31st Annual International Conference of the IEEE EMBS
Minneapolis, Minnesota, USA, September 2-6, 2009

A Graph-Laplacian-Based Feature Extraction Algorithm for Neural Spike Sorting

Yasser Ghanbari, Larry Spence, and Panos Papamichalis, *Fellow, IEEE*

***Abstract*—Analysis of extracellular neural spike recordings is highly dependent upon the accuracy of neural waveform classification, commonly referred to as spike sorting. Feature extraction is an important stage of this process because it can limit the quality of clustering which is performed in the feature space. This paper proposes a new feature extraction method (which we call Graph Laplacian Features, GLF) based on minimizing the graph Laplacian and maximizing the weighted variance. The algorithm is compared with Principal Components Analysis (PCA, the most commonly-used feature extraction method) using simulated neural data. The results show that the proposed algorithm produces more compact and well-separated clusters compared to PCA. As an added benefit, tentative cluster centers are output which can be used to initialize a subsequent clustering stage.**

I. Introduction

THE acquisition and analysis of neural action potentials (spikes) is of fundamental importance in many areas of neuroscience. Results from such analyses can be used in applications such as neural prosthetics, brain-machine interfaces, pharmacology, etc.

In such applications, the neural signal tends to be noisy, as it records not only the spikes from neurons near the recording electrode, but also the activities of more distant neurons. Hence, it is necessary to extract the spikes from that noisy signal and then classify them in groups (clusters), each cluster corresponding to one nearby neuron. This process is known as spike sorting.

In spike sorting, the spikes are detected in the digitized signal, the extracted spike waveforms are projected into a lower-dimensional feature space, the number of neurons generating those spikes is determined, and each spike event is assigned to a neuron.

At the spike detection stage, spike events are detected using one of a number of possible techniques, the most popular among them being amplitude thresholding [1]. The spike detection algorithms usually assume a maximum duration (length) for spikes. This length may vary typically from 0.5 to 4 milliseconds depending on the nature of the neurons being recorded. All the spikes are then represented as digitized waveform segments. Therefore, each detected digitized spike constructs a vector of length $n=0.5\times10^{-3}\times f_s$ to $4\times10^{-3}\times f_s$ where f_s is the sampling frequency. Now, each of the n-length spike vectors represents a point in an n-dimensional space of raw sample values. Since clustering in the n-dimensional space is computationally expensive and difficult to visualize, and because we conjecture that the spike waveforms may be accurately classified in a feature space of low dimensionality, a feature extraction stage is employed to reduce the dimensionality of each spike vector, n, to a d-length feature vector ($d<<n$ where, usually, $d = 2$ or 3) so that the clustering is performed in the d-dimensional space. This dimensionality reduction can be expressed by the linear projection:

$$y_{d\times1} = A^T_{n\times d} x_{n\times1}, \quad (1)$$

where the elements of y correspond to the dimensions of the feature space that will be used for clustering. The most popular algorithms proposed for dimensionality reduction include Principal Components Analysis (PCA) [1], wavelet-based techniques [2,6], and others.

The next stage involves determining the number of firing neurons and assigning each spike (as represented by its projection in feature space) to its originating neuron. Typical clustering algorithms used for spike classification include K-means [3], Bayesian clustering [1], and others.

This paper introduces a new feature extraction algorithm for neural spike sorting and describes the results of applying it to simulated neural data. The generated clusters are visually compared with PCA clusters and a cluster-validity metric is used to assess the performance of these two techniques over several test cases. The improvement is shown to be considerable on the test data.

II. Proposed Algorithm

Principal components analysis (PCA) is the most popular technique among spike feature extraction methods. However, the fact that the PCA assumes the best separation to happen along the direction of the largest variance may not be true in many cases. Therefore there is a need for an algorithm which considers some other criteria to generate more accurate results. This leads us to define a new cost function instead of merely maximizing the variance as in PCA.

A. Graph Laplacian

Suppose that there are N spikes, each of length n, detected

Manuscript received April 7, 2009. This work was supported by Plexon Inc. Patent Pending by Plexon Inc.

Y. Ghanbari is with the Department of Electrical Engineering, Southern Methodist University, Dallas, TX 75275 USA (phone: 1-214-768-3783; fax: 214-768-3573; e-mail: yghanbari@smu.edu).

L. Spence is with Plexon Inc, Dallas, TX 75206 USA, (e-mail: larry@plexoninc.com).

P. Papamichalis is with the Department of Electrical Engineering, Southern Methodist University, Dallas, TX 75275 USA (e-mail: panos@lyle.smu.edu).

at the first stage of the sorting process. Each n-length spike represents a data point x_i in an n-dimensional space, $1 \le i \le N$. It is desirable that the points that are close to each other in the original n-dimensional space remain close to each other after projection to the d-dimensional space. This will help insure that the resulting clusters in the d-dimensional feature space are compact and well separated.

Based on concepts from spectral graph theory [4], we note that the n-dimensional points (spikes) induce a graph G with edges connecting spikes that are near each other in the original space. To construct an appropriate weighted graph G with N nodes, two steps are required [5]:

Step 1. We put an edge between nodes i and j if x_i and x_j are close, where closeness is measured by a distance metric $dist(x_i,x_j)$. In this work, we use the K-nearest-neighbors approach [5] where nodes i and j are connected by an edge if i is among the K nearest neighbors of the node j, or j is among the K nearest neighbors of the node i. K is a parameter we choose for the algorithm.

Step 2. To choose the weights for the edges, we use the heat-kernel method [5]. In this approach, if nodes i and j are connected, the edge weight W_{ij} is defined as:

$$W_{ij} = \exp\left(-\frac{\left[dist(x_i, x_j)\right]^2}{t_{ij}} \right), \tag{2}$$

where x_i and x_j are points in the n-dimensional space, $dist(x_i,x_j)$ represents a distance metric between x_i and x_j, and W_{ij} is the weight of the edge of the graph between the points x_i and x_j. This weight formula is used for at least the K edges with the smallest distance. For the other edges $W_{ij}=0$. The parameter t_{ij} is a real number that will be discussed later. The W_{ij} can be considered as elements of a similarity matrix W.

Let A be the transformation matrix for projecting each n-dimensional data point x_i to a d-dimensional feature point y_i as shown in Equation (1). An appropriate transformation matrix A can be obtained by solving the following minimization problem:

$$\min_A C_1(A) = \min_A \sum_{i,j} \left\| y_i - y_j \right\|^2 W_{ij}, \tag{3}$$

where $y_i=A^T x_i$ and $C_1(A)$ is the cost function to be minimized. This cost function will be minimized if the nearby points x_i and x_j are projected into nearby points y_i and y_j.

Now, we define a diagonal matrix D as the sum of the columns (or rows) of W:

$$D = [D_{ij}]; \quad D_{ij} = \begin{cases} \sum_k W_{ik}, & if\ i = j; \\ 0, & otherwise. \end{cases} \tag{4}$$

The cost function $C_1(A)$ can be rewritten as:

$$\begin{aligned} C_1(A) &= \sum_{i,j} \left\| y_i - y_j \right\|^2 W_{ij} \\ &= \sum_{i,j} tr\left((y_i - y_j)(y_i - y_j)^T \right) W_{ij} \\ &= 2tr\left\{ \sum_i \left(y_i y_i^T \sum_j W_{ij} \right) - \sum_{i,j} \left(y_i y_j^T W_{ij} \right) \right\} \\ &= 2tr\left\{ \sum_i \left(y_i y_i^T D_{ii} \right) - \sum_{i,j} \left(y_i y_j^T W_{ij} \right) \right\}. \end{aligned} \tag{5}$$

Substituting Equation (1) into Equation (5) yields:

$$\begin{aligned} C_1(A) &= 2tr\left\{ A^T \left[\sum_i \left(x_i x_i^T D_{ii} \right) - \sum_{i,j} \left(x_i x_j^T W_{ij} \right) \right] A \right\} \\ &= 2tr\left\{ A^T X (D - W) X^T A \right\} \\ &= 2tr\left\{ A^T X L X^T A \right\}, \end{aligned} \tag{6}$$

where $X = [x_1, x_2, \cdots, x_N]_{n \times N}$. The term $L=D-W$ is called the Laplacian matrix [4,5] of the graph G. L is a symmetric positive semidefinite matrix.

B. Weighted Variance

In addition to minimizing the cost function $C_1(A)$, we wish to maximize the variance of the d-dimensional feature points (similar to PCA) in order to maximize cluster separation. Based on Equation (4) D_{ii} is the sum of the similarities of x_i to its K nearest neighbors in the n-dimensional space. Therefore, assuming clusters whose density is highest near their centers, the greater the D_{ii} value of a point x_i, the more likely it is to be near the center of the cluster containing it. This leads us to take advantage of the fact that the points x_i which are near their cluster centers should contribute more in variance calculation compared to the x_is which are outliers or noise.

We note that those points x_i with the highest D_{ii} values within each connected nearest-neighbor subgraph can serve as tentative cluster centers, as well as indicating the number of clusters, for a subsequent clustering stage. The quality of the results of many clustering algorithms, such as K-means and mixture-of-Gaussians, is improved greatly by specifying the number of clusters and the cluster centers in advance.

Assuming that the feature points have a zero mean, the weighted variance can be defined based on spectral graph theory [4] as follows:

$$\begin{aligned} \mathrm{var}(Y) &= \sum_i \left\| y_i \right\|^2 D_{ii} = \sum_i tr\left\{ y_i y_i^T \right\} D_{ii} \\ &= tr\left\{ A^T \left(\sum_i x_i x_i^T D_{ii} \right) A \right\} \\ &= tr\left\{ A^T X D X^T A \right\}. \end{aligned} \tag{7}$$

Therefore, the second objective function can be defined as:

$$\max_A C_2(A) = \max_A tr\left\{A^T\left(XDX^T\right)A\right\} \quad (8)$$

C. Optimization Problem and Solution

Since the weighted distance of nearby feature points are to be minimized while the weighted variance is to be maximized, we can define the optimization problem as the minimization of the ratio of $C_1(A)$ to $C_2(A)$ as follows:

$$\min_A \frac{C_1(A)}{2C_2(A)} = \min_A \frac{tr\left\{A^T\left(XLX^T\right)A\right\}}{tr\left\{A^T\left(XDX^T\right)A\right\}}. \quad (9)$$

The above cost function can be minimized using the eigenvalue-eigenvector decomposition of:

$$\left(XLX^T\right)\underline{a} = \lambda\left(XDX^T\right)\underline{a}, \quad (10)$$

where λ is the eigenvalue corresponding to the eigenvector $\underline{a}$. The eigenvector which corresponds to the smallest nonzero eigenvalue in Equation (10) will minimize the cost function. This equation can be simplified to:

$$\left(XWX^T\right)\underline{a} = (1-\lambda)\left(XDX^T\right)\underline{a}. \quad (11)$$

Since the minimum λ corresponds to maximum $\lambda'=1-\lambda$, the problem can be reformulated to solve for the largest eigenvalue of the following generalized eigen-problem:

$$\left(XWX^T\right)\underline{a} = \lambda'\left(XDX^T\right)\underline{a}. \quad (12)$$

The solution to Equation (12) includes n eigenvalues $[\lambda'_1,\lambda'_2,\ldots,\lambda'_n]$ where $1\geq\lambda'_1\geq\lambda'_2\geq\ldots\geq\lambda'_n$. The transform matrix A is then constructed as the eigenvectors corresponding to the d largest $\lambda'\neq1$ (or smallest nonzero λ).

III. Experiments and Results

In order to investigate the performance of the proposed algorithm for feature extraction of neural spikes and compare it with the PCA technique, we used simulated data. The simulated data is labeled with the originating neuron for each spike. This makes visual evaluation of the resulting clusters easier and allows us to compute a numerical cluster validity metric for each dataset and feature extraction method.

The simulated data provided with the Wave_clus software [8] were used in this work for test and comparison purposes. There are three advantages in using this test data: First, it was independently generated, second its characteristics closely resemble those of actual in vivo neural signals as mentioned by its developer in [6], and third it can be used by other investigators as a basis for comparison between various algorithms. The simulated dataset here includes four sets (with three templates in each) and each set was used to generate simulated continuous noisy neural signals with different levels of noise (different SNRs.) The number of spikes in each simulated noisy signal varies between approximately 3200 and 3500. These sets are provided in MATLAB .mat files named C_Easy1_noise*.mat, C_Easy2_noise*.mat, C_Difficult1_noise*.mat, and C_Difficult2_noise*.mat, where the asterisk is a place-holder for a number indicating the noise level.

In this work, each noisy test signal provided by Wave_clus is loaded and a standard amplitude thresholding algorithm is employed for spike detection. All the detected spikes are waveform segments of 40 samples in length (n=40).

The detected spikes are then aligned based on the location of the maximum absolute amplitude peak and the aligned spikes are used for feature extraction. Both PCA and the proposed method are used for reducing the dimensionality from n=40 to d=3 and the best 2-D view for each of the methods is provided for visual comparison.

The PCA technique applies the eigenvalue-eigenvector decomposition to the covariance matrix $\Sigma=XX^T$ and the three 40-length eigenvectors corresponding to the three largest eigenvalues are selected to construct $A_{40\times3}$.

The algorithm described in section (II) requires appropriate values for certain parameters. In constructing the graph G, we set K =5 to form edges between each node i of the graph and its five nearest neighbors. Our experiments show that the algorithm is robust to this parameter when values greater than five are used. For the similarity matrix W, the distance metric used in this work is the Euclidean distance. The parameter t_{ij} is proposed in [7] to be calculated as follows:

$$t_{ij} = \sigma_i\sigma_j; \quad \sigma_i = dist\left(x_i, x_K\right), \quad (13)$$

where x_K is the K^{th} nearest neighbor to x_i and here K=5 as explained before.

The dataset "C_Difficult1_noise*.mat" which includes four simulated noisy signals with different levels of noise is used here for visual assessment of feature extraction. The amplitude thresholding algorithm detects more than 3300, 3360, and 3020 spikes for the noise variances of 0.05, 0.10, and 0.15, respectively. PCA and the proposed algorithm are applied to the peak-aligned detected spikes to reduce the dimensionality of all the spikes from 40 to 3. For brevity, we call the graph Laplacian features as "GLF". Both the PCAs and the GLFs are plotted in Figure 1 for the noise levels of 0.05, 0.10, and 0.15. The fourth simulated signal in this set, with a noise level of 0.20, was also tested and did not show any difference between PCA and GLF, and is therefore not shown here (all the clusters merge together in both feature spaces.) The 2-D plots of the best two features (among three) which represent the best cluster separation in both PCA and GLF are selected to be shown in Figure 1. Here, among these three test cases the best features shown in all the PCA cases is the first and the second features while for GLF they are the second and the third ones.

A quantitative metric that can be used to assess the clusters generated by applying the two methods to the labeled data is the cluster validity measurement. We use the ratio of inter-cluster distance to intra-cluster distance suggested in [6] as the quantitative metric of cluster validity. This ratio is defined as:

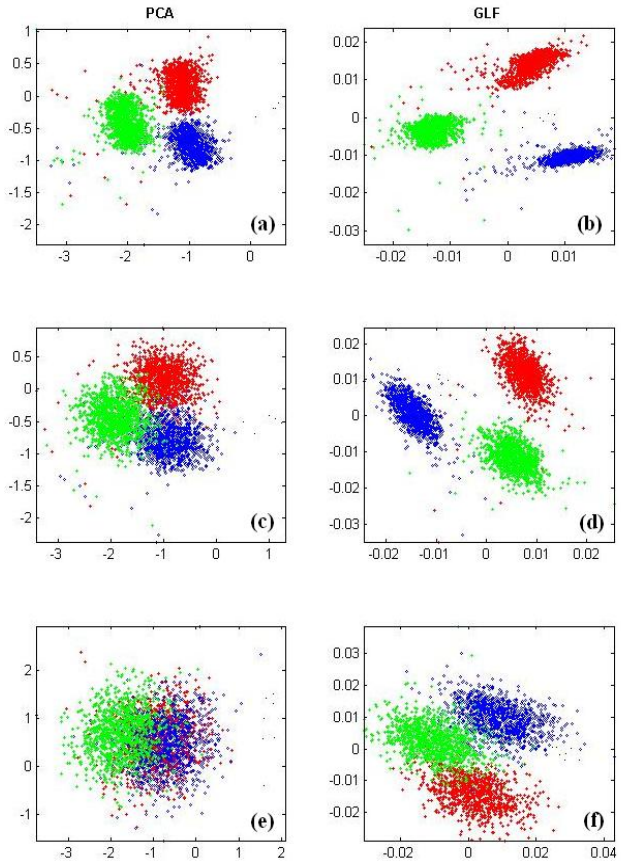

Figure 1. The PCA and GLF obtained from the simulated data "C_Difficult1_noise*.mat" with different levels of noise: (a) and (b) 0.05; (c) and (d) 0.10; (e) and (f) 0.15.

$$Validity = \frac{\text{inter}}{\text{intra}} = \frac{\min\limits_{\substack{i=1,2,..,C-1 \\ j=i+1,...,C}} \left(\left\| z_i - z_j \right\|^2 \right)}{\frac{1}{N} \sum_{i=1}^{C} \sum_{y \in C_i} \left\| y - z_i \right\|^2}, \qquad (14)$$

where z_i is the centroid of the cluster C_i, C is the number of clusters (here C=3), y vectors are the cluster members (here 3-D feature vectors), and N is the number of points, which is equal to the number of detected spikes. The first goal in this expression is to maximize the inter-cluster distance which is defined as the minimum Euclidean distance of the cluster centers. The second goal is to minimize the intra-cluster distance, defined by the summation of the Euclidean distances of the members of each from each other. Therefore, the greater the *Validity* value is, the more distinct the clusters are.

This quantitative metric is applied to all four datasets for noise levels of 0.05, 0.10, 0.15, and 0.20 and the cluster validity metric is calculated for the 3-D clusters generated by both PCA and GLF. The results are plotted in Figure 2. As can be seen from these plots, the GLF method gives better clustering results than PCA.

IV. Conclusion

The proposed method of feature extraction for spike sorting was shown to outperform PCA over the simulated test cases. The locality-preserving property of the cost function tends to produce compact clusters, while the calculation of variance weighted by local density (and therefore proximity to a cluster center) is robust to outliers and produces well-separated clusters. The mutual satisfaction of these two goals leads to a new optimization problem whose solution results in a feature extraction method that yields improved cluster quality compared to PCA. As a by-product, the method outputs the number and location of tentative cluster centers, which can be used to initialize a clustering algorithm.

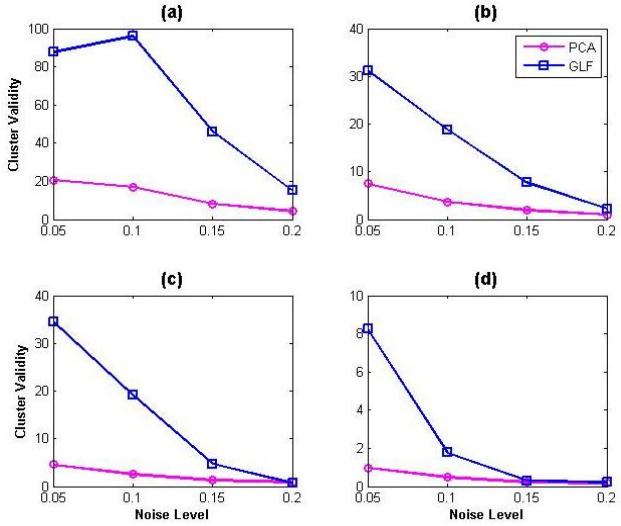

Figure 2. The cluster validity measurements for 3-D PCA and GLF obtained from four simulated datasets with different levels of noise: (a) "C_Easy1_noise*.mat"; (b) "C_Easy2_noise*.mat"; (c) "C_Difficult1_noise*.mat"; (d) "C_Difficult2_noise*.mat".

References

[1] M. S. Lewicki, "A review of methods for spike sorting: the detection and classification of neural action potentials," *Network: Computation Neural Syst.*, vol. 9, pp. R53-R78, 1998.

[2] A. Pavlov, V. A. Makarov, I. Makarova, and F. Panetsos, "Sorting of neural spikes: when wavelet based methods outperform principal component analysis," *Neural computing*, vol. 6, pp. 269-281, 2007.

[3] S. Ray and R. H. Turi, "Determination of number of clusters in K-means clustering and application in colour image segmentation," in *Proc. of the 4th Int. Conf. on Advances in Pattern Recognition and Digital Techniques (ICAPRDT 1999)*, Calcutta, India, December 1999, pp. 137-143.

[4] F. R. K. Chung, *Spectral Graph Theory*. Regional Conference Series in Mathematics, number 92, 1997.

[5] M. Belkin and P. Niyogi, "Laplacian eigenmap and spectral techniques for embedding and clustering," *Advances in Neural Information Processing Systems 14 (NIPS 2001)*, vol. 1, 2002, pp. 585–591, MIT Press, Cambridge.

[6] R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul, "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering," *Neural Computation*, vol. 16, no. 8, pp. 1661–1687, 2004.

[7] L. Z. Manor and P. Perona, "Self-tuning spectral clustering," in *Proc. of Advances in Neural Information Processing Systems (NIPS 2004)*, 2005, pp. 1601–1608.

[8] Available: http://www2.le.ac.uk/departments/engineering/extranet/research-groups/neuroengineering-lab/spike-sorting