33rd Annual International Conference of the IEEE EMBS
Boston, Massachusetts USA, August 30 - September 3, 2011

Near-real-time connectivity estimation for multivariate neural data

Anne C. Smith, Christopher P. Fall, and Andrew T. Sornborger

Abstract— **Optical imaging *in vivo* is an important tool for allowing researchers to understand neural ensemble interactions during awake behavior, sleep, anesthesia and during seizure activity. A major bottleneck in the overall efficiency of neural imaging experiments is the need for *post-hoc* analysis of imaging data. Computational capabilities are now at the point where real- or near-real-time multivariate analysis of imaging data is possible as data is acquired. In this paper we address the feasibility of performing real-time data analysis with a desktop computer, MATLAB, and a graphics processing unit (GPU). Important components of any real-time functional imaging analysis system are 1) dimensional reduction of the data, 2) visualization of the reduced vector space and 3) rapid calculation of functional connectivities. The ability to assess sources of variability in the data, and connectivity estimates on the fly, are potentially transformative for the way imaging laboratories perform their work. Here, we present benchmarks for analysis of functional imaging data using dimensional reduction methods and estimation of functional connectivities using least-squares and ridge regression methods.**

INTRODUCTION

For many years, electrophysiologists have plugged the output voltage from their electrodes into speakers and oscilloscopes to get immediate feedback from their measurement system [1]. This has allowed regions of neural tissue that respond to one or more stimuli to be rapidly identified. With this approach, the experimentalist can quickly optimize the experiment. Similar feedback does not currently exist in laboratories that use imaging methodology. Typically, data is acquired and saved for *post-hoc* analysis [2, 3]. With this approach, the experimentalist only finds out after a time-consuming experiment (and often a time-consuming analysis of a large quantity of low signal-to-noise ratio data) whether the stimulus conditions gave rise to a useable signal. The reason for this is the historical unavailability of inexpensive computational power capable of rapid analysis of large amounts of imaging data, often resulting in many hours of wasted time per experiment.

Manuscript received April 14, 2011. This work was supported in part by the National Institute of Mental Health, MH085973 (ACS, CPF and ATS), MH071847 (ACS), and the National Institute for Neural Disorders and Stroke, NS070159 (ATS).

A.C. Smith is with the Department of Anesthesiology and Pain Medicine, UC Davis, TB-170, One Shields Ave, Davis, CA 95616; and the Department of Bioengineering, University of Illinois at Chicago. (phone: 530-752-7813; fax: 530-752-7807; e-mail: annesmith@ucdavis.edu).

C.P. Fall is visiting the Department of Computer Science, Georgetown University, from the Department of Bioengineering, University of Illinois at Chicago. (e-mail: fall@uic.edu).

A.T. Sornborger is currently visiting UC Davis from the Department of Mathematics and Faculty of Engineering, University of Georgia, Athens, GA 30602. (e-mail: ats@math.uga.edu).

This paper studies some of the important components necessary for performing real- or near-real-time analysis of the multivariate data measured in such imaging experiments. The ability to calculate and visualize functional connectivities on the fly in neural data is increasingly important as our ability to collect data by various imaging and implanted-electrode methods improves and the complexity of experiments increases. Functional connectivity, which refers to the statistical relationship between neuronal observables computed from physiological data rather than anatomical measurements, allows researchers to identify spatio-temporal activation patterns and generate hypotheses concerning how regions interact during an experiment. A clear pre-requisite to computing functional connectivities in neural data is the ability to identify significant structure in noisy data.

Our target application is the development of a near-real-time multivariate analysis pipeline for neural imaging data. We acquire imaging data via a medical optical imaging CCD camera, which directly loads data into MATLAB using the Image Acquisition Toolbox [4]. The focus of this paper is therefore the maximization of computation speed of algorithms for computing functional connectivity for this data. Our specific application area is described in [3] for mapping the propagation of seizure-related neural bursting in the brain of the larval zebrafish transgenic for a ratiometric calcium indicator. However, there are many areas of medical imaging, medicine and neuroscience where real-time capabilities have considerable utility, for example, real-time PCA is currently being developed for spectral imaging video captured on surgical cameras [5]. In that case, PCA has been used to enhance and make possible the visualization of features normally unseen by the human eye. *In vivo* imaging studies in the rat have been used to elucidate UP and DOWN states observed in the neocortex during anesthesia and quiet wakefulness [6]. Estimation of network structure from functional magnetic imaging data and in vivo electrophysiological studies are also an active area of research [2, 7-9].

We report results of the application of linear models to the analysis of simulated imaging data and how, using a fast desktop computer and careful selection of dimensional reduction algorithms, we are able to identify structure in multivariate data in real- or near-real-time (depending on the time window being analyzed) using MATLAB [4] and GPU optimized Jacket DLA [10]

Section I below outlines our simulated dataset and estimation methods. We have chosen two examples to illustrate functional connectivity estimation in continuous data of different dimensions in near-real time. In the first example, we consider well-conditioned data with dimensions typically found in electrophysiology experiments (i.e. where there are more measurements than observables, Fig 1). In the second example, we consider the ill-conditioned case more typical of imaging experiments (i.e. where there are many more observables than measurements, Fig 1). Section II first outlines performance results on the two simulated data sets. These results clearly indicate that the SVD is the crucial algorithm for speed optimization. We therefore then present a more detailed analysis of the performance of SVD algorithms, including performance enhancements from the graphical processing unit (GPU). In Section III we discuss our results and make some concluding comments.

I. Methods

A. Data

Our goal here is to develop near-real-time methods to analyze optical imaging data. We assume the images are acquired at high frequency (for example, at a rate of 500-1000 images per second) with a reasonably large sampling density (for example, 128 x 128 = 16,384 pixels). For reasonable analysis speeds, we propose to analyze the data in windows of 100-1000 frames allowing the analysis to be completed and visualized before the next set of images has finished being acquired, that is a real-time analysis. For simplicity, we consider only monochromatic data. Imaging data matrices typically have many more observables than measurements (Fig. 1). Because such data sets have many more observables than measurements, they are ill-conditioned (i.e. their pixel-pixel covariance matrix is uninvertible) and require some sort of dimensional reduction. This imaging data will be compared with well-conditioned data (fewer observables than measurements and hence invertible pixel-pixel covariance matrix), as would be typical of electrophysiology, EEG or MEG experiments.

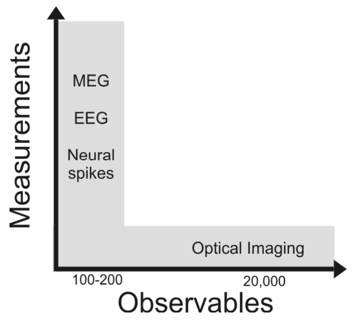

Fig. 1. Schematic of range of capability of near-real-time analysis using existing software. In an ideal situation we would be able to analyze large numbers of observables and many measurements. The gray-shaded area indicates where we are able to do most calculations using current technology.

B. Model and Estimation Algorithms

We represent a block of images changing in time by a matrix, X , of positive values with dimension the number of pixels (P) by the number of points in time (T). Matrix $X(t)$ is composed of snapshot image vectors $x(t)$ where $X(t) = \{x(t+1), x(t+2),..., x(t+T)\}$.

Here we are interested in computing a connectivity matrix, M , from the observations using the autoregressive (AR(1)) process:

$$x(t+1) = M\ x(t) + \rho + \varepsilon(t) \qquad (1)$$

where t (t = 1, ..., T) is time of a given frame, ρ is a constant (positive) drift term and $\varepsilon(t)$ is a zero-mean Gaussian noise term with unknown variance to be estimated. To identify the matrix M , we consider five candidate estimation algorithms. All of these approaches have been well-studied in the context of other application areas. Our focus here is on the speed of the algorithms as implemented within MATLAB.

- Algorithm 1. Linear regression applied to the full system. We apply a regression algorithm to the full system row by row using QR decomposition-based on a least squares method (regress.m in MATLAB; [11]).
- Algorithm 2. SVD-based reduced data algorithms. In this case, we reduce the data set using singular value decomposition, and estimate the connectivity from the reduced data [3]. First, matrix X is transformed into its singular value decomposition USV^T based on the selection of a set of eigenvectors with the largest eigenvalues. In our examples we took this number to be 20 and 50. We perform the SVD decomposition using two methods. The first, which we call method SVD1, uses QR decomposition (svds.m in Matlab, [11], this algorithm uses the standard ARPACK procedure [12]) and the second, method SVD2, relies on computation of the eigenvectors of the smaller of the two covariance matrices (pixel-pixel or time-time) and its eigenvalues. A reduced connectivity matrix is then computed using the same regression algorithm as in B1. Results are projected back to the observation space using the relation: $K \approx UK_{reduced}U^T$.
- Algorithm 3. Non-negative matrix factorization-(NNMF) based reduced data algorithm. We follow the same algorithm as in B2 but use the non-negative matrix factors [13, 14] of matrix $X = WH$ in the regression instead of the SVD. Results are transformed back to the observation space using the relation: $K \approx WK_{reduced}W^{-1}$.
- Algorithm 4. Ridge regression. This algorithm makes use of ridge regression (also known as Tikhonov regularization) which is an estimation method suited to ill-conditioned problems. We chose a small positive value (0.01) for the ridge parameter.

Computations were performed on a Microway Whisperstation desktop computer with an Intel Xeon E5620 Westmere 2.4 GHz Quad Core 32nm CPU with 12 Gb RAM and 64 Mb cache and an NVIDIA "Fermi" Tesla C2050 GPU with 3 GB GDDR5 memory with ECC support (144 GB/sec bandwidth) running 64-bit Linux and 64-bit

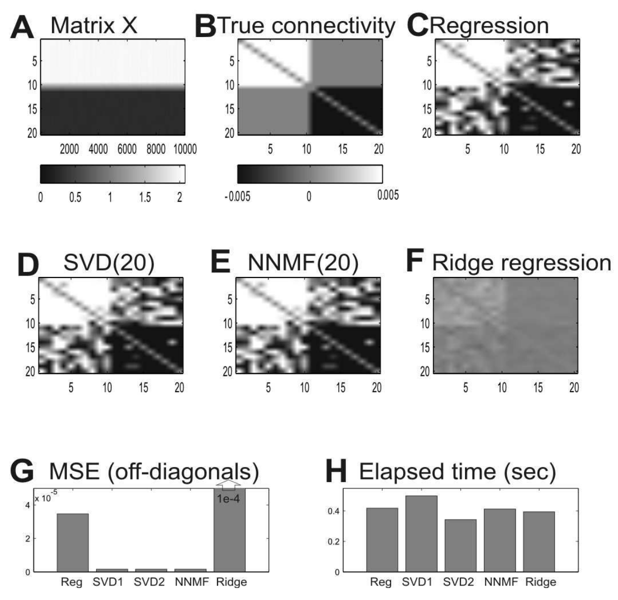

Fig. 2. Estimation of effective connectivity in a well-conditioned system. Panel A shows a 20 by 10,000 matrix of data simulated using an AR(1) process and the connectivity matrix (only off diagonal elements) shown in Panel B. Estimates of the connectivity computed by regression, SVD 1, NNMF and ridge regression methods are shown in Panels C-F. Mean squared error from the true values for the four methods (with two different SVD implementations) are shown in Panel G and corresponding computation time is indicated in Panel H.

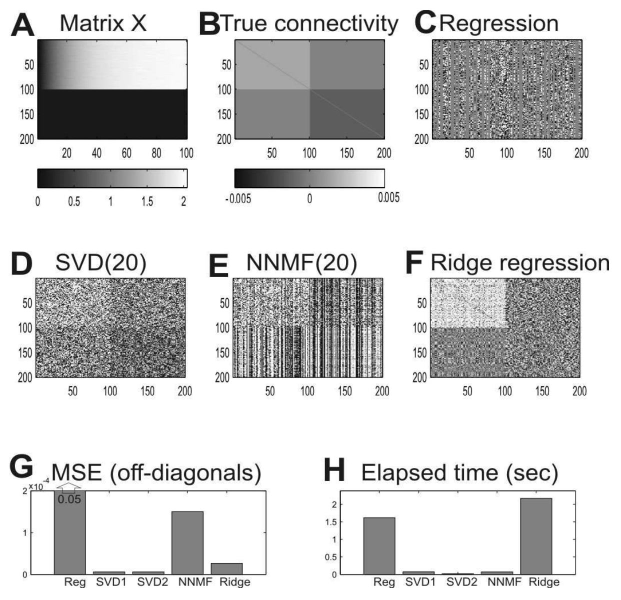

Fig. 3. Estimation of effective connectivity in an ill-conditioned system. Panel A shows a 200 by 100 matrix of data simulated using an AR(1) process and the connectivity matrix (only off diagonal elements) shown in Panel B. Estimates of the connectivity computed by regression, SVD 1, NNMF and ridge regression methods are shown in Panels C-F. Mean squared error from the true values for the four methods (with two different SVD implementations) are shown in Panel G and corresponding computation time is indicated in Panel H.

MATLAB 2010b with Jacket DLA (a linear algebra package that runs on a GPU by Accelereyes).

II. Results

A. Connectivity estimation

In Figs. 2 and 3 we illustrate performance of the five algorithms outlined above for two example data matrices: one where there are fewer observations than time points (Fig. 2) (well-conditioned data) and a second where there are more observations than time points (Fig. 3) (ill-conditioned data).

For the first example we consider an image matrix X of size 20 by 10,000 (Fig. 2A) generated by Eq. 1 using parameters as follows. Matrix M has values of 0.8 on the diagonal and off-diagonal elements composed of two non-zero blocks of positive (0.015) and negative elements (-0.015) and two blocks with zeros (see Fig. 2B). The constant drift term is $K = 0.1$ and zero-mean Gaussian noise is added at each time step with standard deviation 0.01. The initial image is all zeros i.e. $x(t) = 0$.

Estimates of connectivity computed using four of the algorithms (B1, B2A, B3 and B4) are shown in Fig. 2 (Panels C-F, respectively). (Results for B2B were similar in accuracy so we do not show the estimated connectivity matrix.) In Panel 2G and 2H we summarize the mean squared error (MSE) and elapsed computational time. For this example, the two SVD and NNMF methods are the most accurate. Computational times are comparable between all methods though the SVD2 method is the fastest.

In Fig. 3 we show corresponding results for a matrix X of size 200 by 100 generated using Eq. 1. Matrix M has the same diagonal elements as the previous example but off-diagonal values are reduced by a factor of 10. The same drift, noise and initial values were used as in the first example. While this second example has fewer matrix elements, it is quite slow to estimate using regression and ridge regression methods (Fig. 2H). These two methods also yield high mean squared errors (Fig. 2H). For this example the SVD2 method is fastest and most accurate.

Our results indicate that the SVD algorithm is most appropriate for our real-time application and therefore its speed will be crucial to our ability to implement real-time analysis. In the next section we consider only the SVD algorithms and look at timing results for a range of matrix dimensions.

B. Singular value decomposition: timing results

We performed SVD benchmark measurements on random normally-distributed matrices using four different SVD approaches (Fig. 4). These are SVD1 and SVD2 algorithms described in IB using the CPU and the GPU using Accelereyes [10].

Fig 4A shows the elapsed time estimate of the SVD using SVD1 (MATLAB's svds.m) on the CPU (solid lines), and using SVD1 (MATLAB's svds.m) on the GPU (dashed lines) for four choices of T (100, 400, 700 and 1000, marked above each curve). As matrix dimension (i.e. number of pixels) increases, the computation time increases almost linearly for all four values of T. Use of the GPU resulted in up to a 15-fold increase in computational speed. Fig. 4B shows elapsed times computed using the SVD2 algorithm applied to the same test set. In this case, the elapsed times are considerably lower for both CPU and GPU. Use of the GPU showed a small improvement effect in speed for T =1000. In the regime we are considering, (e.g. 15000 by 1000 matrices) use of SVD2 results in a 30-fold speedup on the CPU. Use of the GPU provided no more than a 20% additional improvement in speed.

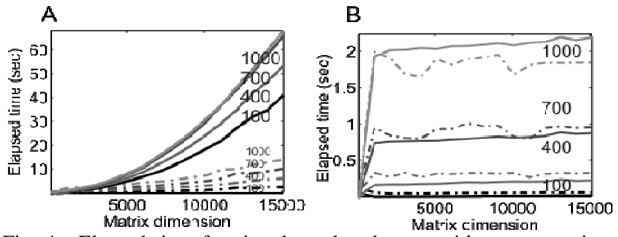

Fig. 4. Elapsed time for singular value decomposition computation on random matrices of varying dimensions. Each curve shows the elapsed time for computation of the highest 50 eigenvalues for non-square matrices with one dimension on the x-axis and the second dimension marked on the curve. We show results found using SVD1 (A) and SVD2 (B) and using the CPU (solid lines) and GPU (dashed lines). Note the very large difference in scale of the y-axes between the figures.

III. Conclusions

Our target was to develop real-time acquisition and analysis software that can perform fast image analysis on neurophysiological data. Using an optimized algorithm, we found that real-time analysis was possible on segments of 1000 frames of streaming imaging data with a 4-5 second delay, but no lost frames (i.e. all data was analyzed).

Our experience with SVD estimation was that considerable speedup can be achieved by optimizing the MATLAB code. For example, we achieved a 30-fold speedup for large matrix sizes by computing the SVD from the covariance matrix (SVD2) compared to the default MATLAB ([12]) algorithm (SVD1). This algorithm is known to be less stable than SVD1, but is clearly much faster. For the data sets that we analyzed, there was no significant difference in the eigenvectors or singular values that were calculated.

Using SVD2 for our particular real-time calculations, the GPU only provided a small speedup in computation time. However, for larger data matrices, other operations, and *post-hoc* analysis, GPU computing is able to contribute significant improvements. In future work, we plan to investigate additional speedup in the real-time domain using code specifically designed for the computer's GPU. Josth and colleagues [5] have demonstrated ten-fold speed-up of PCA calculations for snap-shot high resolution images using a parallel formulation and custom CUDA code on the GPU.

It is worth noting that it should be straightforward to apply these optimization methods to generalized linear models (GLMs), which are more appropriate for analysis of neural firing rate data [15]. Extension to other models should also be possible [16, 17]. The extra speed from general purpose GPU computing should aid in *post-hoc* estimation of confidence bounds on connectivity matrices using Monte Carlo or bootstrap methods which are inherently parallelizable.

Acknowledgments

ATS thanks the Department of Mathematics at UC Davis for providing a Visiting Research Associate position and office space during his year's leave from the University of Georgia. ACS thanks the Department of Statistics at UC Davis for providing a Visiting Researcher position and office space during 2011.

References

[1] D.H. Hubel and T.N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *J Physiol*, vol. 148, pp. 574-91, Oct 1959.

[2] J.N.D. Kerr and W. Denk, "Imaging in vivo: watching the brain in action," *Nature Reviews Neuroscience*, vol. 9, (no. 3), pp. 195-205, Mar 2008.

[3] L. Tao, L. J.D., and A.T. Sornborger, "Mapping functional connectivity between neuronal ensembles with larval zebrafish transgenic for a ratiometic calcium indicator," *Frontiers in Neural Circuits*, vol. 5, pp. 1-11, February 2011.

[4] *MATLAB 2010b*, The Mathworks, Natick, MA, 2010.

[5] R. Josth, J. Antikainen, J. Havel, A. Herout, P. Zemcik, and M. Hauta-Kasari, "Real-time PCA calculation for spectral imaging (using SIMD and GP-GPU)," *Journal of Real-Time Image Processing*, vol. Online First™, 22 January 2011, 2011.

[6] C.C.H. Petersen, T.T.G. Hahn, M. Mehta, A. Grinvald, and B. Sakmann, "Interaction of sensory responses with spontaneous depolarization in layer 2/3 barrel cortex," *P Natl Acad Sci USA*, vol. 100, (no. 23), pp. 13638-13643, Nov 2003.

[7] K. Friston, J. Phillips, D. Chawla, and C. Buchel, "Revealing interactions among brain systems with nonlinear PCA," *Human Brain Mapping*, vol. 8, (no. 2-3), pp. 92-97, 1999.

[8] M. Okatan, M.A. Wilson, and E.N. Brown, "Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity," *Neural Comput*, vol. 17, (no. 9), pp. 1927-1961, Sep 2005.

[9] E.S. Chornoboy, L.P. Schramm, and A.F. Karr, "Maximum-Likelihood Identification of Neural Point Process Systems," *Biological Cybernetics*, vol. 59, (no. 4-5), pp. 265-275, 1988.

[10] AccelerEyes, Atlanta, GA, 2011.

[11] N.R. Draper and H. Smith, *Applied regression analysis*, New York: Wiley, 1998.

[12] R.B. Lehoucq, D.C. Sorensen, and C. Yang, *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, Philadelphia, PA: SIAM Publications, 1998.

[13] M.W. Berry, M. Browne, A.N. Langville, V.P. Pauca, and R.J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," *Computational Statistics & Data Analysis*, vol. 52, (no. 1), pp. 155-173, Sep 15 2007.

[14] D.D. Lee and H.S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, (no. 6755), pp. 788-791, Oct 21 1999.

[15] P. McCullagh and J.A. Nelder, *Generalized linear models*, Boca Raton: Chapman & Hall/CRC, 1998.

[16] S. Roweis and Z. Ghahramani, "A unifying review of linear gaussian models," *Neural Computation*, vol. 11, (no. 2), pp. 305-345, Feb 15 1999.

[17] K.L. Myers, A.E. Brockwell, and W.F. Eddy, "State-space models for optical imaging," *Statistics in Medicine*, vol. 26, pp. 3862-3874, 2007