_{t+1} by drawing a new sample $\mathbf{J}^P_{t+1}$ and updating its associated weight. This is summarized as follows:

Algorithm 1: SIS

```
(i) Sample a new $(\mathbf{J}^P_{t+1})^{(j)}$ from the trial distribution $g_{t+1}(\mathbf{J}^P_{t+1}|(\mathcal{J}^P_t)^{(j)})$ and form $(\mathcal{J}^P_{t+1})^{(j)} = ((\mathcal{J}^P_t)^{(j)}, (\mathbf{J}^P_{t+1})^{(j)})$.
(ii) Compute the incremental weight $u^{(j)}_{t+1} = \frac{\pi_{t+1}((\mathcal{J}^P_{t+1})^{(j)})}{\pi_t((\mathcal{J}^P_t)^{(j)})g_{t+1}(\mathbf{J}^P_{t+1}|(\mathcal{J}^P_t)^{(j)})}$ and update the weight $w^{(j)}_{t+1} = u^{(j)}_{t+1}w^{(j)}_t$.
(ii*) Sample a new stream $S'_{t+1}$ from the stream $S_t$ based on the updated weights $w^{(j)}_t$.
(iii) Assign equal weights to the samples in $S'_{t+1}$.
```

Liu and Chen [60] showed that the new samples and weights $((\mathcal{J}^P_t)^{(j)}, w^{(j)}_{t+1})$ are properly weighted samples from π_{t+1}. As time t increases, a resampling scheme is inserted between adjacent times or one can just resample after the last time. This step is summarized in the (ii*) and (iii) steps. Shephard and Pitt [87] showed that resampling (Step (ii*)) is only necessary when the weights are very skewed; resampling reduces m and thus reduces the computational burden. A schedule for the resampling scheme in SIS is proposed by [36, 54, 58]. The choice of trial distribution $g_{t+1}(\mathbf{J}^P_{t+1}|(\mathcal{J}^P_t)^{(j)})$ is crucial in SIS. The choice of $g_{t+1}(\mathbf{J}^P_{t+1}|(\mathcal{J}^P_t)^{(j)}) = \pi_t(\mathbf{J}^P_{t+1}|(\mathbf{J}^P_t)^{(j)})$ is much easier to implement, although it might bring greater variation (see [13]). This procedure ends up getting $g_{t+1}(\mathbf{J}^P_{t+1}|(\mathcal{J}^P_t)^{(j)}) = p(\mathbf{J}^P_{t+1}|(\mathbf{J}^P_t)^{(j)})$ and incremental weights $f(\mathbf{Y}_{t+1}|\mathbf{J}^P_{t+1})$. There exsits in the literature several kinds of local Monte Carlo methods which could be embedded into SIS to get the weights, or even approximate weights, no matter what g_{t+1} function we choose. This strategy provides the opportunity to find relatively good weights that could be used in SIS so we thus can limit our attention to the choice of trial function when we apply SIS.

3.2.3 Regular SIS Method with Rejection

Liu and Chen's algorithm [60] inserts the standard rejection method as a local Monte Carlo scheme into the SIS procedure. The system collects local samples from the rejection method and estimates the associated weights for each state by the above estimation procedure (**Algorithm** 1). In order to improve the efficiency of SIS, the resampling scheme is used when the SIS arrives at the last time step, rather than resampling after every step. The details of the algorithm are summarized in **Algorithm** 2.

Algorithm 2: Regular SIS method with Rejection

```
(i) Initialize the first time step $\{(\mathbf{J}_0^P)^{(j)}\}_1^m$ and its weights $\{w_0^{(j)}\}_1^m$.
(ii) Sample $(\mathbf{J}_1^P)^{(j)}$ from $p(\mathbf{J}_1^P|(\mathbf{J}_0^P)^{(j)})$ over $j|_1^m$ with
    $p(\mathbf{J}_1^P|(\mathbf{J}_0^P)^{(j)}) = \frac{1}{(2\pi)^{6/2}|\mathbf{\Sigma_2}|^{1/2}} e^{-\frac{1}{2}\left((\mathbf{J}_1^P)^{(j)}-(\mathbf{m}_{com}+\rho(\mathbf{J}_0^{(j)}-\mathbf{m}_{com}))\right)^\top \mathbf{\Sigma_2^{-1}}\left((\mathbf{J}_1^P)^{(j)}-(\mathbf{m}_{com}+\rho(\mathbf{J}_0^{(j)}-\mathbf{m}_{com}))\right)}.$
(iii) Compute the constant $c_1 = \sup_j \prod_{k=1}^L f(Y_{k,1}|(\mathbf{J}_1^P)^{(j)})$.
(iv) Sample $J = j$ with $w_0^{(j)}$. Given $J = j$, draw $\mathbf{J}_1^P$ from $p((\mathbf{J}_1^P)^{(j)}|(J_0^P)^{(j)})$.
(v) Accept$(j, (\mathbf{J}_1^P)^{(j)})$ if $\frac{\prod_{k=1}^L f(Y_{k,1}|(\mathbf{J}_1^P)^{(j)})}{c_1} \geq U(0,1)$ else reject $(j, (\mathbf{J}_1^P)^{(j)})$.
(vi) Estimate the weight $w_1^{(j)}$ by $\hat{f}_j$ = frequency of $\{J = j\}$ in the $\{J_{(l)}, (\mathbf{J}_1^P)^{(l)}\}_{l=1}^{m'}$
    sample. Update the sample $(\mathcal{J}_1^P)^{(j)} = ((\mathbf{J}_0^P)^{(j)}, (\mathbf{J}_1^P)^*)$ if $\hat{f}_j \neq 0$ where $(\mathbf{J}_1^P)^*$ is any
    value of $\mathbf{J}_1^P$ if the associated $\hat{f}_j \neq 0$, or take a random draw from those with $\hat{f}_j \neq 0$ if
    associated $\hat{f}_j = 0$.
(vii) Repeat steps (ii)-(vi) with
    $p(\mathbf{J}_{t+1}^P|(\mathbf{J}_t^P)^{(j)}) = \frac{1}{(2\pi)^{6/2}|\mathbf{\Sigma_2}|^{1/2}} e^{-\frac{1}{2}\left((\mathbf{J}_{t+1}^P)^{(j)}-(\mathbf{m}_{com}+\rho(\mathbf{J}_t^{(j)}-\mathbf{m}_{com}))\right)^\top \mathbf{\Sigma_2^{-1}}\left((\mathbf{J}_{t+1}^P)^{(j)}-(\mathbf{m}_{com}+\rho(\mathbf{J}_t^{(j)}-\mathbf{m}_{com}))\right)}$
    and $c_{t+1} = \sup_j \prod_{k=1}^L f(Y_{k,t+1}|(\mathbf{J}_{t+1}^P)^{(j)})$.
(viii) Resample $m'$ out of $m$ rows from $\mathcal{J}_t^P$ without replacement based on the weights
    $w_T|_1^m$.
```

3.2.4 Improved SIS Method with Resampling

The disadvantage of the regular SIS with the rejection method is that it requires computing the constant c_{t+1} within the embedded rejection method and re-estimation of the weights for the SIS procedure from the samples $\{J_{(l)}, (\mathbf{J}_{t+1}^P)^{(l)}\}_{l=1}^{m'}$. Both of these computations could be quite inefficient in the state space model with high dimension. However, an improvement could be made when the local importance resampling takes place where the samples are not collected based on the accept/reject ratio, but instead by assigning a weight to each sample [60]. It has been proven that the samples from the local importance resampling method would automatically achieve the resampling effect. Thus, we could just keep those weights from any of the local Monte Carlo methods and directly iterate the SIS. The details of the algorithm are summarized in **Algorithm** 3.

```
Algorithm 3: Improved SIS method with Resampling
    (i) Initialize the first state $\{(\mathbf{J}_0^P)^{(j)}\}_1^m$ and weights $\{w_0^{(j)}\}_1^m$.
    (ii) Sample $J = j$ with $w_0^{(j)}$.
    (iii) Given $J = j$, draw $\mathbf{J}_1^P$ from $p((\mathbf{J}_1^P)^{(j)}|(\mathbf{J}_0^P)^{(j)})$ with
    $p(\mathbf{J}_1^P|(\mathbf{J}_0^P)^{(j)}) = \frac{1}{(2\pi)^{6/2}|\mathbf{\Sigma_2}|^{1/2}} e^{-\frac{1}{2}\left((\mathbf{J}_1^P)^{(j)}-(\mathbf{m}_{com}+\rho(\mathbf{J}_0^{(j)}-\mathbf{m}_{com}))\right)^\top \mathbf{\Sigma_2^{-1}}\left((\mathbf{J}_1^P)^{(j)}-(\mathbf{m}_{com}+\rho(\mathbf{J}_0^{(j)}-\mathbf{m}_{com}))\right)}.$
    (iv) Given $J = j$, update the weights $w_0^{(j)}$ with $p((\mathbf{J}_1^P)^{(j)}|(\mathbf{J}_0^P)^{(j)})$ or assign $w_0^{(j)}$ with 0 if $j$ is not sampled.
    (v) Repeat step (ii)-(iv) with
    $p(\mathbf{J}_{t+1}^P|(\mathbf{J}_t^P)^{(j)}) = \frac{1}{(2\pi)^{6/2}|\mathbf{\Sigma_2}|^{1/2}} e^{-\frac{1}{2}\left((\mathbf{J}_{t+1}^P)^{(j)}-(\mathbf{m}_{com}+\rho(\mathbf{J}_t^{(j)}-\mathbf{m}_{com}))\right)^\top \mathbf{\Sigma_2^{-1}}\left((\mathbf{J}_{t+1}^P)^{(j)}-(\mathbf{m}_{com}+\rho(\mathbf{J}_t^{(j)}-\mathbf{m}_{com}))\right)}.$
    (vi) Resample $m'$ out of $m$ rows from $\mathcal{J}_t^P$ without replacement based on last weights $w_T|_1^m$.
```

3.3 SIMULATION STUDY

In a typical MEG experiment, time is measured in milliseconds (the sampling rate is 1 KHz). However, for better understanding, from now on, we will use timesteps rather than milliseconds. We ran two simulated cases to verify that the methods work. First, we present some preliminary results for the single dipole case with a few parameters and low-dimension in time. Second, an extension to the single dipole case with six parameters to high-dimension in time is given. We used 40 radially oriented magnetometers and 15 timesteps in one case, 100 radially oriented magnetometers and 100 timesteps in the other case and we restricted movement of the dipole to remain inside the brain.

3.3.1 Simulated Case 1

In order to work in a known situation, we generated artificial data as follows. The head was modeled by a homogeneous sphere of radius 100 mm. The measurements of the magnetic field were simulated for 40 radially oriented magnetometers, located on the upper half of a sphere with 110 mm radius. Only one dipole was used in our simulation. We added normally distributed noise to the source. The magnetometer data were calculated from the electric source data using the Biot-Savart equation at each sensor and normally distributed noise was added. Before running our algorithms for a long time, we tested a simplified case where the simulation was run for only 15 timesteps with only one of the six parameters allowed to vary. In this very simple example, the dipole only moves in the z dimension in the brain and both the strength and moments of the dipole remain constant. The parameters of the simulated dipole are summarized in Table 1.

Table 1: Illustration of dipole simulation 1. The location parameters of the dipole are expressed in terms of Cartesian coordinates (x(cm),y(cm),z(cm)); m_1 and m_2 are the dipole moment parameters. s(mA) is the strength parameter of a dipole.

$\mathbf{m}_{int} = (x, y, z, m_1, m_2, s)$	(1,1,5,3,3,3)
$\mathbf{m}_{com} = (x, y, z, m_1, m_2, s)$	(0,0,0,0,0,0)
$\rho = \text{diag}[\rho_1, \rho_2, ..., \rho_6]$	diag[1 1 0.9 1 1 1]
$\mathbf{\Sigma}_1 = \text{diag}[\sigma_1^2, \sigma_1^2, ..., \sigma_1^2]$	$\text{diag}[0.0625, 0.0625, ..., 0.0625]$
$\mathbf{\Sigma}_2 = \text{diag}[\sigma_{11}^2, \sigma_{22}^2, ..., \sigma_{66}^2]$	$\text{diag}[0, 0, 0.0225, 0, 0, 0]$
# of timesteps	15

The Regular SIS method with Rejection (**Algorithm** 2) and the Improved SIS method with Resampling (**Algorithm** 3) were tested on this dataset. The random-walk MCMC+Gibbs and the hybrid MCMC+Gibbs were also run for comparison. In our SIS related methods, we used $(1.00, 1.09, 4.75, 3.10, 2.98, 3.15)$ for the initial state as a starting value. In the random-walk MCMC+Gibbs and the hybrid MCMC+Gibbs, we generated random values from the joint source distribution and used them as starting values for all the states. Figure 5 summarizes sample plots for 4 selected timesteps from all the methods. We observe that both the random-walk MCMC+Gibbs and hybrid MCMC+Gibbs do not provide a stable estimate for each timepoint and their samples are highly correlated. Both of our methods produce much stable samples which oscillate around the true values.

3.3.2 Simulated Case 2

In addition to Case 1, a case of multiple source parameters (three location parameters and three moment and strength parameters) was done. In this simulation, the source was modeled as a moving dipole following a multivariate autoregressive time series model. The dipole moves in the three coordinate directions x, y and z and both strength and moments of the dipole change as well. The magnetic measurements were simulated for 100 radially oriented magnetometers, located on the upper half of a sphere with 110 mm radius. To control the movement of the simulated dipole (to not move outside of the brain when the number of timepoints are large), we restricted the range of each parameter for the dipole. In order to do this, we set boundary values for each parameter (i.e., maximum and minimum). The autoregressive model for $\mathbf{J}_t^P$ in Section 3.1.2 occured only at certain timepoints specified in advanced. In other words, the dipole had two type of moves: one is a move based on the autoregressive model, and the other is a random walk move. The dipole moved according to the autoregressive model at certain specified timesteps, whereas the random walk was applied to the dipole at the other timepoints. We had similar restrictions on the other parameters of the dipoles. The total length of simulation is 100 timesteps (we will run 2000 timesteps for data in Section 3.4.1). The starting values for the initial state are set to $(5.9, 7.15, 7.97, 2.89, 5.09, 4.97)$. The parameter setup is given in Table 2. The location and moments of simulated source of 100 timesteps are given in Figure 4. The plots (histogram) for each dipole location parameter and pairwise plots for the location parameters are shown in Figure 6. These side by side histograms show the distribution of each location parameter at 6 selected timepoints. Similar plots for the other three moment and strength parameters are also shown in Figure 7. We can see the distribution (not Gaussian) of each parameter of the source is varying at each timestep as we expected.

Table 2: Illustration of dipole simulation 2. The location parameters of dipole are expressed in terms of Cartesian coordinates (x(cm),y(cm),z(cm)); m_1 and m_2 are the dipole moment parameters. s(mA) is the strength parameter of a dipole. The diagonal elements of $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$ are 0.0625 fT^2 and 0.01 cm^2, respectively.

Initial Timepoint	
$\mathbf{m}_{int} = (x, y, z, m_1, m_2, s)$	(6,7,8,3,5,5)
$\mathbf{m}_{com} = (x, y, z, m_1, m_2, s)$	(0,0,0,0,0,0)
$\rho = \text{diag}[\rho_1, \rho_2, ..., \rho_6]$	diag[0.65 0.7 0.75 0.8 0.85 0.9]
Random-walk Move	
(x, y, z, m_1, m_2, s)	based on previous value
# of timesteps	10
Autoregressive Move	
(x, y, z, m_1, m_2, s)	based on previous value
$\mathbf{m}_{com} = (x, y, z, m_1, m_2, s)$	(0,0,0,0,0,0)
$\rho = \text{diag}[\rho_1, \rho_2, ..., \rho_6]$	diag[0.65 0.7 0.75 0.8 0.85 0.9]
Random-walk Move	
(x, y, z, m_1, m_2, s)	based on previous value
# of timesteps	10
...	...
repeat until 100th timepoint	

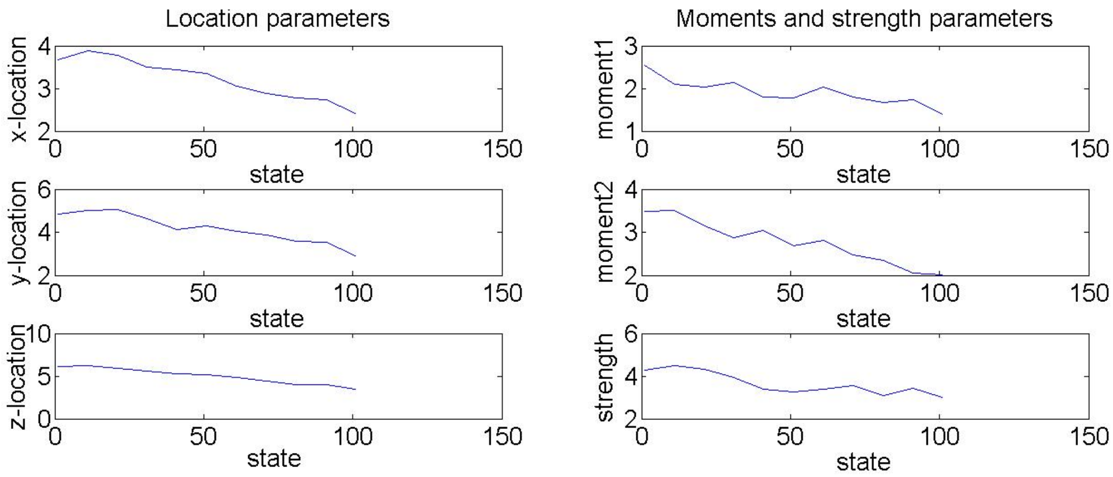


Figure 4: Illustration of the case where all the source parameters are allowed to vary. Left: simulated source of 100 timesteps for the three location parameters. Right: simulated source of 100 timesteps for the moment and strength parameters.

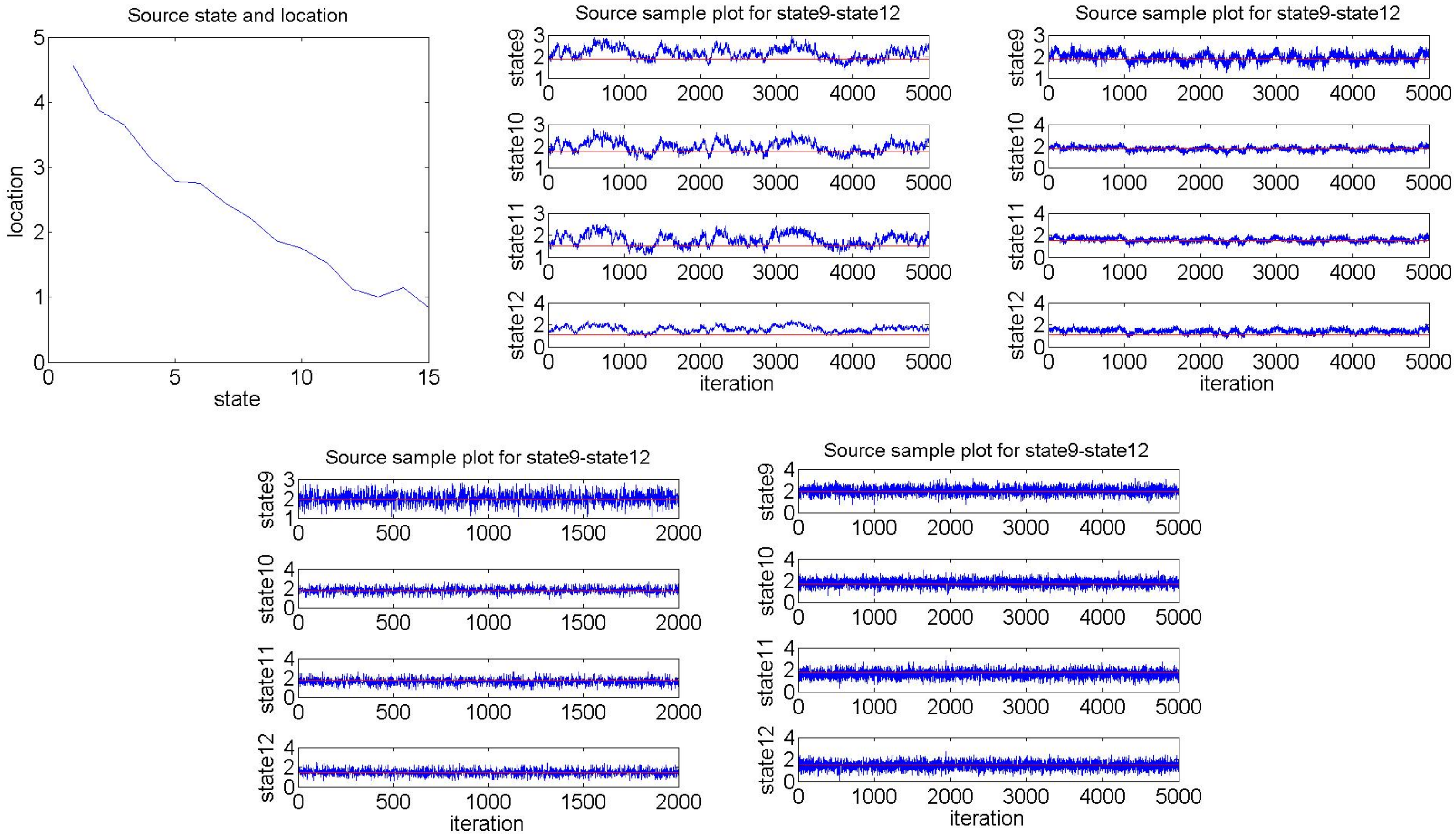


Figure 5: Comparison of methods for a simple test case where only one source parameter z is allowed to vary. Top left: simulated source for 15 timesteps for location parameter z. Top middle: sample plots of location parameter z at four selected timesteps (9th, 10th, 11th and 12th) by the random-walk MCMC+Gibbs. Top right: sample plots of location parameter z at four selected timesteps (9th, 10th, 11th and 12th) by the hybrid MCMC+Gibbs. Bottom left: sample plots of location parameter z at four selected timesteps (9th, 10th, 11th and 12th) by the Regular SIS method with Rejection (**Algorithm** 2). Bottom right: sample plots of location parameter z at four selected timesteps (9th, 10th, 11th and 12th) by the Improved SIS method with Resampling (**Algorithm** 3).

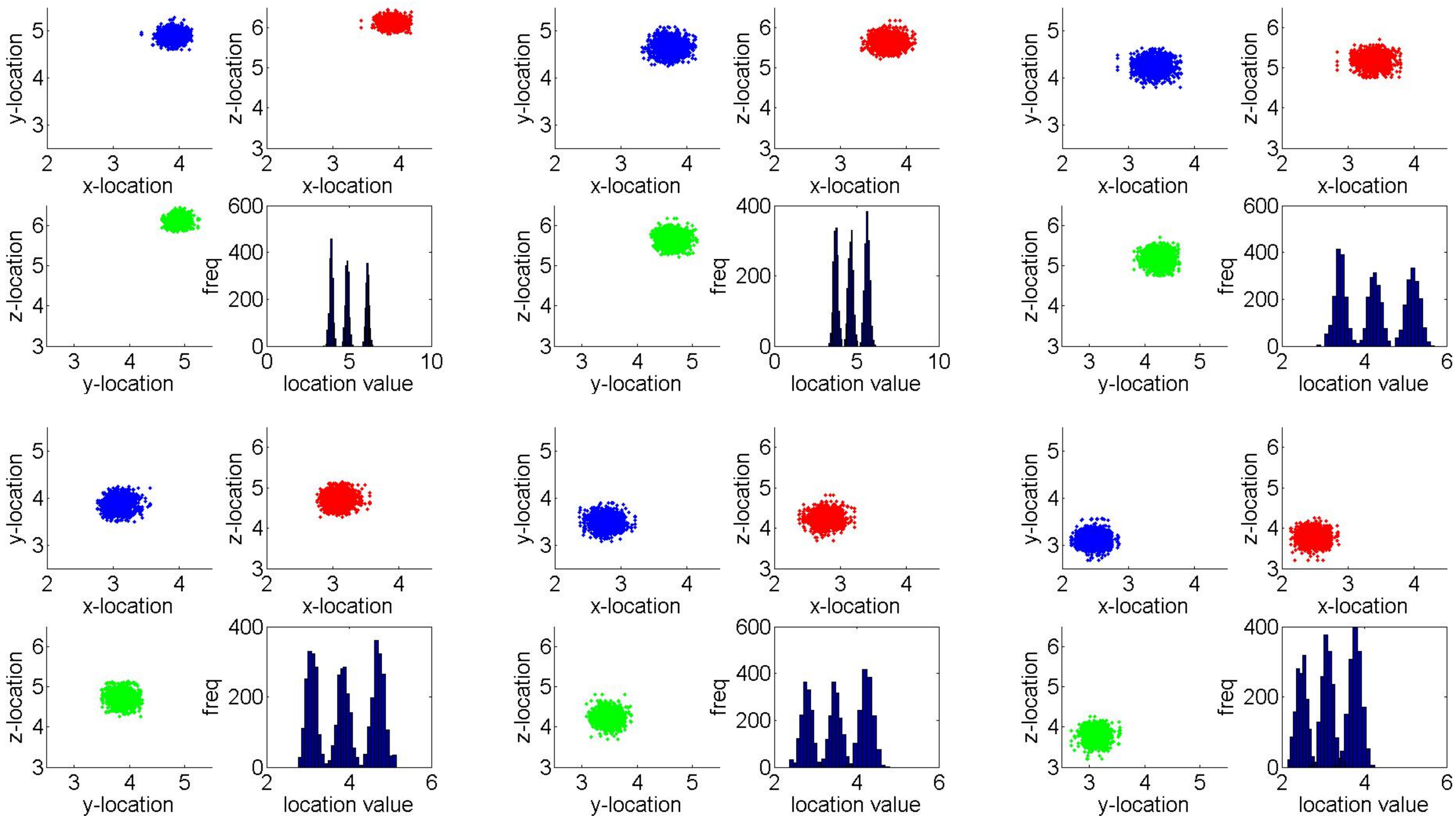


Figure 6: Distribution of source location parameters at six timesteps. Top left: pairwise plots of the source location parameters (x and y, x and z, y and z) at 1st timestep; side by side histogram plots for the source location parameters (x, y and z) at 1st timestep. The rest of the five plots give the same information for different timesteps: 20th timestep (Top middle), 40th timestep (Top right), 60th timestep (Bottom left), 80th timestep (Bottom middle) and 100th timestep (Bottom right).

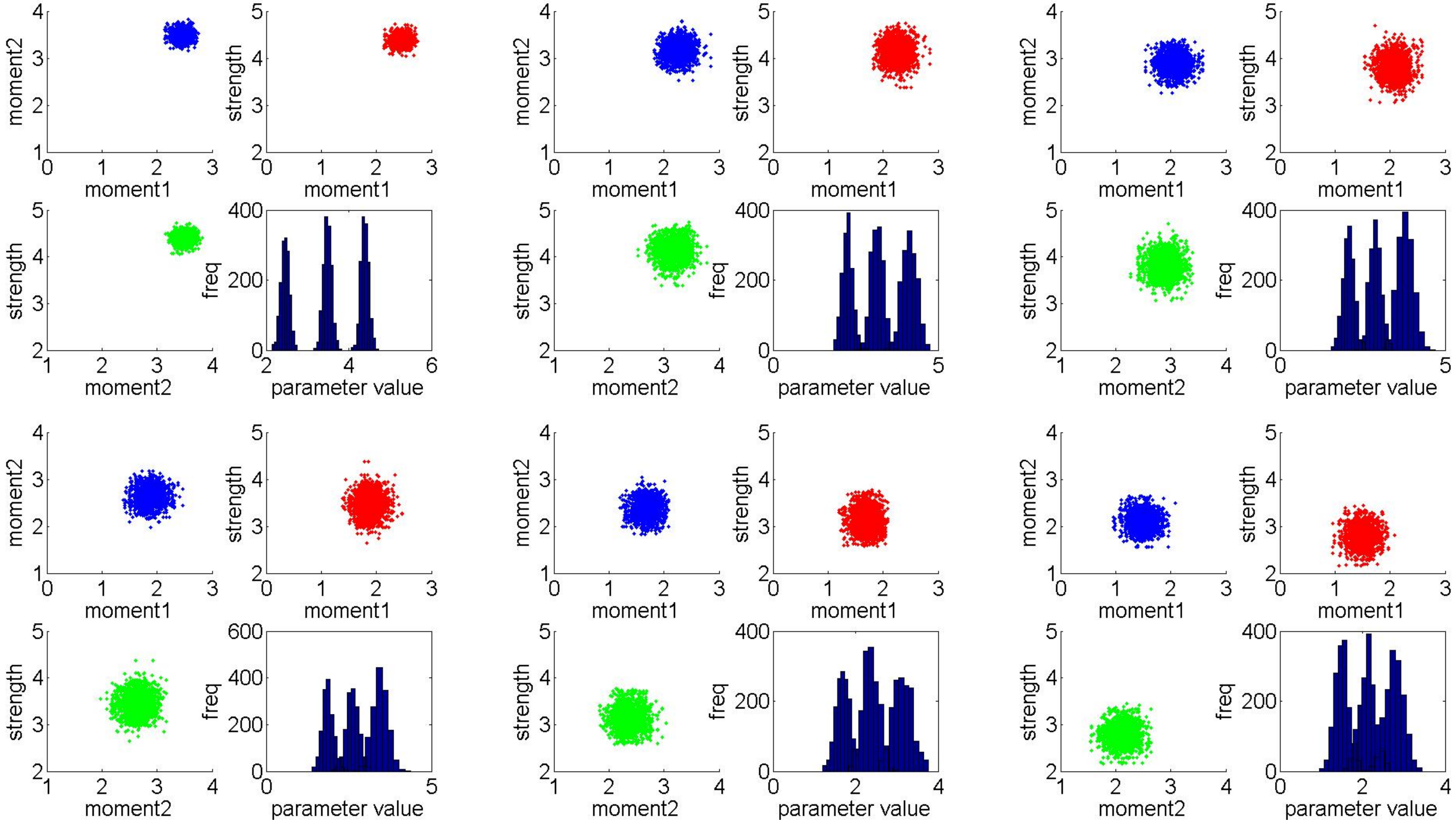


Figure 7: Distribution of source moment and strength parameters at six timesteps. Top left: pairwise plots of the source moment and strength parameters (m_1 and m_2, m_1 and s and m_2 and s) at 1st timestep; side by side histogram plots for the moment and strength parameter (m_1, m_2 and s) at 1st timestep. The rest of the five plots give the same information for different timesteps: 20th timestep (Top middle), 40th timestep (Top right), 60th timestep (Bottom left), 80th timestep (Bottom middle) and 100th timestep (Bottom right).

3.4 PARALLEL VIRTUAL MACHINE (PVM) FOR HIGH-DIMENSION IN TIME

In practice, the MEG dataset we have from an experiment is very large (e.g., hundreds of thousands of timesteps or more). A natural extension of running our algorithms (Section 3.2.3 and Section 3.2.4) is to run them for a much longer time. To be exact, if we run for 5000 timesteps with 1500 replications (sample paths) for each $\mathbf{J}_t^P$, we are supposed to get a stream of $S_{5000} = \{(\mathcal{J}_{5000}^P)^{(j)}, j = 1, \ldots, 1500\}$ (S_t is defined in Section 3.2.2). Because of the sequential character of our algorithms, sample paths for each time are computed in a sequential fashion and the weights updated at each time. Therefore, it is very inefficient to get the sample paths for a longer time. Based on our previous experience, the time spent for running 100 timesteps has increased significantly from 15 minutes (a single dipole parameter) to about 40 minutes (multiple parameters) on a single computer. It turns out that a computational challenge arises even running for only 5000 timesteps. So far, we have only used time spent (waiting time) to measure the running time of an algorithm. In the next section, a formal terminology for time spent will be evaluated and compared.

Improving the speed is necessary and meaningful for the practical use of our algorithms. Since we always need the sample path for the previous time ($\mathbf{J}_{t-1}^P$) when we work on the current time ($\mathbf{J}_t^P$) and they are not independent, therefore, the speed cannot be possibly improved in the direction of time (e.g., sequentially). However, the sample paths are independent within each timestep (e.g., at time t), so they can be computed in a separate fashion. In other words, it is always possible for us to compute several sample paths (several chunks) for the same timestep (at time t) simultaneously. This simultaneous computation for sample paths at each time t until the final timestep (5000) could be achieved by computing in parallel where each parallel scheme contains a sequential calculation for all the time t ($1 \leq t \leq 5000$) with fewer samples, so that our sequential problem can be solved in parallel.

We use a parallel computing paradigm called Parallel Virtual Machine (PVM) [34] here to speed up the computation. PVM software allows parallel computing using a message-passing paradigm for a parallel network of computers. It is designed to allow a network of heterogeneous machines to be used as a single distributed parallel processor. Thus large computational problems can be solved more cost effectively by using the aggregate power and memory of many computers. The PVM structure we use is a Master-Worker model (Figure 8) where there are several worker programs performing tasks in parallel and a master program collecting the outcomes from each worker. Each task is to separately compute a partial sample path for all the timesteps. The resampling scheme is included in the worker program and there is no parallelism in time. To be exact, if there are three worker programs in the Master-Worker model to generate a steam $S_T = \{(\mathcal{J}_T^P)^{(j)}, j = 1, \ldots, m\}$, the way of running PVM is:

```
Algorithm 4: PVM schedule
    (i) Initialize each worker program and let each worker run Algorithm 2 (or Algorithm 3) for $1 \leq t \leq T$ for a substream $S'_T = \{(\mathcal{J}_T^P)^{(j)}, j = 1, \ldots, \frac{m}{3}\}$.
    (ii) Stack each $S'_T$ and get a complete $S_T$.
```

The size of S'_T can be adjusted according to the size of S_T and the number of worker programs that are in use. The PVM speed is mainly influenced by hardware and software components of network and I/O systems. The speed of PVM also depends on the number of worker programs, e.g., adding too many parallel workers does not enhance the speed when most of the time is spent on communication among the workers. In practice, deciding on a good number of workers requires experience and it varies for the different performance of machines. An application of running PVM for our simulated data is in Section 3.4.1. The whole program at this time handles 102 channels of MEG data and works with one brain source with multiple parameters. Since the magnetic fields generated by independent dipoles add, there is no complexity (other than increased computation) brought by multiple dipoles.

3.4.1 Numerical Results for Running PVM

The PVM program was initially run on a single Linux workstation (Intel Pentium 4 CPU 3.80GHz, Memory 2 GB) for different PVM configurations. The data size was 2000 MEG timesteps with 1500 sample paths for each timestep. We split the computation into a number of tasks: 1 (without PVM), 3, 5, 10 and 15 workers respectively and run for 100 timesteps, 500 timesteps, 1000 timesteps, 1500 timesteps and 2000 timesteps. The user CPU time (total number of CPU-seconds for master and worker programs) is used to measure the time spent by our program for each PVM run. The real time elapsed (Minutes) is also attached

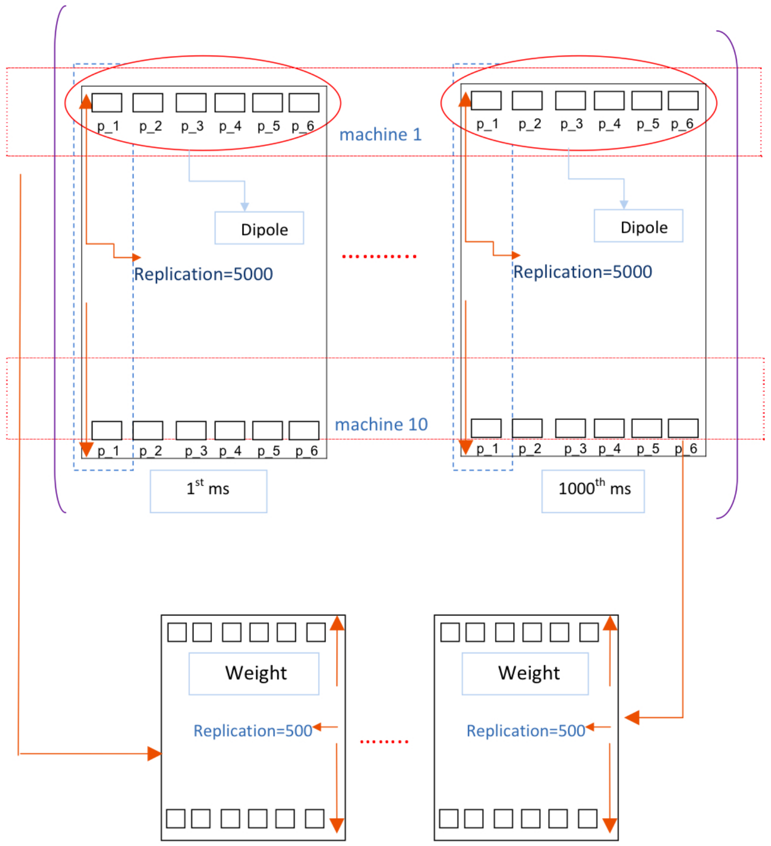


Figure 8: PVM Structure (Master-Worker model). Top plots: each large rectangle represents a collection of sample paths (size=5000) for six source parameters (each small rectangle underlined by p-i,i=1...6) for a single timestep. The collections for all the timesteps can be computed by different workers in parallel with each worker computing some part of the sample paths for all the 1000 timesteps (small rectangle). Each row represents one sample path for the six parameters. Bottom plots: the weight matrix (size=5000) for each sample path from each worker.

Table 3: Machine configuration for PVM

Machine Name	Model Name	CPU (MHz)	Stepping	Cache Size (KB)
machine1	Intel(R) Pentium(R) 4	3790.644	3	2048
machine2	Intel(R) Xeon(TM)	3000.000	1	1024
machine3	Intel(R) Xeon(TM)	3000.000	1	1024
machine4	Intel(R) Xeon(TM)	2800.000	3	2048
machine5	Intel(R) Xeon(TM)	3000.000	3	2048

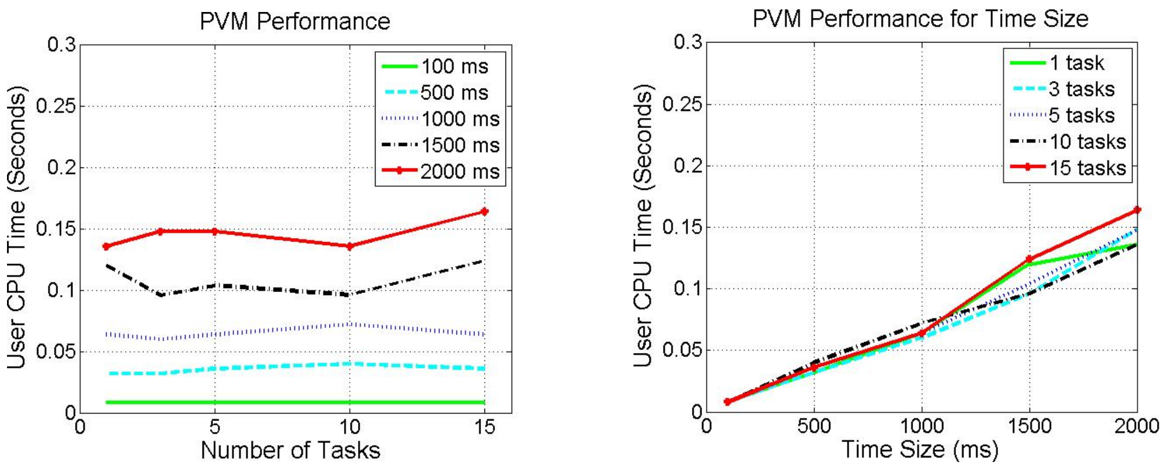
in parenthesis behind the user CPU time. The result is shown in Table 4.

We can see that the user CPU time increases roughly linearly in the number of timesteps from 0.008 second to 0.146 second on average. The linear relationship of user CPU time on experiment time is almost the same for each of these PVM configurations as we expected. This can be clearly observed from Figure 9: in Figure 9 (a), these lines (user CPU time/Task) are nearly equally distant and stay roughly constant for different tasks within the samesteps time run; in Figure 9 (b), the slope of each line (user CPU time/Timesteps) is almost the same. Notice that there is a significant difference in real time elapsed for different PVM configurations. This should not be considered a contradiction with user CPU time because real time elapsed is mostly affected by other programs and it includes time spent in memory, I/O and other resources.

The performance can still be improved when extra machines are included. Table 5 lists the PVM performance of 1 machines, 2 machine, 3 machines and 4 machines with experiment time 1500 timesteps. First, since user CPU time is the sum of the CPU time for master and worker programs, it is expected that the user CPU time for each of these PVM runs is roughly 0.120 second. Second, the real time elapsed of each PVM runs is cut into 50%-70% if one machine is added. The real time spent goes down to 40%-50% when three computers are employed. The real time elapsed decreases to 10%-30% when four computers are added. These performances are based on our public computer cluster with heterogeneous CPU speed

Table 4: Illustration of PVM application on a single workstation. Five different PVM configurations were run. The number of workers in PVM is denoted "# of Tasks." The number of sample paths within each worker is denoted "# per Task." Each PVM run eventually generates 1500 sample paths. Each PVM configuration was run for 100 timesteps, 500 timesteps, 1000 timesteps, 1500 timesteps and 2000 timesteps. This table shows the user CPU time (Seconds) for each PVM run and real time elapsed (Minutes) in parenthesis.

# of	# per	CPU Time				
Tasks	Task	Timesteps 1	Timesteps 2	Timesteps 3	Timesteps 4	Timesteps 5
		(100)	(500)	(1000)	(1500)	(2000)
1	1500	0.008(1.00)	0.032(5.12)	0.064(10.35)	0.120(16.47)	0.136(22.08)
3	500	0.008(0.24)	0.032(2.05)	0.060(4.12)	0.096(6.23)	0.148(8.40)
5	300	0.008(0.17)	0.036(1.27)	0.064(3.17)	0.104(4.25)	0.148(6.47)
10	150	0.008(0.11)	0.040(0.59)	0.072(1.59)	0.096(3.00)	0.136(4.51)
15	100	0.008(0.10)	0.036(0.50)	0.064(1.43)	0.124(2.33)	0.164(3.24)



(a) CPU time with Tasks

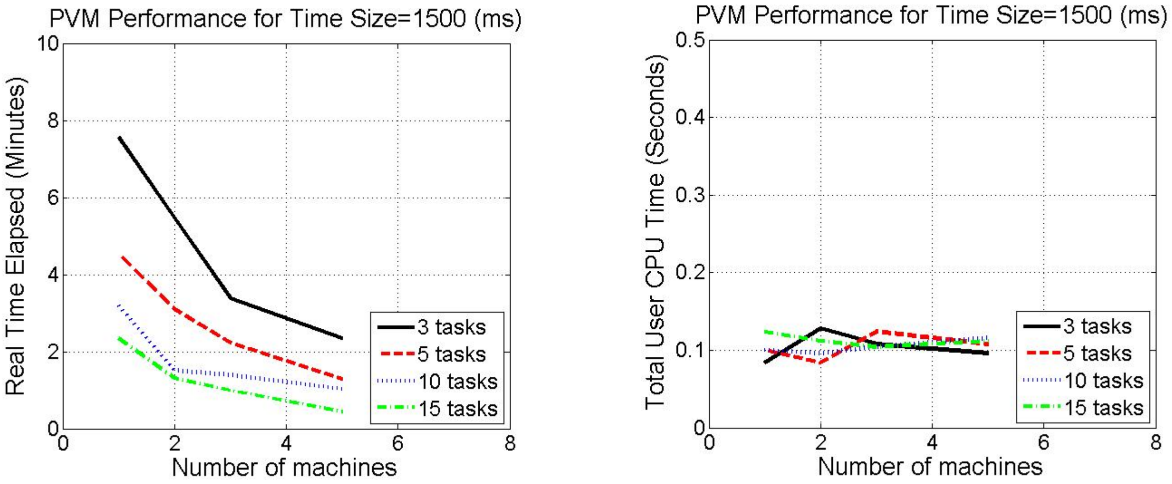
(b) CPU time with Timesteps

Figure 9: PVM Performance: user CPU time (Seconds) for number of tasks and different time run. (a) Each line (with a specific timesteps) is a plot of user CPU time for different number of tasks. (b) Each line (with a specific number of tasks) is a plot of user CPU time for different timesteps.

Table 5: Illustration of PVM application on multiple workstations. This table shows the user CPU time (Seconds) for each PVM run and real time elapsed (Minutes) in parenthesis for using one, two, three and four machines. The length of each PVM run was 1500 timesteps.

# of	# per	PVM Performance			
Tasks	Task	1	2	3	4
		(1500)	(1500)	(1500)	(1500)
3	500	0.084(7.56)	0.128(5.46)	0.108(3.39)	0.096(2.34)
5	300	0.100(4.55)	0.084(3.10)	0.124(2.23)	0.108(1.29)
10	150	0.100(3.19)	0.096(1.51)	0.104(1.39)	0.116(1.03)
15	100	0.124(2.34)	0.112(1.31)	0.104(1.00)	0.112(0.44)

and cache size (Table 3); the theoretical reduction in execution time of our program by PVM is not necessarily expected. For example, in our 3-task run, it is expected that we include no more than 3 machines and the time should remain the same if an extra one is added. However, we still have decreased real time when we run this on four machines. The reason is that it is always the first three machines (sequentially) that are used and one standby. Finally, we still get reasonable time reduction from our computer cluster for each of those runs. It is suggested that, in order to get a good time execution by PVM, we need to adjust number of CPU, number of tasks and use relatively similar machines. To summarize, Figure 10 is a graphic illustration of both real time elapsed and user CPU time for our PVM run.



(a) Real time for PVM run

(b) Total user CPU time for PVM run

Figure 10: PVM Performance: real time elapsed (Minutes) and user CPU time (Seconds) graph for number of machines for 1500 timesteps PVM run. (a) Each line (with a specific number of tasks) is a plot of real time elapsed for different number of machines. (b) Each line (with a specific number of tasks) is a plot of total user CPU time of master and worker programs for different number of machines.

3.5 A REAL DATA APPLICATION

Data was collected by a 306-channel (102 magnetometers and 204 planar gradiometers) system (Elekta-Neuromag) at the Center for Advanced Brain Magnetic Source Imaging (CABMSI) at UPMC Presbyterian hospital in Pittsburgh in an experiment related to Brain-controlled interfaces (BCI). A BCI expresses motor commands via neural signals directly from the brain. The experiment involves two parts (see Figure 11): in the first part the subjects were asked to imagine performing the "center-out" task using the wrist (imagined movement task) and in the second part the subjects controlled a 2-D cursor using the wrist to perform the center-out task following a visual target (overt movement task). The magnetic field at each sensor was acquired at sampling rate of 1000 Hz.

Our data consists of one trial recording 37000 milliseconds long at 102 MEG sensors (magnetometers). We used this data for testing our model along with our PVM scheme rather than decoding the intended movement direction of subjects. Instead of analyzing the whole trial of data, we only analyzed about 400 milliseconds (dashed box in Figure 12) after movement onset (12000 - 12400 milliseconds in the original data) from all the channels. To simplify our calculation for the real data, we were only estimating the location of the source (x, y, z). The moment and strength parameters (m_1, m_2, s) were not of our interest (not varying too much by assumption). The choice of prior for real data is an open question; we used almost the same prior as we did in Section 3.3.2 for simplification. We set the mean $\mathbf{m}_{ini}$ of the initial state $\mathbf{J}_0^P$ as $(-4, -4, 11)$ motivated by the minimum norm estimate [43], which is $(-2, -2, 10)$. The starting values for the initial state are set $(-4.06, -3.77, 13.13, 1.11, 0.98, 1, 12)$. The empirical density plots of the dipole location parameter (x, y, z) at two selected timesteps are shown in Figure 13. Different initial values might have different performance due to the complexity of this problem and the real data; thus, a more realistic prior needs to be investigated in our future work. We ran PVM for 1500 milliseconds (12000 milliseconds - 13500 milliseconds in the original data) with the same PVM configuration as our simulation; the time spent is very close to that from our result in Section 3.4.1.

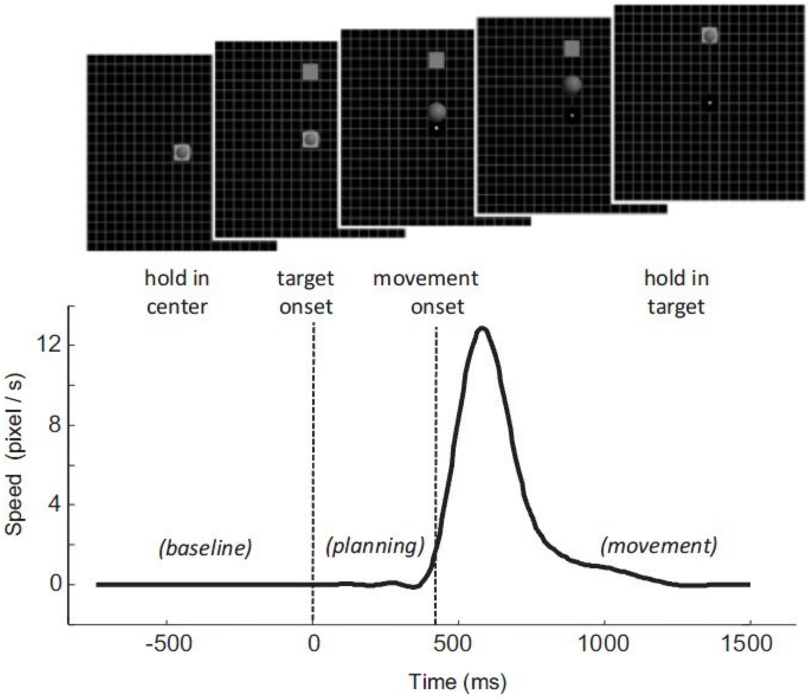


Figure 11: The subject controls the 2-D cursor position using wrist movements. The cursor needs to go to the center and stay there for a hold period until the peripheral target appears. Then the cursor moves from the center out to the target and stays there for another hold period to complete the trial successfully. The target changes color when hit by the cursor, and disappears when the holding period has finished. The bottom trace shows the speed profile of the cursor from a representative trial, and the dotted lines delimit the pre-movement/planning period. Picture and explanations were obtained from J Neurophysiol (August 25, 2010). doi:10.1152/jn.00239.2010.

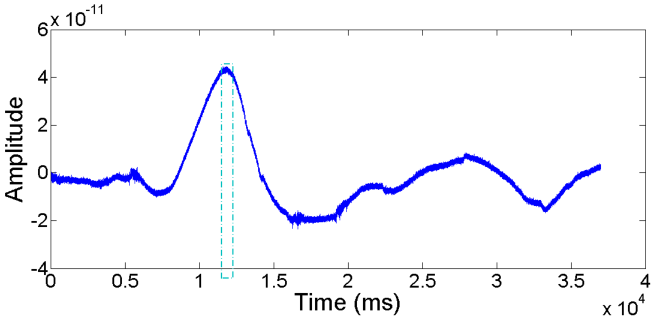


Figure 12: The MEG signal of a typical trial at a sensor. The Horizontal line is the time (ms); the vetical line shows the magnitude of the signal (fT).

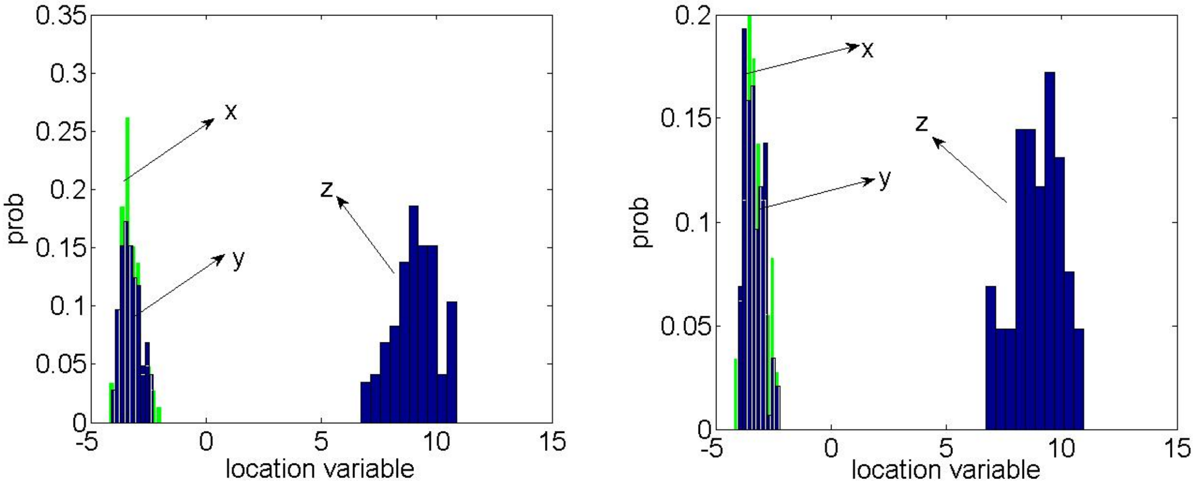


Figure 13: Empirical density for source location parameter (x, y, z). Left: density plot for 1st millisecond; Right: density plot for 100th millisecond.

3.6 DISCUSSION

We studied the MEG inverse problem. For this important scientific problem, we formulated a predictive model aiming to find the source distribution. We modelled a time-varying source by a state-space model and tackled the inverse problem by finding the posterior distribution for the source at each time point rather than fitting a single estimate. Due to the complexity of the problem (i.e., non-linearity of likelihood; high-dimension), we discussed why the conventional MCMC methods (e.g., random-walk MCMC+Gibbs or hybrid MCMC+Gibbs) would not be able to work efficiently in this situation. Two algorithms based on sequential methodology were proposed in order to find the source distributions. We also addressed developing an efficient computing scheme to speed up the computation of the methods. A practical aspect of our study is that we could provide useful information for a doctor who might be interested to know, before conducting brain surgery, where the source area might be. To make this huge computation possible, a set of C programs under LINUX has been developed and the PVM extension has been used.

Our results so far were mainly based on a one-source model where we assume there is only one dipole in the MEG data. We are still developing a multiple-source model for the MEG inverse problem. The extension from one source to having multiple sources is natural and only the computational complexity increases. Our algorithms will still work in this multiple-source model case. However, to determine the number of sources in the MEG data is still an open question. To experiment with this issue, there are three general ways of finding the number of sources for the advanced model. The first one, which is relatively easy, is to use a pre-defined number of sources for the data. The second one is to estimate the number of sources from the data in advance [100, 103]. The third one is to model the number of the sources in a Bayesian way using a prior distribution.

We simulated a dataset to run our algorithms and PVM. We compared our algorithms with other MCMC methods. In our simulation sudy, in order to focus on the source parameters we fixed several parameters (source noise parameters, measurement noise parameters, etc.) in the model. In fact, those parameters could be estimated along with the source distribution. The natural way of implementing this is to iterate those parameters and the source distribution until all of them become stable. Furthermore, the skewness of weights that arise in the sequential importance sampling could be a tradeoff between the efficiency of the program and the quality of the source distribution. Residual sampling can be used to replace regular weight sampling. For a real data analysis, a collection of high frequency (1 KHz) MEG data was analyzed. We used the same prior in this analysis and calculated the distribution of the source location.

Our results show that PVM did improve the speed by computing in parallel. Since our PVM program involves randomness and a resampling scheme, several issues from our PVM implementation still need to be resolved. First, if our algorithm were implemented in a single program without parallelism, all samples generated before resampling from this program should be simply related to the random number generator. However, when there were several workers with each of them doing the same thing as a single program but in parallel, the unique randomness within each worker will eventually come up with different but similar samples before resampling. To be exact, in order to have the two programs generate the same results, in the PVM structure we need to explicitly and precisely choose different workers correspondingly according to a pre-defined random sequence. This random sequence can be obtained from the single program without parallelism. Unfortunately, this needs a lot of work in programming. Second, in a single program without parallelism, we would only have one resampling procedure. The samples would be generated from the resampling procedure. However, there was one resampling procedure within each of our worker programs in PVM. The samples were generated from each of these workers and should eventually be pooled together. In principle, the weights from each worker should be pooled first and then we would perform the resampling procedure. The reason is each worker might generate different weights so that the normalizing constants might be different. If the resampling happens only one time (at the end of all timesteps), a reasonable way to solve this problem is that we can do the resampling scheme in the master program after normalizing all the weights when pooled. If there are several resampling schemes before the end of timestep, we can still return to the master program at each time. Again, it needs extensive programming. In our current program, sums of weights within each worker were almost the same (normalizing constants are almost the same), so we retained the resampling procedure in each worker program. Third, there is always a tradeoff between resampling in parallel or not. Since we are dealing with a huge amount of data, our goal is to discover some distributed samples and it is not our interest to get the exact same samples as a single sequential program does.

4.0 STATISTICAL APPROACHES TO ESTIMATING THE NUMBER OF SIGNAL SOURCES IN MEG

4.1 DETERMINING THE NUMBER OF SIGNAL SOURCES BY INTRINSIC DIMENSIONALITY (ID)

4.1.1 Matrix View of Finding the Number of Sources for MEG Data

The conventional estimation of the number of signal sources for any hyperdimensional data (e.g., $\mathcal{Y}^t_{obs}$ defined in Eq. (3.5)) is equivalent to estimating the minimum number of parameters required to account for the observed properties of the data. In practice, this number is difficult to find because it is much smaller than the dimensionality of the data sample vector (e.g., L). Ideally, this problem can be formulated as a maximization problem based on intrinsic dimensionality (ID). The maximization procedure (see [32]) is to derive a matrix $\mathbf{X}$ (a L by L matrix) such that

$$\underset{\mathbf{X}}{\text{argmax}} \quad \frac{\mathbf{X}^T\mathbf{R}\mathbf{X}}{\mathbf{X}^T\mathbf{R}_n\mathbf{X}} = \underset{\mathbf{X}}{\text{argmax}} \quad \frac{\mathbf{X}^T\mathbf{R}_s\mathbf{X}}{\mathbf{X}^T\mathbf{R}_n\mathbf{X}} + 1, \tag{4.1}$$

where $\mathbf{R}$ is the covariance matrix of the observed data matrix $\mathcal{Y}^t_{obs}$ defined in Section 3.1.2; specifically, $\mathbf{R}_s$ and $\mathbf{R}_n$ are the covariance matrices for the pure signal $\mathbf{B}(\mathbf{J}^P_t) = (\mathbf{B}_1(\mathbf{J}^P_t), \cdots, \mathbf{B}_L(\mathbf{J}^P_t))^T$ $(1 \le t \le T)$ and the noise $\mathbf{U}_t = (U_{1,t}, \cdots, U_{L,t})^T$ $(1 \le t \le T)$, respectively. If we assume that the signal is uncorrelated, then the matrix $\mathbf{R}$ can simply break down to $\mathbf{R} = \mathbf{R}_s + \mathbf{R}_n$ (see i.e., [37, 92]). Suppose we know the noise covariance matrix $\mathbf{R}_n$, then a whitening process can be applied to transform both $\mathbf{R}$ and $\mathbf{R}_n$

$$\mathbf{W}'_n\mathbf{R}\mathbf{W}_n = \mathbf{W}^T_n\mathbf{R}_s\mathbf{W}_n + \mathbf{W}^T_n\mathbf{R}_n\mathbf{W}_n = \mathbf{R}_{s,adj} + \mathbf{I} = \mathbf{R}_{adj} \tag{4.2}$$

where $\mathbf{W}_n = \mathbf{\Phi}_n \mathbf{\Lambda}_n^{-1/2}$ denotes the transformation matrix, in which $\mathbf{\Phi}_n$ and $\mathbf{\Lambda}_n$ are the associated eigenvectors and eigenvalues of $\mathbf{R}_n$, respectively. Therefore we have

$$\mathbf{\Phi}_{adj}^T \mathbf{R}_{adj} \mathbf{\Phi}_{adj} = \mathbf{\Lambda}_{adj} \tag{4.3}$$

Finally we have $\mathbf{X} = \mathbf{\Phi}_n \mathbf{\Lambda}_n^{-1/2} \mathbf{\Phi}_{adj}$ and

$$\mathbf{X}^T \mathbf{R} \mathbf{X} = \text{diag}[\lambda_1, \lambda_2, ..., \lambda_p, \lambda_{p+1}, ..., \lambda_L] \tag{4.4}$$

where $\mathbf{\Phi}_{adj}$ and $\mathbf{\Lambda}_{adj}$ are the associated eigenvectors and eigenvalues of $\mathbf{R}_{adj}$, respectively. The two sequences $\{\lambda_i\}_{i=1}^p$ and $\{\lambda_i\}_{i=p+1}^L = 1$ are the associated eigenvalues for $\mathbf{R}_s$ and $\mathbf{R}_n$. The constant 1s are the eigenvalues of $\mathbf{R}_n$ from the whitening process. Therefore, the intrinsic dimensionality of the data can be determined by counting the number of eigenvalues of $\mathbf{R}_{adj}$ that are larger than unity.

4.1.2 Previous Work on Estimating the Number of Signal Sources

Model-choosing methods, such as PCA, AIC, etc., have been used for a while for both multispectral data (with only a small number of channels) and hyperspectral data (a large number of channels). All of these methods try to find the minimum number of parameters that are required to account for the data, and use minimum number that as an estimate of the number of sources for the data. While these methods sometimes work well on multispectral data, they are very limited in hyperspectral data where hundreds of channels of data are presented. Hyperspectral imagery has a very high component dimensionality (306 channels for MEG here), and to determine their intrinsic dimenionality could be problematic. This is because a high spectral resolution hyperspectral sensor has the capability of uncovering many unknown target sources spectrally that could not be identified by visual inspection or “a prior”. Moreover, when signal sources are relatively weak or noise is not negligible (such as in MEG data), methods based on data covariance (see Section 4.1.1) become difficult. The eigenvalue distribution will be strongly affected by the noise whitening process. Hence, in practice it will be very critical to estimate the noise structure $\mathbf{R}_n$ before the whitening process is applied. However, estimating the noise structure is not easy. In the literature, there exist several categories of methods that have been used in hyperspectral data analysis: (1) noise-adjusted PCA [37] and fast ICA [49, 50]; (2) spectral data explorer [92]; (3) a wavelet-based approach [29] and (4) the most recent methods for high dimensional covariance estimation [15, 18]. The methods from (1), (2) and (4) tend to need more information about the noise structure before the estimation; the methods from (4) also have high computational cost. Here, we assume the MEG sensors are independent such that $\mathbf{R}_n = \text{diag}[\sigma_1^2, \sigma_2^2, ..., \sigma_L^2]$ and focus on the wavelet approach, residual analysis method and Fourier method, respectively, and use them to estimate the noise covariance matrix for our MEG data.

4.2 ESTIMATION OF THE NUMBER OF SIGNAL SOURCES BY VIRTUAL DIMENSIONALITY (VD)

4.2.1 Noise Estimation by Using Wavelet Basis

For convenience, we rewrite the pure signal function $\mathbf{B}_k(\mathbf{J}_t^P)$ by $\bar{Y}_{k,t}$, and the magnetic observation at the k^{th} sensor at time t can be described as

$$Y_{k,t} = \bar{Y}_{k,t} + U_{k,t}, \;\; 1 \leq t \leq T, 1 \leq k \leq L. \tag{4.5}$$

The whitening process of the covariance matrix $\mathbf{R}$ largely relies on the accuracy of noise estimation; we need a robust estimation of the noise. We will use the wavelet coefficients of $Y_{k,t}$ to estimate the noise. In case of the time varying source of the MEG, the estimator is insensitive to the time-varying characteristic of the signal [53]. Since we will be using discrete wavelets, a brief review of the discrete wavelet transformation (DWT) [28] is necessary. Starting from a single basic wavelet $\Psi(t)$, called the mother function, the discrete wavelets are generated as follows

$$\Psi_{m,n}(t) = \frac{1}{\sqrt{a^m}} \Psi(\frac{t - na^m b}{a^m}) \tag{4.6}$$

where $m \in \mathbb{Z}$ is the scale factor, $n \in \mathbb{Z}$ is the translation factor, and $a > 1$, $b > 0$ are real numbers ($\mathbb{R}$). The DWT is the inner-product of the signal $\mathbf{Y}_k$ ($\mathbf{Y}_k = (Y_{k,1}, \cdots, Y_{k,T})$) and the wavelets $\Psi_{m,n}$,

$$\gamma(m,n) =< \mathbf{Y}_k, \Psi_{m,n} > . \tag{4.7}$$

The definition of $\mathbf{Y}_k$ here is different from that in Eq. (3.2). The set of functions $\Psi_{m,n \in \mathbb{Z}}$ is a complete and orthogonal basis in $L^2(\mathbb{R})$. The reconstruction of any signal $Y_{k,t}$ can be obtained by

$$Y_{k,t} = \sum_{m \in \mathbb{Z}} \sum_{n \in \mathbb{Z}} \gamma(m,n) \cdot \Psi_{m,n}(t) = \sum_{m \in \mathbb{Z}} \sum_{n \in \mathbb{Z}} < \mathbf{Y}_k, \Psi_{m,n} > \cdot \Psi_{m,n}(t). \tag{4.8}$$

In theory, to get the wavelet coeffcients $W_{\mathbf{Y}_k}$, we need to do the transform of the data $\mathbf{Y}_k$ by $W_{\mathbf{Y}_k} = \mathcal{W}\mathbf{Y}_k$. The orthogonal wavelet transform matrix $\mathcal{W}$ (formed by the orthogonal wavelet basis, $\Psi_{m,n \in \mathbb{Z}}$) is T by T. In practice, one performs the DWT without explicitly calculating all the wavelet functions. Many fast filtering algorithms based on the filter bank that uniquely correspond to the wavelet of choice are used to do the wavelet transformation. Suppose that the DWT is applied to the vector $\mathbf{Y}_k$ transforming it into a vector $W_{\mathbf{Y}_k}$. The decomposition can be written as

$$W_{\mathbf{Y}_k} = (H^n \mathbf{Y}_k, GH^{n-1}\mathbf{Y}_k, \cdots, GH^2\mathbf{Y}_k, GH\mathbf{Y}_k, G\mathbf{Y}_k) \tag{4.9}$$

where G and H are high-pass and low-pass filters corresponding to the wavelet basis. The high-pass filter G and the low-pass filter H are related (knowing the low-pass filter implies knowing the high-pass filter) and thus together they are known as a quadrature mirror filter [64]. Let $W^j_{\mathbf{Y}_k}$ be the j^{th} element of the vector $W_{\mathbf{Y}_k}$ such that the elements of $W^j_{\mathbf{Y}_k}$ are the wavelet coefficients representing different levels in the wavelet decomposition. To be specific, the wavelet coeffcients of the j^{th} level of decomposition is $GH^{n-j-1}\mathbf{Y}_k$. For example, $G\mathbf{Y}_k$ contains $T/2$ coefficients representing the finest level scale ($(n-1)^{th}$ level). At each level, the high-pass filter produces detailed information coefficients (from G) while the low-pass filter produces coarse approximation coefficients (from H). We note that, for a more complex model one could choose a higher order wavelet, but for simplicity, we have chosen the Daubechies 4 wavelet [28]. The high-pass filter coefficients (g_0, g_1, g_2, g_3) are given by

$$(-0.1294095226, -0.2241438680, 0.8365163037, -0.4829629131)$$

and the low-pass filter coefficients (h_0, h_1, h_2, h_3) are given by

$$(0.4829629131, 0.8365163037, 0.2241438680, -0.1294095226).$$

Each row of the high-pass filter matrix G consists of (g_0, g_1, g_2, g_3) and each row of the low-pass filter matrix H consists of (h_0, h_1, h_2, h_3). The i^{th} detailed information coefficient can be computed by

$$G\mathbf{Y}_k(i) = g_0 Y_{k,2i} + g_1 Y_{k,2i+1} + g_2 Y_{k,2i+2} + g_3 Y_{k,2i+3}.$$

Similarly, the i^{th} coarse approximation coefficient can be computed by

$$H\mathbf{Y}_k(i) = h_0 Y_{k,2i} + h_1 Y_{k,2i+1} + h_2 Y_{k,2i+2} + h_3 Y_{k,2i+3}.$$

It is reasonable to assume that the signal function $\bar{Y}_{k,t}$ (see Eq. (4.5)) is a continuous function and piecewise smooth. Therefore $\bar{Y}_{k,t}$ can be approximated by a polynomial function of degree of M according to Stone-Weierstrass theory [77]

$$\bar{Y}_{k,t} = a_{k,0} + a_{k,1}t + ... + a_{k,M}t^M. \tag{4.10}$$

If $\Psi(t)$ has a vanishing moment $c(c > M)$ ($\int_{-\infty}^{\infty} t^c \Psi(t)dt = 0$, $c = 0, 1, ..., M-1$ and $\int_{-\infty}^{\infty} t^M \Psi(t)dt \neq 0$), then after the discrete wavelet transformation as defined in [28], the signal $\bar{Y}_{k,t}$ is supressed and only the noisy components $U_{k,t}$ are left; that is

$$W_{\mathbf{Y}_k} = W_{\mathbf{U}_k} \tag{4.11}$$

where $\mathbf{U}_k = (U_{k,1}, \ldots, U_{k,T})$; the definition of $\mathbf{U}_k$ here is different from that in Eq. (3.2) in Section 3.1.2. The standard deviation of the noise $\mathbf{U}_k$ can be estimated from the median of the finest scale wavelet coefficients provided that the signal function $\bar{Y}_{k,t}$ is a linear combination of a set of wavelet basis [29].

$$\hat{\sigma}_k \approx \frac{1}{0.6745}\text{Med}(\left|W_{\mathbf{Y}_k}^{n-1}\right|) \tag{4.12}$$

The $W_{\mathbf{Y}_k}^{n-1}$ are the detailed information coefficients (finest scale) of size $T/2$ contained in $W_{\mathbf{Y}_k}$. Med represents the median of the data sequence of absolute value $|W_{\mathbf{Y}_k}^{n-1}|$. The factor 0.6745 is chosen for calibration with the Gaussian distribution. The square of this estimator above is a robust estimator of the variance of noise σ_k^2 at the k^{th} sensor. Finally we have the estimated noise covariance matrix $\hat{\mathbf{R}}_n = \text{diag}[\hat{\sigma}_1^2, \hat{\sigma}_2^2, ..., \hat{\sigma}_L^2]$.

4.2.2 Noise Estimation by Residual Analysis

To overcome the random property that the wavelet method leaves on the noise estimation, which might cause a problem in the eigenvalue distribution of the de-noised data covariance, we will use the noise estimation method developed by [76] based on residual analysis. The decomposion of the sample covariance matrix $\mathbf{R}^*$ can be expressed as $\mathbf{R}^* = \mathbf{D}_L \mathbf{E}_L \mathbf{D}_L^T$, where $\mathbf{D}_L = \text{diag}[\sigma_1^*, \sigma_2^*, ..., \sigma_L^*]$ with $\left\{(\sigma_j^*)^2\right\}_{j=1}^{L}$ being diagonal elements of $\mathbf{R}^*$, and

$$\mathbf{E}_L = \begin{bmatrix} 1 & \rho_{12} & \rho_{13} & \cdots & \rho_{1L} \\ \rho_{21} & 1 & \rho_{23} & \cdots & \rho_{2L} \\ \rho_{31} & \rho_{32} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \rho_{(L-1)L} \\ \rho_{L1} & \rho_{L2} & \cdots & \rho_{L(L-1)} & 1 \end{bmatrix} \tag{4.13}$$

with ρ_{mn} being the correlation coefficient at the $(m, n)^{th}$ entry of $\mathbf{R}^*$. Similarly, the decomposition of the inverse, $(\mathbf{R}^*)^{-1}$, is $(\mathbf{R}^*)^{-1} = \mathbf{D}_{L^{-1}} \mathbf{E}_{L^{-1}} \mathbf{D}_{L^{-1}}^T$, where $\mathbf{D}_{L^{-1}} = \text{diag}[\varsigma_1^*, \varsigma_2^*, ..., \varsigma_L^*]$ with $\left\{(\varsigma_j^*)^2\right\}_{j=1}^{L}$ being diagonal elements of $(\mathbf{R}^*)^{-1}$, and

$$\mathbf{E}_{L^{-1}} = \begin{bmatrix} 1 & \xi_{12} & \xi_{13} & \cdots & \xi_{1L} \\ \xi_{21} & 1 & \xi_{23} & \cdots & \xi_{2L} \\ \xi_{31} & \xi_{32} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \xi_{(L-1)L} \\ \xi_{L1} & \xi_{L2} & \cdots & \xi_{L(L-1)} & 1 \end{bmatrix} \tag{4.14}$$

with ξ_{mn} being the correlation coefficient at the $(m,n)^{th}$ entry of $(\mathbf{R}^*)^{-1}$. This method estimates the noise covariance matrix $\mathbf{R}_n$ by $\hat{\mathbf{R}}_n = \text{diag}[1/(\varsigma_1^*)^2, 1/(\varsigma_2^*)^2, ..., 1/(\varsigma_L^*)^2]$, which is a diagonal matrix, and

$$\varsigma_j^* = \frac{1}{\sqrt{(\sigma_j^*)^2(1 - r_{L-j}^2)}} \tag{4.15}$$

where r_{L-j}^2 is the multiple correlation coefficients of channel $\mathbf{Y}_j$ on the other $L-1$ channels $\{\mathbf{Y}_k\}_{k=1,k\neq j}^L$ from the multiple regression theory. The advantage of using ς_j^* is that ς_j^* removes its correlation on the other ς_j^*s while σ_j^* does not.

4.2.3 Noise Estimation by Using Fourier Basis

The utility of the Fourier transform lies in its ability to analyze a signal in the time domain by its frequency content. The transform works by first translating a function in the time domain into a function in the frequency domain. The signal can then be analyzed for its frequency content because the Fourier coefficients of the transformed function represent the contribution of the complex exponential function at each frequency. The discrete Fourier transform (DFT) relates two finite sequences. In terms of our previous definition for $Y_{k,t}$ in Section 3.1.2 (a sequence indexed by t, k is fixed), the discrete Fourier transform of the sequence $Y_{k,t}$ where $1 \leq t \leq T$ is a sequence of C_r for $r = 0, 1, \ldots, T-1$ defined by

$$C_r = \frac{1}{T}\sum_{t=1}^{T} Y_{k,t}\text{exp}(-\frac{i2\pi rt}{T}). \tag{4.16}$$

The corresponding inverse transform is

$$Y_{k,t} = \sum_{r=0}^{T-1} C_r\text{exp}(\frac{i2\pi rt}{T}). \tag{4.17}$$

The complex numbers C_r are the Fourier coefficients (based on the complex exponential basis functions). Recall $Y_{k,t} = \bar{Y}_{k,t} + U_{k,t}$ for the k^{th} sensor $(1 \leq k \leq L)$, where we have the assumption that the noise $U_{k,t}$ is additive and independent of the signal $\bar{Y}_{k,t}$. When we perform the Fourier transform on the observed $Y_{k,t}$, the Fourier coefficients evaluated by Eq. (4.16) can be considered as a sum of the true Fourier coefficients of the signal C_r^s and the

Fourier coeffcients generated by noise C_r^ϵ; that is $\hat{C}_r = C_r^s + C_r^\epsilon$. The first two moments of the calculated Fourier coefficients $\hat{C}_r$ can be found in [89]

$$\text{Mean}(\hat{C}_r) = C_r^s \quad \text{Var}(\hat{C}_r) = \frac{\sigma_k^2}{T}. \tag{4.18}$$

These formulas give the relation between the variances of the calculated Fourier coefficients and the variance of noise. This means that we can estimate the noise variance by calculating the sample variance of the calculated Fourier coefficients. However, the effect of signal on the calculated Fourier coefficients is not always negligible. To avoid using Fourier coefficients that are largely affected by the signal, the median of the modulus of the Fourier coefficients is used instead of the sample variance of the coefficients. Because the coefficients are Hermitian, we need only use the first half (or first quarter) of the Fourier coefficients (not including coefficients for $r = 0$) in the noise estimation. For each sensor k, if the complex coefficients $\hat{C}_r$ are used, the standard deviation of the noise is estimated by

$$\hat{\sigma}_k \approx M\text{Med}(|\hat{C}_r|, 1 \leq r \leq T/2)/0.6745 \tag{4.19}$$

where $|\hat{C}_r|$ is the modulus of C_r and M is the scale term $\sqrt{T/6}$ (see the APPENDIX A for details). Finally we have the estimated noise covariance matrix $\hat{\mathbf{R}}_n = \text{diag}[\hat{\sigma}_1^2, \hat{\sigma}_2^2, ..., \hat{\sigma}_L^2]$.

4.2.4 Virtual Dimensionality (VD) and Eigensystem Thresholding

Although we can estimate the covariance of the noise ($\hat{\mathbf{R}}_n$) from the data by the three methods above on a relatively accurate basis, in practice, we still face a problem of determining the cutoff threshold between the eigenvalues caused by signals and noise (such as eigenvalues from $\mathbf{R}_{adj}$). In other words, it is difficult to decide when a change between two adjacent eigenvalues is significant or not. Therefore, for real MEG data, using 1 as the threshold to decide the number of significant eigenvalues may not be reliable. To solve this problem, we will be using the concept of virtual dimensionality (VD) [21, 23], which is the minimum number of spectrally distinct signal sources that characterize the hyperspectral data from the perspective of target detection and classification. The idea comes from the remote sensing field where VD provides an effective solution for estimating signal sources in a huge hyperspectral imagery dataset. Because of its similarity to estimating the number of sources in

MEG data, we believe VD is a reasonable choice for our data and can be used after possible adjustment.

The idea of VD is realized by simply calculating the eigenvalues of both the sample correlation and covariance matrices, $\left\{\hat{\lambda}_j\right\}_{j=1}^{L}$ and $\{\lambda_j\}_{j=1}^{L}$, for the j^{th} sensor. A signal source is present if the difference, $\hat{\lambda}_j - \lambda_j$ is positive. Let $\mathbf{C}^*$ (L by L) and $\mathbf{R}^*$ (L by L) be the sample correlation matrix and covariance matrix. Their eigenvalues are ordered and have the following form, $(\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq ... \geq \hat{\lambda}_L)$ and $(\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_L)$, respectively. We expect

$$\hat{\lambda}_j > \lambda_j \qquad j = 1, ..., \text{VD} \tag{4.20}$$

$$\hat{\lambda}_j = \lambda_j \qquad j = \text{VD} + 1, ..., L. \tag{4.21}$$

Estimating VD is usually through a series of Neyman-Pearson tests [21, 23, 44, 1], that is, for $j = 1, ..., L$, we test

$$H_0: \quad z_j = \hat{\lambda}_j - \lambda_j = 0 \tag{4.22}$$

$$H_1: \quad z_j = \hat{\lambda}_j - \lambda_j > 0. \tag{4.23}$$

Since the noise energy represented by $\left\{\hat{\lambda}_j\right\}_{j=1}^{L}$ is the same as the one represented by $\{\lambda_j\}_{j=1}^{L}$, when H_1 is true, it implies there is a signal source contributing to the correlation eigenvalue in addition to noise. Thus, each pair of eigenvalues $\hat{\lambda}_j$ and λ_j can be modeled as random variables under both H_0 and H_1 as

$$p_0(z_j) = p(z_j|H_0) \cong N(0, \sigma_{z_j}^2) \qquad j = 1, ..., L \tag{4.24}$$

$$p_1(z_j) = p(z_j|H_1) \cong N(\mu_j, \sigma_{z_j}^2) \qquad j = 1, ..., L \tag{4.25}$$

where μ_j is unknown and the variances $\sigma_{z_j}^2$ are asymptotically zero [3] given the assumption that $\text{Cov}(\hat{\lambda}_i, \lambda_i) \to 0$, $\text{Var}(\hat{\lambda}_i) \cong 0$ and $\text{Var}(\lambda_i) \cong 0$. When sample size T is large, from [23], we have $\text{Var}(\hat{\lambda}_i) \cong 0$ and $\text{Var}(\lambda_i) \cong 0$, therefore $\text{Cov}(\hat{\lambda}_i, \lambda_i) \to 0$ is guaranteed (Schwarz' Inequality). However, this is not always true when T is not large enough. To avoid this trouble, we will not be working with either $\mathbf{C}^*$ or $\mathbf{R}^*$. To be precise, the Neyman-Pearson test is applied directly on $\mathbf{R}_{adj}$ introduced in Section 4.1.1. The whitening matrix $\mathbf{W}_n$ is obtained by any of the wavelet method, the residual analysis method or the Fourier method (because

$\mathbf{R}_n$ is diagonal, the whitening process defined in Eq. (4.2) is equivalent to $\mathbf{R}_n^{-1/2}\mathbf{R}^*\mathbf{R}_n^{-1/2}$).
Let $\bar{\mathbf{R}}_{adj}$ denote the estimated $\mathbf{R}_{adj}$ using $\hat{\mathbf{R}}_n^{-1/2}\mathbf{R}^*\hat{\mathbf{R}}_n^{-1/2}$, then

$$\bar{\mathbf{R}}_{adj} = \sum_{j=1}^{\mathrm{VD}} \bar{\lambda}_{j,adj}\mathbf{u}_j\mathbf{u}_j^T + \sum_{j=\mathrm{VD}+1}^{L} \bar{\lambda}_{j,adj}\mathbf{u}_j\mathbf{u}_j^T \tag{4.26}$$

where $\left\{\bar{\lambda}_{j,adj}\right\}_{j=1}^{L}$ and $\{\mathbf{u}_j\}_{j=1}^{L}$ are the associated eigenvalues and eigenvectors. Now since after whitening, $\bar{\lambda}_{j,adj} = 1$ for $j = 1, ...\mathrm{VD}$ in theory, the test becomes

$$H_0 : y_j = \lambda_{j,adj} = 1 \tag{4.27}$$

$$H_1 : y_j = \lambda_{j,adj} > 1 \tag{4.28}$$

where $p_0(y_j|H_0) \cong N(1, \sigma_{y_j}^2)$ and $p_1(y_j|H_1) \cong N(\mu_j, \sigma_{y_j}^2)$. Now, from [23] $\sigma_{y_j}^2$ is given by

$$\sigma_{y_j}^2 = \mathrm{Var}(\lambda_{j,adj}) \approx \frac{2\lambda_{j,adj}^2}{T}. \tag{4.29}$$

The VD is based on the target detection and classification point of view. Define a Neyman-Pearson detector as $\delta_{NP}(\hat{\lambda}_j - \lambda_j)$ in the j^{th} binary composite hypothesis test introduced in Eq. (4.20). For a fixed false-alarm probability $p_F = \alpha$, the threshold τ_j $(1 \le j \le L)$ can be obtained by maximizing the detection probability p_D, where p_F and p_D are defined as

$$p_F = \int_{\tau_j}^{\infty} p_0(z)dz \tag{4.30}$$

$$p_D = \int_{\tau_j}^{\infty} p_1(z)dz. \tag{4.31}$$

Thus, according to the Neyman-Pearson lemma, a case of $\hat{\lambda}_j - \lambda_j > \tau_j$ means that $\delta_{NP}(\hat{\lambda}_j - \lambda_j)$ fails the test. The threshold τ_j depends on the index j. Under H_0, $\sigma_{y_j}^2 \approx \frac{2}{T}$, which means we have the same threshold for each $\lambda_{j,adj}$. In order to determine τ_j, we have

$$\int_{\tau_j}^{\infty} p(y_j|H_0)dy_j = \int_{\tau_j}^{\infty} \frac{1}{\sqrt{2\pi\sigma_{y_j}}} e^{-(y_j-1)^2/2\sigma_{y_j}^2} dy_j = \alpha \qquad j = 1, ..., L. \tag{4.32}$$

Therefore, $\tau_j = 1 + \mu_\alpha\sigma_{y_j}$, where μ_α is the $100(1-\alpha)$ of the standard normal distribution. This threshold depends on the false-alarm probability α, the eigenvalue $\lambda_{j,adj}$ and the number of samples T. When the same threshold is chosen for all sensors under the H_0, it depends on the false-alarm probability α and the number of samples T. When the data dimension is large (L is large), the $\lambda_{j,adj}$ computed from spectral decomposition are not necessarily greater than 1, since the whitening process is affected by noise estimation.

4.2.5 AIC, MDL and Malinowski's Method

The common AIC and MDL methods will be also used here for comparison. The following formulas can be found in [101],

$$\mathrm{AIC}(N) = -2\log\left(\frac{\prod_{j=N+1}^{L}\lambda_j^{\frac{1}{L-N}}}{\frac{1}{L-N}\prod_{j=N+1}^{L}\lambda_j}\right)^{(L-N)T} + 2N(2L-N) \tag{4.33}$$

and

$$\mathrm{MDL}(N) = -\log\left(\frac{\prod_{j=N+1}^{L}\lambda_j^{\frac{1}{L-N}}}{\frac{1}{L-N}\prod_{j=N+1}^{L}\lambda_j}\right)^{(L-N)T} + \frac{1}{2}N(2L-N)\log T \tag{4.34}$$

where $(\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_L)$ are the eigenvalues calculated from $\mathbf{R}^*$ and the N is the number of free parameters. In our case, N refers to the number of signal sources. If the noise is independent and identically distributed, the problem of finding the number of signal sources can be achieved by minimizing the following,

$$\text{Number of sources} = \underset{N}{\operatorname{argmin}}\ \mathrm{AIC}(N) \tag{4.35}$$

and similarly

$$\text{Number of sources} = \underset{N}{\operatorname{argmin}}\ \mathrm{MDL}(N). \tag{4.36}$$

Malinowski's method [63], a popular factor analysis method, is also used here for comparison, where an empirical indicator function (EIF) [62] is introduced as a criterion

$$\mathrm{EIF}(N) = \frac{\left(\sum_{j=N+1}^{L}\lambda_j\right)^{1/2}}{T^{1/2}\left(L-N\right)^{3/2}} \tag{4.37}$$

and the number of sources is estimated by

$$.\text{Number of sources} = \underset{N}{\operatorname{argmin}}\ \mathrm{EIF}(N). \tag{4.38}$$

Each of the AIC, MDL and EIF methods tends to overestimate the number of signal sources and rely on the independence and normality assumption. In Sections 4.3 and 4.4, their performance is compared with our methods as well as PCA in our simulation study and a real data application.

4.3 SIMULATION STUDY

Before running our algorithms on a real data set, we tested a simplified case. In this example, 10 dipoles were simulated. The locations of these simulated dipoles are summarized in Table 6. The associated parameters for each dipole such as locations, moments and strength did not vary during the simulation. In other words, each dipole contributed a different but constant signal at the same sensor. However, to work with time-varying dipoles, we applied a different frequency to the magnitudes of each dipole so that we can create a distinct time series for each dipole. In order to work in a known situation, we generated artificial data as follows. The head was modeled by a homogeneous sphere of radius 85 mm. The measurements of the magnetic field were simulated for 45 radially oriented magnetometers, located on the upper half of a sphere with 90 mm radius. The pure magnetic signal produced by each dipole at each sensor was calculated using the Biot-Savart equation. To get a time series for each dipole with unique frequency, the pure magnetic signal was transformed by a sine function with frequency $1, 2, \cdots, 10$. The simulated noise data was normally distributed having constant and independent variance across sensors. The magnetometer data were obtained by adding up the contributions from each dipole at each sensor and simulated noise. The total length of the simulation is 1024 timesteps. We have used the wavelet approach (Daubechies 4 wavelet transformation), the residual analysis method, and the Fast Fourier transform (FFT) in our noise covariance estimation.

We have worked with five datasets with each dataset containing 1, 2, 3, 4 and 8 dipoles, respectively. In each of those datasets having more than one dipole, the magnetic fields of the different dipoles add. Each data set has five different SNRs. Our methods and PCA are tested on data sets where the number of dipoles is small (1, 2 and 3) and number of dipoles is large (4, 6 and 8) under different SNRs. In addition, one more dataset with ten dipoles is also tested against PCA without varying the SNRs.

From Table 7, we can see that when there is only one dipole in our simulated data, all of the methods (NPW, NPR, NWF) and PCA can detect the correct number of dipoles when the SNR is large (e.g., SNR=220 and 110). When the SNR is small (e.g., SNR=75, 55 and 45), PCA tends to detect more dipoles (2, 12 and 23 dipoles) while our methods do not. The

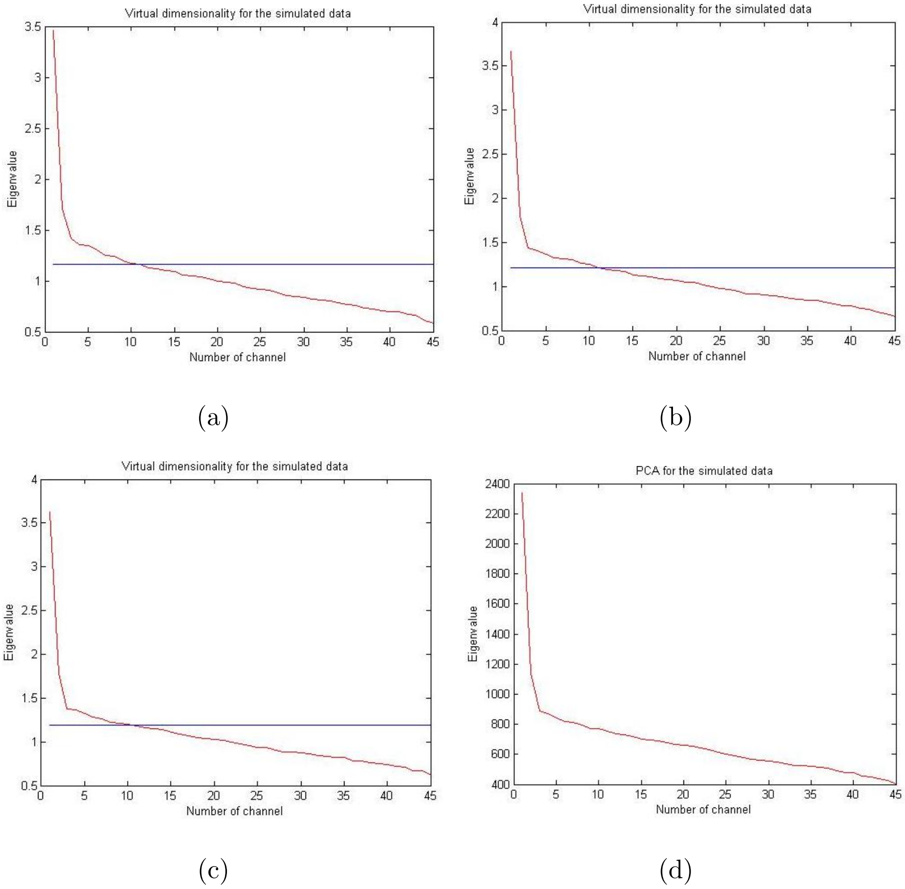
Table 6: Illustration of dipole simulation 3. In this simuation, 10 different dipoles are simulated. The location parameters of each dipole are expressed in terms of spherical coordinates (r,θ,ϕ), where r is radial distance, θ is inclination and ϕ is azimuth. m_1 and m_2 are the dipole moment parameters. s is the strength parameter of a dipole.

Dipole index	1	2	3	4	5	6	7	8	9	10
r (mm)	20	30	40	50	60	70	80	15	10	25
ϕ	$\pi/3$	$\pi/4$	$\pi/5$	$\pi/6$	$\pi/7$	$\pi/5.5$	$\pi/3$	$4\pi/5$	$5\pi/6$	$6\pi/7$
θ	$3\pi/2$	$\pi/3$	$\pi/4$	$3\pi/5$	$\pi/6$	$\pi/4$	$\pi/8$	$3\pi/4$	$4\pi/5$	$5\pi/6$
m_1	2.5	2.6	2.7	2.8	2.9	3.0	3.1	3.2	3.3	3.4
m_2	3.5	3.6	3.7	3.8	3.9	4.0	4.1	4.2	4.3	4.4
s (mA)	5.5	5.6	5.7	5.8	5.9	6.0	6.1	6.2	6.3	6.4

more dipoles that are in the data, the more difficult it is for PCA to detect the correct number of dipoles. PCA comes up with eigenvalues that are very close, thus it tends to find a larger number of dipoles. In particular, when the number of dipoles is two, PCA becomes very sensitive under small SNR (i.e., SNR=110); in contrast, our methods still detect the correct number of dipoles. When the number of dipoles is more than two, PCA is not able to detect the correct number of dipoles under any SNR; however, we notice that when the number of dipoles becomes large, our methods still find about the right number of dipoles. Figure 14 shows the eigenvalue plots for all of our methods and PCA when the true number of dipoles is ten. The PCA method comes up with 39 dipoles (3 very large eigenvalues but needs 39 eigenvalues to achieve the cut-off point of 90%). However, the NPW estimates 10 dipoles (false alarm probability $P_F = 10^{-4}$), the NPR estimates 11 dipoles (false alarm probability $P_F = 10^{-7}$) and the NPF estimates 10 dipoles (false alarm probability $P_F = 10^{-5}$).

Table 7: Comparison of results from NPW (Neyman-Pearson with wavelet), NPR (Neyman-Pearson with residual analysis), NPF (Neyman-Pearson with FFT) and PCA when the number of dipoles is one, two, three, four and eight.

Number of sources	NPW	NPR	NPF	PCA	SNR
1	1	1	1	1	220
1	1	1	1	1	110
1	1	1	1	2	75
1	1	1	1	15	55
1	1	1	1	23	45
2	2	2	2	1	220
2	2	2	2	2	110
2	2	2	2	17	75
2	2	2	2	25	55
2	2	2	2	30	45
3	3	3	3	2	220
3	3	3	3	2	110
3	3	3	3	18	75
3	4	3	3	26	55
3	3	2	2	30	45
4	4	4	4	29	220
4	4	3	3	36	110
4	4	2	3	38	75
4	3	1	3	38	55
4	3	1	3	39	45
8	8	9	8	33	220
8	9	9	9	39	110
8	9	9	8	39	75
8	8	9	9	39	55
8	8	9	9	39	45



(a)

(b)

(c)

(d)

Figure 14: Graphical illustration of NPW, NPR, NPF and PCA when the number of dipoles is ten. (a) Plot of eigenvalues with threshold (horizontal line) used in NPW. (b) Plot of eigenvalues with threshold (horizontal line) used in NPR. (c) Plot of eigenvalues with threshold (horizontal line) used in NPF. (d) Plot of eigenvalues from PCA. NPW(a), NPR(b) and NPF(c) estimate roughly 10 dipoles; PCA(d) estimates about 39 dipoles.

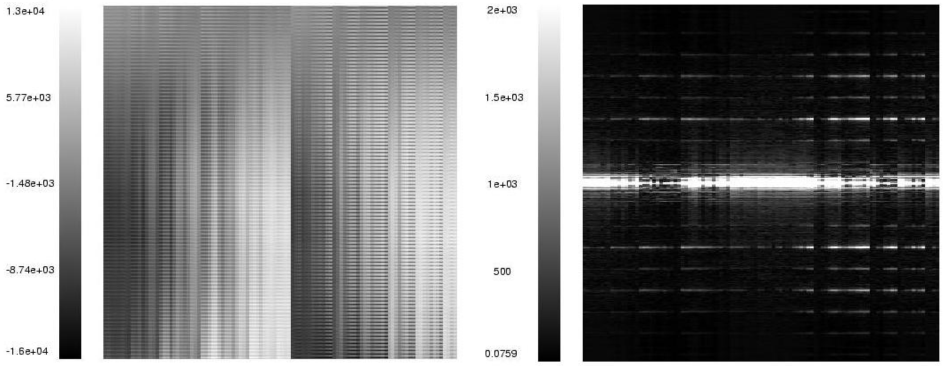
Table 8: Comparison of result from NPW (Neyman-Pearson with wavelet), NPR (Neyman-Pearson with residual analysis) and NPF (Neyman-Pearson with FFT) with PCA, AIC, MDL and EIF when the number of dipoles is two. The first column is the number of dipoles in the simulation data. The second column to seventh column are the number of dipoles estimated from the simulation data by each method. The last column is the corresponding signal-to-noise ratio.

Number of sources	NPW	NPR	NPF	PCA	AIC	MDL	EIF	SNR
2	2	2	2	1	2	2	2-6	220
2	2	2	2	2	2	2	2-6	110
2	2	2	2	17	2	2	2-5	75
2	2	2	2	25	2	2	1-5	55
2	2	2	2	30	2	2	1-5	45

The performance of all the methods are shown in Table 8. As we can see, both the AIC and MDL work as well as our methods in this particular simulation. The number of signal sources by EIF (2-6, 2-5 and 1-5) shows that it is not very easy to pick up the number of free parameters achieving the minimum (used as an estimate of the number of signal sources). We will show in real MEG data application (Section 4.4) since the normality and independence asuumptions of the data are not satisfied, AIC, MDL and EIF methods do not work well in estimating the number of sources (they overestimate the number of signal sources).

4.4 A REAL DATA APPLICATION

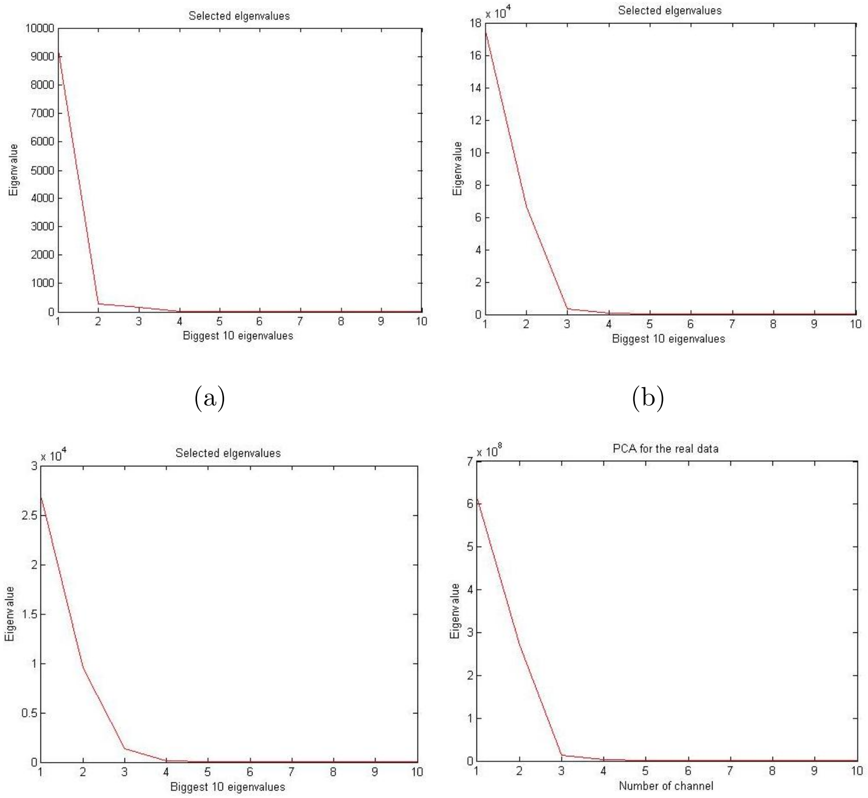
For real MEG data, we cannot clarify the accuracy of our methods since the truth of how many sources are present in the data is unknown. However, it will still be quite interesting to see the performance of our methods on a specific real MEG data where we do know the truth. In the following analysis, a dataset from an empty MEG room will be used; that is, there is no subject in the MEG room. To our knowledge, all the devices in the room that might cause electric potential were turned off but one device was consistently producing energy around 60 Hz. The magnetic field distribution was recorded by a 306-channel system (Elekta-Neuromag) at the Center for Advanced Brain Magnetic Source Imaging (CABMSI) at UPMC Presbyterian Hospital in Pittsburgh. The MEG data at each sensor was acquired at sampling rate of 1000 Hz. A small portion of the dataset of 5000 milliseconds long with only 102 channels was used in our analysis. Those 102 channels were direct magnetic field measurements (the other 204 channels were measuring the change of the magnetic field). Conservatively speaking, there was only one source (60 Hz) or at least one with high frequency in our data. Our attempt is to verify the existence of this high frequency source and hopefully to estimate the number of active sources using our proposed methods on this data.



(a) (b)

Figure 15: Raw data (empty room) and the modulus plot of the data after Fourier transform. (a) The gray scale plot of raw data of 5000 milliseconds. Horizontal axis is time (milliseconds); Vertical axis is the channel number (102 channels in total). (b) The modulus plot of the complex-valued Fourier coefficients of the raw data. Horizontal axis is time (milliseconds); Vertical axis is frequency.

The magnitude of raw data (Figure 15(a)) of the empty room is in a range of -1.6×10^4 fT to 1.3×10^4 fT. We can see that those white lines are equally distant in the modulus plot (Figure 15(b)) of complex Fourier coefficients truncated to 2000 for raw data. This is a clear indication of a periodic source at about 60 Hz in the data. The number of sources estimated by our methods and PCA are in Figure 16. We list the 10 largest eigenvalues from each of the four methods. NPW finds there are three eigenvalues above the threshold. But there is a significantly large eigenvalue out of the three and it is significantly greater than the second one; the second one is significantly greater than the third, so the threshold does not matter too much here. Thus, we report by NPW one or two sources exist in the data (Figure 16(a)). Similarly, we report three dipoles by NPR (Figure 16(b)) and three dipoles by NPF (Figure 16(c)). PCA finds two or three significant sources from the data (Figure 16(d)).



(a)

(b)

(c)

(d)

Figure 16: Graphical illustration of NPW, NPR, NPF and PCA for the empty room data. (a) Plot of the 10 largest eigenvalues from NPW. (b) Plot of the 10 largest eigenvalues from NPR. (c) Plot of the 10 largest eigenvalues from NPF. (d) Plot of the 10 largest eigenvalues from PCA. NPW(a) estimates 2 dipoles; NPR(b) estimates 3 dipoles; NPF(c) estimates three dipoles; PCA(d) estimates three dipoles.

In order to check if the number of sources that our methods detect does include the

60 Hz one, it is necessary for us to run our methods in an environment when the 60 Hz is not available. This means we need to filter the 60 Hz signal from the raw data. In fact, we filtered all frequencies above 50Hz. Figure 17(a) shows the modulus plot of the Fourier coefficients after filtering all frequencies above 50Hz; all the white lines associated with 60 Hz, 120 Hz, 180 Hz and so on disappear. The image after filtering (shown in Figure 17(b)) is reconstructed by the inverse Fourier transform of the real part after filtering. We do not show the imaginary part of the filtered inverse transformed data, because they are all nearly zero (less than 10×10^{-11} fT).

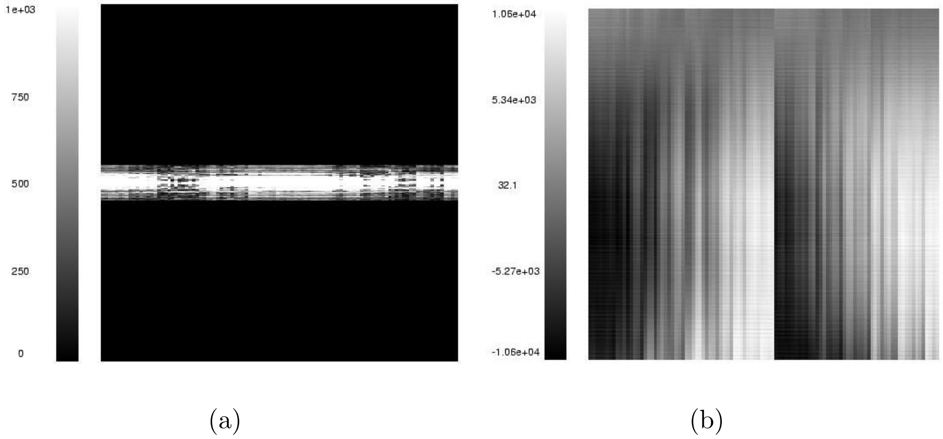
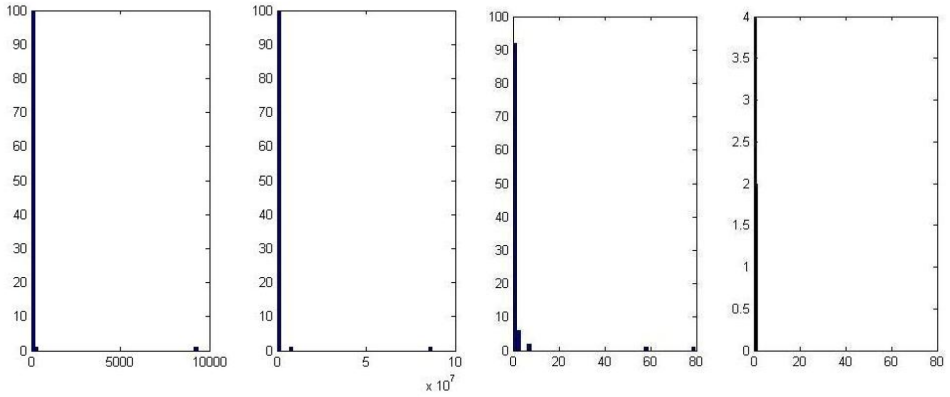


Figure 17: Raw data of empty room after filtering. (a) The modulus plot of the complex-valued Fourier coefficients after filtering all frequencies above 50 Hz truncated to 1000. Horizontal axis is time (milliseconds); Vertical axis is frequency. (b) The gray scale plot of the real part of inverse Fourier transform after filtering coefficients; that is, the data after all coefficients above 50 Hz are zeroed out.

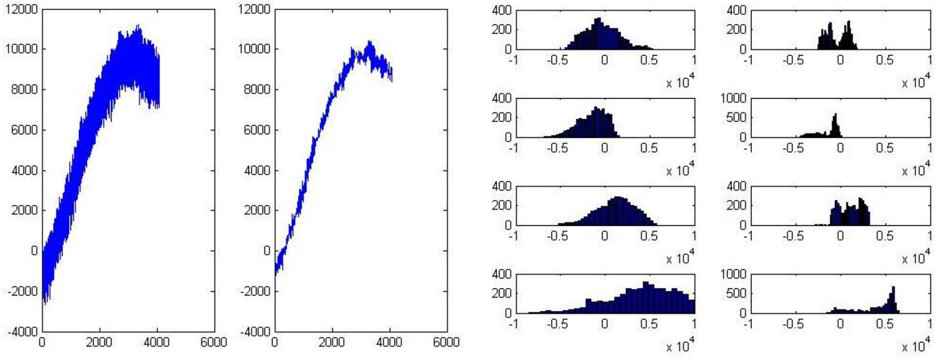
We begin analyzing the real data (with and without filtering) by the NPW method. All the eigenvalues from the filtered data become much larger than 1 (right plot in Figure 18(a)) and they are much larger than the corresponding eigenvalues before filtering (left plot in Figure 18(a)). This makes the NPW not applicable. It is necessary to investigate the eigenvalue distributions for these two datasets. One of the reasons that eigenvalues are very large when one source signal is filtered out lies in the change of the variation in the data; if the covariance $\mathbf{R}_n$ is decreased which is the case after filtering (see Figure 19(a)), we will have smaller eigenvalues $\mathbf{\Lambda}_n$. This makes the whitening matrix $\mathbf{W}_n = \mathbf{\Phi}_n \mathbf{\Lambda}_n^{-1/2}$ actually larger. So eventually, we will have larger eigenvalues $\lambda_{j,adj}$ from $\mathbf{R}_{adj} = \mathbf{W}_n^T \mathbf{R} \mathbf{W}_n$. In addition, the distribution of the eigenvalues $\lambda_{j,adj}$ of $\mathbf{R}_{adj}$ also matters in the situation. It is clear that $\lambda_{j,adj}$ is proportional to a Chi-square random variable with degrees of freedom L-1 if $\mathbf{W}_n$ is not a random variable. However, since we estimate the noise by wavelets, $\mathbf{W}_n$ is a random variable. Therefore, the eigenvalue distribution $\lambda_{j,adj}$ does not follow a Chi-squared distribution. Furthermore, the NPW is affected by the normality assumption. The raw data is not normally distributed (left plot in Figure 19(b)) and the filtered data is much further away from normal (right plot in Figure 19(b)). All of this makes the eigenvalue decomposition problematic.



(a) (b)

Figure 18: (a) Eigenvalue distribution for empty room data by NPW (without filtering (left plot) and filtered (right plot)). There are two very large eigenvalues which are significantly larger than the threshold before filtering (left plot). All the eigenvalues after filtering (in the level of 10^7, right plot) are much larger than the threshold. The scale of the eigenvalues change much before and after filtering. (b) Eigenvalue distribution for simulation data (without filtering (left plot) and filtered (right plot)). There are several eigenvalues which are significantly larger than others before filtering (left plot). Those very large eigenvalues disappear after filtering (right plot). The scale of the eigenvalues does not change much before and after filtering.

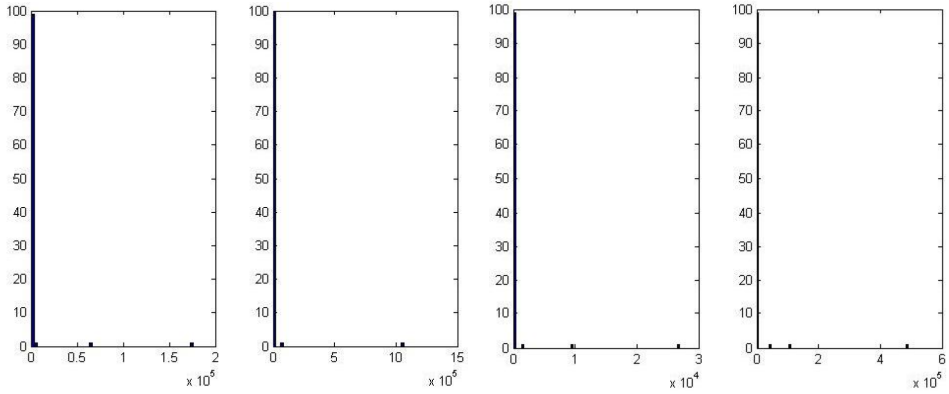
To check our analysis, we did a simulation. We added independent white noise to the simulated signal at each sensor. Figure 18(b) is a plot of the eigenvalue distribution of the simulated data and the filtered simulation data (only white noise). Notice that there are some significantly large eigenvalues (between 60-80 Hz) in the simulation data (left plot in Figure 18(b)), but the eigenvalues become very close to each other and the large eigenvalues disappear when the signal is filtered out (right plot in Figure 18(b)). However, from the eigenvalue distribution plot in Figure 18(a), we only have amplified the eigenvalue scale after filtering; the shape of the eigenvalue distribution before and after filtering does not change much. Since Figure 19(a) already shows the filtered data still has the similar trend as the raw data but with small variation, we get a similar eigenvalue distribution with only the scale changed. This makes us believe there is still another source (the 60 Hz signal is not the only one). For the simulation data, the independent and white noise explains why the eigenvalue distribution does not have much skewness in Figure 18(b).



(a) (b)

Figure 19: (a) Time series plot (channel 102) of the empty room data (with and without filtering). The variation of data before filtering (left plot) is much larger than the data after filtering (right plot). (b) Histogram plot of raw data of four selected channels (before filtering (left plot) and after filtering (right plot)). The data before filtering is skewed but not far from normal; the data after filtering is much further away from normal.

We re-estimated the noise of the empty room data by NPR as well as NPF, and calculated the eigenvalue distribution. The results are shown in Figure 20. Interestingly, we can see that the scale problem of eigenvalues has been solved well by both NPR and NPF. In Figure 20(a), there are three significantly large eigenvalues before the data has been filtered (left plot); there are two significant eigenvalues in the filtered data which matches the fact that we filtered out the 60 Hz signal (right plot). In Figure 20(b) there are three significantly large eigenvalues before the data has been filtered (left plot); there are still three significantly large eigenvalues in the data after we filtered out the 60 Hz signal (right plot).



(a)

(b)

Figure 20: (a) Eigenvalue distribution for empty room data by NPR (without filtering (left plot) and filtered (right plot)). There are three very large eigenvalues which are significantly larger than the threshold before filtering (left plot). There are two eigenvalues which are significantly larger than the threshold after filtering (right plot). The scale of the eigenvalues does not change much before and after filtering. (b) Eigenvalue distribution for empty room data by NPF (without filtering (left plot) and filtered (right plot)). There are three eigenvalues which are significantly larger than others before filtering (left plot). There are three eigenvalues which are significantly larger than the threshold after filtering (right plot). The scale of the eigenvalues does not change much before and after filtering.

A summary of the performance of the different methods (NPR, NPF, PCA, AIC, MDL and EIF) applied on the real MEG data is shown in Table 9. Table 9 shows the results for the data before filtering and also after filtering. Since the normality and independency of the data is not met, the AIC, MDL overestimate the number of signal sources very much

Table 9: Comparison of result from NPR and NPF with PCA, AIC, MDL and EIF for the real MEG data. The top is the data that has not been filtered and the bottom row is the filtered data.

	NPR	NPF	PCA	AIC	MDL	EIF
Before filtering	3	3	3	21	16	14
After filtering	2	3	3	43	29	19

as we expected. The EIF method is better than AIC and MDL but still overestimates the number of signal sources. The NPR still can tell the difference in the number of sources before and after filtering. The NPF has a good performance but cannot tell the change in the number of sources. The PCA method provides a reasonable estimate of the number of sources although it does not in the simulation study. However, the PCA method could not tell the difference in the number of sources for the data before and after filtering.

4.5 DISCUSSION

The determination of signal sources in the MEG data is a very challenging problem. Due to the high-dimensional (306 channels) structure of MEG data, effective methods are lacking for this problem. Regular approaches such as PCA-based methods or methods involving information criteria such as AIC are essentially not helpful. The difficulty lies in the fact that those approaches are simply using the eigenvalue distribution, and the eigenvalues are still mixtures of signal sources and noise in the data. In addition, the MEG signal is much weaker than the noise; it is quite hard to detect the energy that such a signal contributes to the eigenvalues compared to noise.

We treat the MEG data in concept as an analogue of hyperspectral image data from a remote sensing imaging technique. The large number of channels corresponds to the high frequency band across the electromagnetic spectrum in hyperspectral imaging. The number of signal sources in the MEG data is determined by finding the eigenvalues through the eigenthresholding method. In order to achieve this, a set of statistical tests are performed to select the significant eigenvalues which we believe contain energy from this distinct signal source. We need to maximize the power of each statistical test by controlling the false-alarm probability.

The whitening process of the data covariance relies on the accuracy of the noise estimation. To estimate the noise in MEG, we use a wavelet method, a residual analysis method and a Fourier method. In our simulation, we use these methods to estimate the nosie covariance structure from the data where we assume the noise from each sensor is normal and independent. We perform our methods on five datasets where each dataset has 1, 2, 3, 4, and 8 signal sources, respectively. The number of signal sources estimated by our methods is very satisfactory while the PCA, AIC, MDL and EIF approach only works for a few cases and fails for the other situations. Our methods are also tested and compared with other methods on 5 different SNRs for each dataset. Our methods still work very well but the other methods fail when SNR increases.

We also attempt to deal with the real MEG data from an empty MEG room. Our methods (NPW, NPR, and NPF) confirm the existence of a single 60 Hz source in the MEG data. In addition, another one or two potential sources are detected by our methods. We also believe the MEG data is far away from the normal distribution which is an assumption the NPW method relies on. This causes a problem in eigenvalue magnitude for the data when we filter out the 60 Hz known source. The NPR and NPF are used to replace NPW for noise estimation. Both of them cure the scaling problem of the eigenvalues and the result confirms our previous result. We compare our methods (NPR and NPF) with AIC, MDL and EIF methods for both the real data and the data after filtering. Our methods (NPR and NPF) tend to be very robust in either of the two situations. We verify that both AIC and MDL overestimate the number of the sources when the data is not normal or noise is not independent; the EIF method also overestimates the number of sources. The PCA method is not as good as our method and could not tell the change in number of sources when the data is filtered to remove one source.

In conclusion, we have been making an effort to find a way of estimating the number of signal sources in the MEG data. The number of sources from our methods are the number of signals that are spectrally distinct. One advantage of using our methods is that we might possibly detect the hidden signal sources that are different in frequency. Our methods outperform others on both the simulation and the real data; in practice, since the data is always very complicated (i.e., far from normal), our methods can be used as a reference.

5.0 FUTURE WORK: REAL-TIME ANALYSIS OF THE MEG DATA

5.1 REAL-TIME ANALYSIS OF THE MEG IMAGING

The dissertation is essentially about studying the inverse problem in MEG in which theoretical methodology is developed. The real data analysis needs a sophisticated computing approach due to its high dimensionality and extremely large size. A PVM scheme was inspired and successfully implemented in the thesis in order to run our algorithms for MEG data; this scheme permits a heterogeneous collection of Unix and/or Windows computers hooked together to be used as a single large parallel computer so that theoretically the time spent for a task can be reduced at most by the same times as the number of computers. To investigate the brain activity for much longer time (30 minutes - 1 hour), an even more challenging computational problem must be faced; if we want to accomplish the real-time analysis of this incredibly massive data. We plan to replace the PVM structure (Master-Worker) in a multiple central processing unit (CPU) system with the most recent programmable graphic processor units (GPUs). To better understand the brain activity in real-time (at 1 millisecond temporal resolution), many hidden activities might be explored but the computational task is cost-prohibitive. A supercomputer's power might be a choice for this purpose but supercomputer time is also pricey. Hence, a more practical approach is desired.

More specifically, the discrepancy in floating-point capability between the CPU and the GPU is that the GPU is specialized for highly parallel data processing rather than for data caching and flow control as in the case of the CPU. The GPUs have very high ratio of arithmetic operations to memory operations and the same program is executed on many data elements in parallel. Both multicore CPUs and manycore GPUs are parallel systems and their parallelism continues to scale with Moore's law. However, to develop application software that transparently scales its parallelism to leverage the increasing number of processor cores is hard. The GPU system that we will use is the newly developed NVIDIA's compute unified device architecture (CUDA) which transparently scales its parallelism to leverage the increasing number of processor cores while multicore CPUs do not. To run our model for 1-hour data, we will rewrite our previous LINUX program in CUDA and implement it in GPUs with each one having 480 cores. The source distribution in 3D within the brain can be seen in real-time (1/1000 sec) on a personal computer with CUDA-enabled GPUs and this will help significantly in understanding brain activity.

CUDA computing scheme is the latest computing scheme in which a massive parallel computing architecture enables dramatic increases in computing performance by harnessing the power of the GPUs. This matches perfectly with our interest in parallel computing, and most importantly, this high degree of parallelism can be achieved by CUDA GPUs on a desktop computer, whereas CPUs cannot. Using the result of the pilot study in Section 3.4, for a one-hour experiment, we need almost three days to run our program by the PVM scheme (in terms of parallel CPUs). Now by CUDA, we can roughly reduce the time spent on computing MEG data of one hour long to less than 15 minutes. This is a very appealing improvement. We have created a set of Linux codes for our PVM program. Moving from PVM to a CUDA program requires modifying the C program that we have, but this is very natural without too much difficulty. There are functions and procedures based on C language in CUDA programming language which are analogous to the C extensions of PVM. Thus, we will be able to implement our program in CUDA very quickly and run it on GPU. The improvement is not only good for the thesis research, but also helpful for other brain imaging reseachers.

We used PVM as an illustration to speed up the computation for the 2000-timestep case (see pilot study in Section 3.4) where PVM allows a computing paradigm for a parallel networking of computers. In our PVM structure (Master-Worker model), there are several worker programs performing tasks in parallel and a master program collecting the outcomes from each worker. Each task is to separately compute partial sample paths for all the timesteps. There is no parallelism in time. The PVM speed is mainly influenced by hardware and software components of network and I/O systems. It also depends on the number of worker programs; for instance, adding too many parallel workers does not enhance the velocity where most of the time is wasted on the communications among the workers. The parallelism of PVM (CPU-based) is not appropriate for situations, where data of much more timesteps is included; thus, PVM is not helpful for real MEG data.

The idea of CUDA is simply to exploit computing resources (namely cores) as much as possible. There are hundreds of cores in NVIDIA's modern GPUs (i.e., 100s of processor cores per GPU). The speed of the GPU increases at a much higher rate as compared to the CPU and this makes the GPU as a co-processor for handling a large number of calculations per second. After rewriting our Linux code for the MEG analysis in CUDA, the program will be bifurcated into two portions: one portion is delivered to CPU (because CPU is best for such tasks), while the other portion, involving extensive calculations, is delivered to the GPUs that execute the code in parallel (CUDA exposes a fast-shared memory region). Figure 21 shows the processing flow of CUDA: 1. Copy data from main memory to GPU memory; 2. CPU instructs the process to GPU; 3. GPU execute parallel in each core; 4. Copy the result from GPU memory to main memory.

5.2 A NSF-FUNDED PROJECT

As shown above, to reduce the real time elapsed of running the MEG data in the pilot study, we increased the number of worker stations and tasks in PVM. The typical MEG experiment lasts 30 minutes to 1 hour; the computational issue increases to the point that computing 3.6 million milliseconds of data is required. Although the user CPU time increases roughly linearly in the number of timesteps which is significantly long, the real time elapsed is even much longer since it is also affected by other programs and it includes time spent in memory, I/O and other resources. The PVM scheme is apparently not capable if the real-time MEG data analysis is of interest. Due to the nature of the computation that each worker does the same work in parallel, a GPU computing architecture is strongly needed. A GPU can be regarded as a many-cores processor supporting numerous fine-grain threads.

Consequently, previous GPU applications were largely in nature stream processing, which performs identical operations onto each element of the input arrays. CUDA provides sets of on-chip, fast shared memories for data exchange between threads, as well as flexible access to the device memory. This in theory greatly broadens the scope of application kernels that can be effectively executed on CUDA GPUs provided they exhibit substantial parallelism. A CUDA GPU has a set of streaming multiprocessors (SM) with each SM consisting of many processor cores called sreaming processors (SP). CUDA GPUs are single instruction multiple data (SIMD) processors which execute from the same instruction stream on each SP. Thread Block is a group of threads, where they are executed on the same SM so that data exchange between the threads is possible using the shared memory of the SM. The current CUDA GPUs are based on the same architecture, but vary in different architectural parameters.

Table 10 gives a few specifications of NVIDIA GPUs with different parameters (e.g., there are 480 cores in one GPU of model GeForce GTX 480). The computing environment of CUDA-enabled GPUs that we will be using is a 4-GPU Tesla Personal Supercomputer (see Table 11 in APPENDIX B for details). This desktop computer has more than 1600 cores each equivalent to the cluster level computer (300 times faster than standard PCs and workstations). In order to compile our program, we have been learning the CUDA programming (C extension) for parallel computing. To be exact, our former Master-Worker model (PVM scheme) assigns tasks for each worker program in parallel; in our expected CUDA program, even parameters (such as 6 source parameters) within each worker are going to be computed in parallel. The parallel scheme also applies on a multi-source model where computation of each source is achieved in parallel as well as the parameters within each task. We will have the program run on this CUDA computer at the end of our NSF grant period.

5.3 IMPACT AND DISSEMINATION OF THE RESEARCH

The proposed research will contribute to the neuroscience community by facilitating our understanding of brain function using various imaging modalities. Any algorithm that performs efficiently using GPU computing would also be a worthwhile contribution to the field of source localization. The proposed source localization algorithm used to estimate time varying sources could be easily modified for use with EEG which can be used for practical brain-computer interfaces; the real-time analysis can also be achieved by CUDA in similar experiments that use fMRI. These two modalities that used to appear to be independent of each other now can actually work together and will provide more information about the brain. Programming using GPU hardware could encourage other scientists to do likewise with their problems. This CUDA idea may allow these sophisticated computational algorithms to be performed in real-time opening up applications in scientific computing such as three-dimensional Fourier transformations applied to extremely large datasets, finding solutions of massive sets of diffierential equations, and so on. Our work may also increase the visibility of GPU computing for biologists and computer scientisits more generally. CUDA provides a very affordable package that works in a high degree of parallelism on desktop computers. Consequently, it becomes possible for experimenters to test their experimental designs in advance of experimentation without having to leave their laboratories. The benefit of increasing the popularity and use of GPU computing is motivating the update of the computing environment within universities and institutions.

The proposed research will produce a set of computer programs based on the original Linux code for the MEG analysis on the CUDA machine. The CUDA program will be posted on my personal website at the end of the grant period. I will make this program available for other researchers who might be interested in MEG analysis. For real-time analysis, the localization algorithm written in the CUDA program could be easily modified for use with EEG, so that people in the EEG field can have further investigation based on the MEG code. The framework of the MEG CUDA program will be helpful for writing other algorithms that perform efficiently using GPU computing. The CUDA program is expected to be embedded in the FIASCO (Functional Image Analysis Software - Computational Olio) tool (http://www.stat.cmu.edu/~fiasco/). A final report as well as a manuscript of the program will be produced at the end of the research.

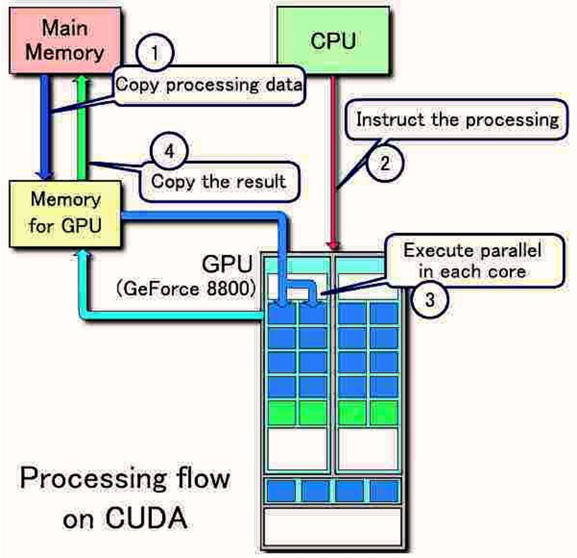


Figure 21: Illustration of the processing flow of CUDA. Above picture taken from wikipedia http://en.wikipedia.org/wiki/File:CUDA_processing_flow_(En).PNG, authored by Tosaka

Table 10: Specifications of NVIDIA GeForce, Quadro and Telsa series GPUs.

Type	Model	Cores	Clock	GFLOPS	Capability	Interface	Bandwith
GeForce	GTX 480	480	1.4	1344	1.5 Gbyte	384	177.4
Quadro	Quadro 6000	448	1.3	933	4 Gbyte	512	102
Telsa	C2070	480	1.15	515	6 Gbyte	384	144

APPENDIX A

DETERMINING THE SCALE TERM IN THE NOISE ESTIMATION BY USING FOURIER BASIS

In Eq. (4.19), when the complex coefficients $\hat{C}_r$ are used for the noise estimation, the scale term is chosen $M = \sqrt{T/6}$. To find an exact scale term in Eq. (4.19) is difficult due to the median operation taken on $|\hat{C}_r|, 1 \leq r \leq T/2$. Since we prefer to use complex coefficients, the empirical scale term $M = \sqrt{T/6}$ is derived by the following procedure:

1. Draw 10^5 samples from $N(0, 1)$ to represent the data (only noise).

2. Set the standard deviation for the real part coefficients ($\sigma_r = \sqrt{1 \times 2/10^5} = 0.0045$) and imaginary part coefficients ($\sigma_i = \sqrt{1 \times 2/10^5} = 0.0045$).

3. Draw 10^5 samples from $N(0, \sigma_r^2)$ to represent the real part coefficients and draw 10^5 samples from $N(0, \sigma_i^2)$ to represent the imaginary part coefficients (the coefficients from the Fourier transform of $N(0, 1)$ are still normal but not $N(0, 1)$).

4. Calculate the modulus of the simulated coefficients, compute the median ($M = 0.0053$) of the first half of them except the first one ($1 \leq r \leq T/2$) and then devided by 0.6745 (M/0.6745=0.0078).

5. Compute the scale term from the known standard devation of the data, that is $10^5/(1/0.0078)^2 \approx 6$.

We used this scale term on all the five simulated datasets in Section 4.3 when we knew the true noise variance. In any of these cases, the scale term worked very well. We also used this scale term in the NPF on our real data in Section 4.4.

APPENDIX B

TECHNICAL SPECIFICATIONS OF THE NSF 4-TELSA WORKSTATION

Table 11: The workstation consists of 4 Tesla GPUs (Telsa C2050) with 2 CPUs and a motherboard. Technical Specifications of Colfax CXT5000 Personal Supercomputer (PSC). The first column is the name of each item. The second column is the specification. The last column is the price of each item in the budget.

Item	Model	Price
Motherboard	Colfax CXT5000 Personal Supercomputer Base Platform	1595 USD
Primary CPU	Intel Xeon DP Quad Core L5530 2.26Ghz	636 USD
Secondary CPU	Intel Xeon DP Quad Core L5530 2.26Ghz	636 USD
GPU 1	NVIDIA Telsa C2050 Computing Processor	2277 USD
GPU 2	NVIDIA Telsa C2050 Computing Processor	2277 USD
GPU 3	NVIDIA Telsa C2050 Computing Processor	2277 USD
GPU 4	NVIDIA Telsa C2050 Computing Processor	2277 USD
Operation System	Scientific Linux	0 USD
Total		11975 USD

BIBLIOGRAPHY

[1] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

[2] S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

[3] T. W. Anderson. *An Introduction to Multivariate Statistical Analysis.* New York: Wiley, second edition, 1984.

[4] C. Andrieu and E. Moulines. On the ergodicity properties of some adaptive Markov chain Monte Carlo algorithms. *Annals of Applied Probability*, 16(3):1462–1505, 2006.

[5] C. Andrieu and C. P. Robert. Controlled MCMC for optimal sampling. Technical report, Department of Mathematics at the University of Bristol, 2001.

[6] Y. F. Atchade and J. S. Rosenthal. On adaptive Markov chain Monte Carlo algorithms. *Bernoulli*, 11(5):815–828, 2005.

[7] X. Bai and B. He. On the estimation of the number of dipole sources in EEG source localization. *Clinical Neurophysiology*, 116:2037–2043, 2005.

[8] X. Bai and B. He. Estimation of number of independent brain electric sources from the scalp EEGs. *IEEE Transactions on Biomedical Engineering*, 53(10):1883–1892, 2006.

[9] Y. Bai. Simultaneous drift conditions for adaptive Markov chain Monte Carlo algorithms. Technical report, Department of Statistics at the University of Toronto, 2009.

[10] S. Baillet and L. Garnero. A Bayesian approach to introducing anatomo-functional priors in the EEG/MEG inverse problem. *IEEE Transactions on Biomedical Engineering*, 44(5):374–385, 1997.

[11] A. K. Barros, R. Vigario, V. Jousmaki, and N. Ohnishi. Extraction of event-related signals from multichannel bioelectrical measurements. *IEEE Transactions on Biomedical Engineering*, 47(5):583–588, 2000.

[12] C. Bertrand, M. Ohmi, R. Suzuki, and H. Kado. A probabilistic solution to the MEG inverse problem via MCMC methods: the reversible jump and parallel tempering algorithms. *IEEE Transactions on Biomedical Engineering*, 48(5):533–542, 2001.

[13] C. Berzuini, N. G. Best, W. R. Gilks, and C. Larizza. Dynamic conditional independence models and Markov chain Monte Carlo methods. *Journal of the American Statistical Association*, 92(440):1403–1412, 1997.

[14] J. Besag and C. Kooperberg. On conditional and intrinsic autoregressions. *Biometrika*, 82(4):733–746, 1995.

[15] P. J. Bickel and E. Levina. Regularized estimation of large covariance matrices. *Annals of Statistics*, 36(1):199–227, 2008.

[16] P. Bouthemy and E. Francois. Motion segmentation and qualitative dynamic scene analysis from an image sequence. *International Journal of Computer Vision*, 10(2):157–182, 1993.

[17] E. Brookner. Phased-array radar. *Scientific American*, 252(2):94–102, 1985.

[18] T. Cai, W. Liu, and X. Luo. A constrained l_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494):594–607, 2011.

[19] B. P. Carlin, N. G. Polson, and D. S. Stoffer. A Monte Carlo approach to nonnormal and nonlinear state-space modeling. *Journal of the American Statistical Association*, 87(418):493–500, 1992.

[20] C. K. Carter and R. Kohn. On Gibbs sampling for state space models. *Biometrika*, 81(3):541–553, 1994.

[21] C.-I. Chang. *Hyperspectral Imaging: Techniques for Spectral Detection and Classification.* Springer, first edition, 2003.

[22] C.-I. Chang and Q. Du. Interference and noise-adjusted principal components analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 37(5):2387–2396, 1999.

[23] C.-I. Chang and Q. Du. Estimation of number of spectrally distinct signal sources in hyperspectral imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 42(3):608–619, 2004.

[24] D. Cohen. Magnetoencephalography: evidence of magnetic fields produced by alpha rhythm currents. *Science*, 161(3843):784–786, 1968.

[25] S. S. Dalal, K. Sekihara, and S. S. Nagarajan. Modified beamformers for coherent source region suppression. *IEEE Transactions on Biomedical Engineering*, 53(7):1357–1363, 2006.

[26] A. M. Dale, A. K. Liu, B. R. Fischl, R. L. Buchner, J. W. Belliveau, J. D. Lewine, and E. Halgren. Dynamic statistical parametric mapping: combining fMRI and MEG for high-resolution imaging of cortical activity. *Neuron*, 26(1):55–67, 2000.

[27] A. M. Dale and M. I. Sereno. Improved localization of cortical activity by combining EEG and MEG with MRI cortical surface reconstruction: A linear approach. *Journal of Cognitive Neuroscience*, 5(2):162–176, 1993.

[28] I. Daubechies. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, first edition, 1992.

[29] D. L. Donoho and I. M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994.

[30] J. Friedman, T. Hastie, H. Hofling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

[31] J. H. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

[32] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press: New York, second edition, 1990.

[33] D. Gamerman. Markov chain Monte Carlo for dynamic generalised linear models. *Biometrika*, 85(1):215–227, 1998.

[34] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, and V. S. Sunderam. *PVM: Parallel Virtual Machine: A Users' Guide and Tutorial for Network Parallel Computing (Scientific and Engineering Computation)*. The MIT Press, 1994.

[35] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984.

[36] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings-F Radar and Signal Processing*, 140(2):107–113, 1993.

[37] A. Green, M. Berman, P. Switzer, and M. Craig. A transformation for ordering multispectral data in terms of imagequality with implications for noise removal. *IEEE Transactions on Geoscience and Remote Sensing*, 26(1):65–74, 2001.

[38] D. J. Griffiths. *Introduction to Electrodynamics*. Prentice Hall, third edition, 1999.

[39] J. Gross and A. Ioannides. Linear transformations of data space in MEG. *Physics in Medicine and Biology*, 44(8):2081–2097, 1999.

[40] P. Grunwald, M. A. Pitt, and I. J. Myung. *Advances in Minimum Description Length: Theory and Applications*. MIT Press, 2005.

[41] H. Haario, E. Saksman, and J. Tamminen. An adaptive Metropolis algorithm. *Bernoulli*, 7(2):223–242, 2001.

[42] M. Hamalainen, R. Hari, R. Ilmoniemi, J. Knuutila, and O. V. Lounasmaa. Magnetoencephalography—theory, instrumentation, and applications to noninvasive studies of signal processing in the human brain. *Reviews of Modern Physics*, 65:413–497, 1993.

[43] M. S. Hamalainen and R. J. Ilmoniemi. Interpreting magnetic fields of the brain: minimum norm estimates. *Medical and Biological Engineering and Computing*, 32(1):35–42, 1994.

[44] J. C. Harsanyi. *Detection and Classification of Subpixel Spectral Signatures in Hyperspectral Image Sequencess*. PhD thesis, University of Maryland, Baltimore County, 1993.

[45] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[46] S. Haykin, J. H. Justice, N. L. Owsley, J. L. Yen, and A. C. Kak. *Array Signal Processing*. Prentice-Hall, 1985.

[47] H. Helmholtz. Ueber einige gesetze der vertheilung elektrischer strome in korperlichen leitern mit anwendung auf die thierisch-elektrischen versuche. *Annals of Physics and Chemistry*, 89(6):211–233, 1853.

[48] M. X. Huang, C. Aine, L. Davis, J. Butman, R. Christner, M. Weisend, J. Stephen, J. Meyer, J. Silveri, M. Herman, and R. R. Lee. Sources on the anterior and posterior banks of the central sulcus identified from magnetic somatosensory evoked responses using multistart spatiotemporal localizations. *Human Brain Mapping*, 11(2):59–76, 2000.

[49] A. Hyvarinen. A family of fixed-point algorithms for independent component analysis. Technical report, Helsinki University of Technology, Laboratory of Computer and Information Sciencel, 1996.

[50] A. Hyvarinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):624–634, 1999.

[51] S. Ikeda and K. Toyama. Independent component analysis for noisy data–MEG data analysis. *Neural Networks*, 13(10):1063–1074, 2000.

[52] K. Jerbi, S. Baillet, J. C. Mosher, G. Nolte, L. Garnero, and R. M. Leahy. Localization of realistic cortical activity in MEG using current multipoles. *Neuroimage*, 22(2):779–793, 2004.

[53] I. M. Johnstone. Wavelet shrinkage for correlated data and inverse problems: adaptivity results. *Statistica Sinica*, 9(1):51–83, 1998.

[54] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.

[55] L. Knorr-Held. Conditional prior proposals in dynamic models. *Scandinavian Journal of Statistics*, 26(1):129–144, 1999.

[56] T. R. Knosche, E. M. Berends, H. R. A. Jagers, and M. J. Peters. Determining the number of independent sources of the EEG: A simulation study on information criteria. *Brain Topography*, 11(2):111–124, 1998.

[57] J. Kybic, M. Clerc, O. Faugeras, R. Keriven, and T. Papadopoulo. Generalized head models for MEG/EEG: boundary element method beyond nested volumes. *Physics in Medicine and Biology*, 51(5):1333–1346, 2006.

[58] J. Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6(2):113–119, 1996.

[59] J. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer-Verlag, 2001.

[60] J. Liu and R. Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998.

[61] S. Makeig, T. P. Jung, A. J. Bell, D. Ghahremani, and T. J. Sejnowski. Blind separation of auditory event-related brain responses into independent components. *Proceedings of the National Academy of Sciences*, 94(20):10979–10984, 1997.

[62] E. R. Malinowski. Determination of the number of factors and the experimental error in a data matrix. *Analytical Chemistry*, 49(4):612–617, 1977.

[63] E. R. Malinowski. Theory of error in factor analysis. *Analytical Chemistry*, 49(4):606–612, 1977.

[64] S. G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989.

[65] J. Mattout, C. Phillips, W. Penny, M. Rugg, and K. Friston. MEG source localization under multiple constraints: An extended Bayesian framework. *NeuroImage*, 30(3):753–767, 2006.

[66] J. C. Mosher and R. M. Leahy. Recursive music: A framework for EEG and MEG source localization. *IEEE Transactions on Biomedical Engineering*, 45(11), 1998.

[67] J. C. Mosher and R. M. Leahy. Source localization using recursively applied and projected (rap) MUSIC. *IEEE Transactions on Signal Processing*, 47(2):332–340, 1999.

[68] J. C. Mosher, P. S. Lewis, and R. M. Leahy. Multiple dipole modeling and localization from spatio-temporal MEG data. *IEEE Transactions on Biomedical Engineering*, 39(6):541–557, 1992.

[69] H. Oster and Y. Rudy. The use of temporal information in the regularization of the inverse problem of Electrocardiography. *IEEE Transactions on Biomedical Engineering*, 39(1):65–75, 1992.

[70] W. Ou, M. S. Hamalainen, and P. Golland. A distributed spatio-temporal EEG/MEG inverse solver. *Neuroimage*, 44(3):932–946, 2009.

[71] R. D. Pascual-Marqui, C. M. Michel, and D. Lehmann. Low resolution electromagnetic tomography: a new method for localizing electrical activity in the brain. *International Journal of Psychophysiology*, 18(1):49–65, 1994.

[72] C. Phillips, M. D. Rugg, and K. J. Friston. Anatomically informed basis functions for EEG source localization: combining functional and anatomical constraints. *NeuroImage*, 16(3):678–695, 2002.

[73] J. W. Phillips, R. M. Leahy, and J. C. Mosher. Dynamic MEG imaging of focal neuronal sources. In *Proceedings of IEEE Nuclear Science Symposium and Medical Imaging Conference*, Anaheim, CA , USA, 1996.

[74] F. Pulvermuller, Y. Shtyrov, and R. Ilmoniemi. Spatiotemporal dynamics of neural language processing: an MEG study using minimum-norm current estimates. *NeuroImage*, 20(2):1020–1025, 2003.

[75] J. A. Richards. *Remote Sensing Digital Image Analysis: An Introduction.* New York: Springer-Verlag, second edition, 1993.

[76] R. E. Roger and J. F. Arnold. Reliably estimating the noise in AVIRIS hyperspectral images. *International Journal of Remote Sensing*, 17(10):1951–1962, 1996.

[77] W. Rubin. *Principles of Mathematical Analysis.* McGraw-Hill Science, third edition, 1976.

[78] E. Saksman and M. Vihola. On the ergodicity of the adaptive Metropolis algorithms on unbounded domains. *The Annals of Applied Probability*, 20(6):2178–2203, 2010.

[79] J. Sarvas. Basic mathematical and electromagnetic concepts of the biomagnetic inverse problem. *Physics in Medicine and Biology*, 32(1):11–22, 1984.

[80] M. Sato. Online model selection based on the variational Bayes. *Neural Computation*, 13(7):1649–1681, 2001.

[81] M. Sato, T. Yoshioka, S. Kajihara, K. Toyama, N. Goda, K. Doya, and M. Kawato. Hierarchical Bayesian estimation for MEG inverse problem. *NeuroImage*, 23(3):806–826, 2004.

[82] H. M. Schey. *Div, Grad, Curl, and All That: An Informal Text on Vector Calculus.* W. W. Norton & Company, forth edition, 1996.

[83] D. M. Schmidt, J. S. George, and C. C. Wood. Bayesian inference applied to the electromagnetic inverse problem. *Human Brain Mapping*, 7(3):195–212, 1999.

[84] R. O. Schmidt. Multiple emitter location and signal parameter estimation. *IEEE Transactions on Antennas and Propagation*, 34(3):276–280, 1986.

[85] K. Sekihara, S. S. Nagarajan, D. Poeppel, A. Marantz, and Y. Miyashita. Reconstructing spatio-temporal activities of neural sources using an MEG vector beamformer technique. *IEEE Transactions on Biomedical Engineering*, 48(7):760–771, 2001.

[86] K. Sekihara, D. Poeppel, and Y. Miyashita. Application of eigenspace beamformer to virtual depth-electrode measurement using MEG. In *Proceedings of the first joint BMES/EMBS conference*, Atlanta, GA , USA, 1999.

[87] N. Shephard and M. K. Pitt. Likelihood analysis of non-gaussian measurement time series. *Biometrika*, 84(3):653–667, 1997.

[88] R. Srinivasan. *Importance Sampling: Applications in Communications and Detection.* Springer-Verlag, first edition, 2002.

[89] L. N. Thibos. *Fourier Analysis for Beginners.* Visual Sciences Group, third edition, 2003.

[90] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[91] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill Posed Problems.* Vh Winston, 1977.

[92] T.-M. Tu, P. S. Huang, and P.-Y. Chen. Blind separation of spectral signatures in hyperspectral imagery. *IEE Proceedings Vision, Image and Signal Processing*, 148(4):217–226, 2001.

[93] K. Uutela, M. Hamalainen, and E. Somersalo. Visualization of Magnetoencephalographic data using minimum current estimates. *NeuroImage*, 10(2):173–180, 1999.

[94] B. D. Van Veen, J. Joseph, and K. Hecox. Localization of intra-cerebral sources of electrical activity via linearly constrained minimum variance spatial filtering. In *Proceedings of IEEE Sixth Signal Processing Workshop on Statistics and Signal Array Processing*, Victoria, BC , Canada, 1992.

[95] B. D. Van Veen, W. van Drongelen, M. Yuchtman, and A. Suzuki. Localization of brain electrical activity via linearly constrained minimum variance spatial filtering. *IEEE Transactions on Biomedical Engineering*, 44(9):867–880, 1997.

[96] S. Vanni and K. Uutela. Foveal attention modulates responses to peripheral stimuli. *Journal of Neurophysiology*, 83(4):2443–2452, 2000.

[97] R. N. Vigario. Extraction of ocular artefacts from EEG using independent component analysis. *Electroencephalography and Clinical Neurophysiology*, 103(3):395–404, 1997.

[98] J. Vrba and S. E. Robinson. Linearly constrained minimum variance beamformers, synthetic aperture magnetometry, and MUSIC in MEG applications. In *Proceedings of the Thirty-Fourth Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA , USA, 2000.

[99] L. J. Waldorp, H. M. Huizenga, R. P. P. P. Grasman, K. B. E. Broker, J. C. D. Munck, and P. C. M. Molenaar. Model selection in electromagnetic source analysis with an application to VEF's. *IEEE Transactions on Biomedical Engineering*, 49(10):1121–1129, 2002.

[100] L. J. Waldorp, H. M. Huizenga, A. Nehorai, R. P. P. P. Grasman, and P. C. M. Molenaar. Model selection in spatio-temporal electromagnetic source analysis. *IEEE Transactions on Biomedical Engineering*, 52(3):414–420, 2005.

[101] M. Wax and T. Kailath. Detection of signals by information theoretic criteria. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 33(2):387–392, 1995.

[102] C. Yang. Recurrent and ergodic properties of adaptive MCMC. Technical report, Department of Mathematics at the University of Toronto, 2007.

[103] Z. Yao and W. F. Eddy. Statistical approaches to estimating the number of signal sources in Magnetoencephalography. Technical report, Department of Statistics at the University of Pittsburgh, 2010.

[104] H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.