Improving spike-sorting techniques in multielectrode recordings from the primate retina

Timothy A Machado[1]
Undergraduate Thesis

Advisors: E.J. Chichilnisky, Virginia de Sa

[1] Systems Neuroscience Laboratory (SNL-E),
The Salk Institute for Biological Studies, La Jolla, CA 92037
tamachado@salk.edu

1. Introduction

Understanding how populations of cells in the brain encode information is a fundamental question in modern neuroscience. Researchers have attempted to understand this problem in a number of unique ways ranging from computational models using unsupervised learning algorithms (Hinton and Sejnowski, 1999) to probabilistic interpretations of the neural code in electrophysiological recordings (Shlens et al., 2006). Although the field of cognitive science is rather broad, a central goal of much of the research is to unravel the nature of computation within the human mind (Clark, 2001). While the brain does not encode information in a manner similar to a digital computer, thought and neural computation seem to be emergent properties of the spiking of many discrete neurons (Rieke, 1997). In fact, some even go as far as to argue that a select population of neurons actually encodes for the conscious perception of an individual at a given moment (Koch, 2004). If it is accepted that the properties of spike trains do indeed code for "thought," models can theoretically be constructed that emulate various properties of neural systems.

Indeed, in recent years many useful findings have arisen from algorithms used to solve problems similar to ones solved by the brain. One particularly strong example of this can be seen in the development of Independent Component Analysis (ICA) and its application to solving the "cocktail party problem" (Brown et al., 2001). Imagine a party in which there are an arbitrarily large number of groups engaged in independent discussions. From the perspective of a single microphone positioned on the ceiling, the sum of all the conversations in the room would likely seem like an indecipherable mess of noise. Yet, an individual present in the room will immediately attend to someone in the crowd who says his name. ICA can solve this problem as well, as long as there are a sufficient number of detectors (in this example, microphones) such that each conversation can be isolated (Brown et al., 2001). This is rather remarkable, especially considering that ICA receives no feedback about its performance after completing its analysis (consequently, it makes sense that ICA can be implemented using an unsupervised learning algorithm). Thus, one could argue that ICA solves the "cocktail party problem" in a similar way to the brain, especially since there seem to be more neurons in auditory cortex devoted to spatial localization of sound than are necessary to do so. This supports the idea that auditory cortex is capable of localizing multiple sources simultaneously (von der Malsburg and Schneider, 1986; Brown et al., 2001).

As can be seen from the above example, computational models can be used to make predictions about neural systems and guide further research. Obviously in order to be validated, computational models must be supported by a continually growing body of evidence. Since so much is currently unknown, it seems unlikely that any contemporary model will offer a comprehensive view of any neural system. In light of this, why do so

many researchers focus on developing biologically implausible learning algorithms? In order to answer that question, let us briefly discuss various approaches to the problem of understanding the neural code.

In the past, a popular approach for examining the neural code involved *in vivo* single unit electrophysiological recordings from regions of cortex (Hubel and Wiesel, 1968). While this approach certainly generated many useful findings, it only allowed researchers to examine individual cells at a time. This made it rather difficult to form models and generalizations about how entire populations of cells encode information. Consequently, in recent years it has become much more common to construct experimental apparatuses for multielectrode extracellular recording. Much of this work is done using clusters of four electrodes (tetrodes) in order to study areas of cortex. Unfortunately, when recording from multiple cells it is a nontrivial task to separate the raw electrical activity present on a single electrode into a series of spike trains corresponding to individual neurons (Sahani, 1999).

Fortunately, this problem is solvable as it seems to be a manifestation of the "cocktail party problem" discussed above. As a result, many unsupervised learning techniques[1] developed to model neural activity directly can be applied in a different way—to isolate individual spike trains from raw electrical recordings containing many cells firing simultaneously. This analytical process is known as "spike sorting."

In tetrode recordings from the brain, spike sorting is especially difficult. This is in part due to the fact that the signal to noise ratio[2] (SNR) is generally very low in such

[1] Certainly not all unsupervised learning algorithms were created by the neuroscience and artificial intelligence communities. Many other fields of science and engineering are interested in finding solutions to similar "blind source separation" (BSS) problems.
[2] SNR is defined as the variance of the signal divided by the variance of the noise.

preparations. Indeed, one group evaluated the efficacy of their spike sorting techniques by comparing extracellular measurements to intracellular measurements acquired simultaneously. They reported that manual spike sorting techniques resulted in errors (types I and II) ranging from 0 to 30% while the implementation of more advanced techniques reduced the error range from 0 to 8% (Harris et al., 2000). While the methods studied in this paper can be applied to cortical tetrode recordings, the spike sorting system being discussed is implemented on a planar microelectrode array used to record from retinal tissue[3].

The process of spike sorting on data obtained from planar electrode arrays offers its own set of unique challenges. The primary one is that of size. Most planar arrays used to record from patches of retinal tissue are significantly smaller—about twenty electrodes (Segev et al., 2004). Our arrays are on the order of twenty five times larger (512 electrodes). Consequently, the spike sorting techniques must be automated as much as possible, as any manual component would consume an unreasonable amount of human time. This also means that the techniques used must be very robust as even a manual validation process is prohibitively costly in terms of time.

Our recordings are also unique in another significant manner—as a direct consequence of their large scale: receptive fields and firing patterns of large populations of morphologically distinct cell classes can be examined at a previously unseen scale (Chichilnisky and Kalmar, 2002; Shlens et al., 2006). The retina is uniquely suited for analyses of this type as the visual input into the retina can be explicitly controlled by our apparatus. The properties of the cells firing in response to these input stimuli can then be

[3] The arrays used in the lab were developed by AM Litke, N Bezayiff, EJ Chichilnisky, and others (Litke et al., 2004). The most commonly used arrays contain 512 electrodes that cover a total area of 1.2 mm^2. The electrode spacing on these arrays is 60μm.

analyzed with the knowledge that in the greater context of the visual system there is virtually no feedback to the retina[4] (see Reperant et al., 2006). Thus, the retina is an ideal model system for understanding the properties of large populations of neurons.

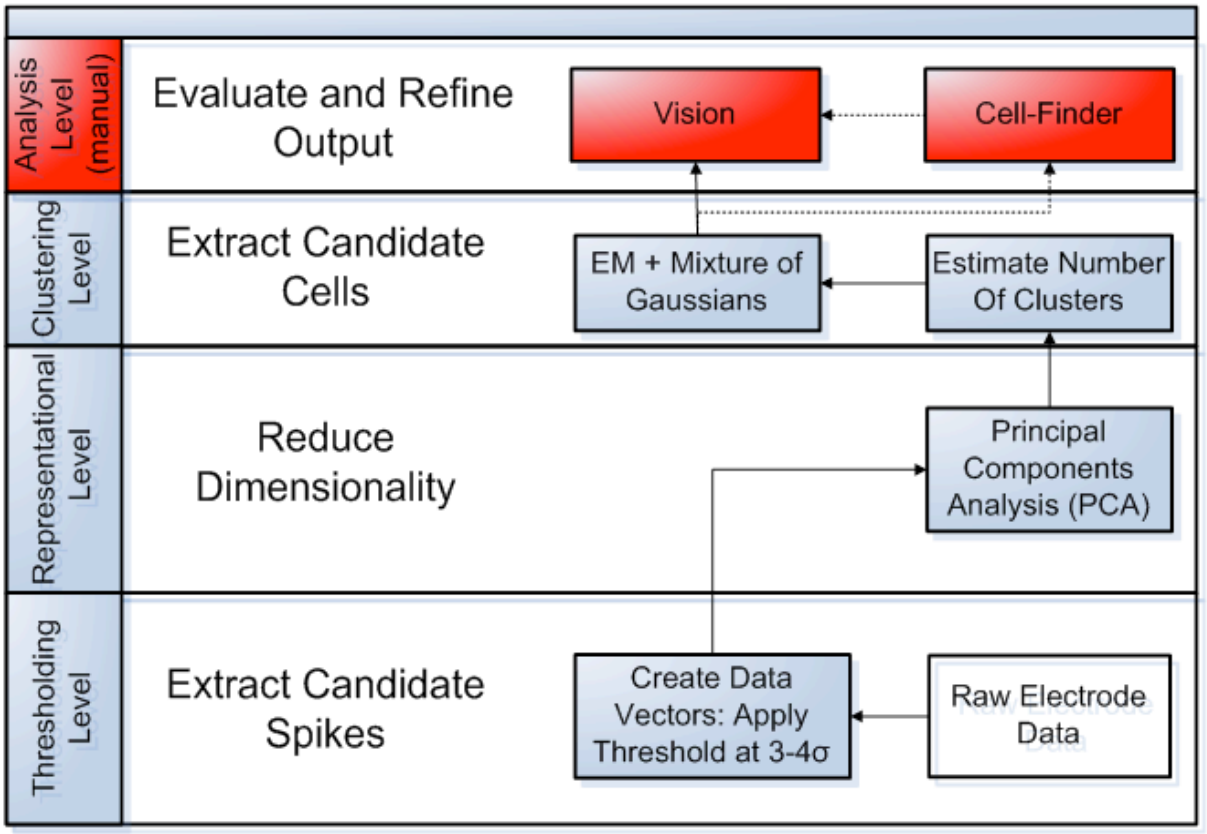

Figure 1 – Overview of current spike sorting system. Automated thresholding is performed at a predefined level, PCA is used to reduce the dimensionality of the data, and then clustering algorithms are used to isolate candidate cells. The final level, evaluation of output, involves the use of two custom Java software packages.

[4] While it is known that there is feedback to the retina (in the form of "centrifugal fibers"; see Reperant et al., 2006), there are very few of such fibers present in primate retina (on the order of 10 fibers per eye).

2. Further Background and Aims

The current spike sorting system (the "standard analysis procedure") involves three main steps: thresholding and feature extraction, dimensional reduction[5] using Principal Component Analysis (PCA), and finally clustering the data using a mixture of Gaussians model (see Figure 1 for an overview of the system; the beginning of Section 4 discusses the system in more explicit detail). It is very successful compared to many other systems: it can locate hundreds of cells from a single preparation in less than half an hour—without any user intervention. However, it performs far from optimally in some cases. Preliminary work indicates that the system may fail to adequately isolate up to 50% of the cells present in a preparation, in some cases (J. Gauthier, unpublished data). Isolating many additional cells will likely provide information about retinal ganglion cell morphological classes that are currently not well understood. Additionally, finding more cells from well-studied morphological classes will increase the amount of data available to draw inferences about the encoding properties of the retina as a whole. Consequently, this project intends to investigate a variety of different techniques that could be used in order to improve the performance of the spike sorting system.

A number of algorithms seem especially promising for successful implementation, across all levels of the spike sorting system. However, deciding which

[5] Throughout the paper the terms "dimensional reduction" and "representation" will be used interchangeably. This is somewhat misleading because techniques used for dimensional reduction, such as Principal Components Analysis, can more broadly be described as "representational algorithms"—techniques that find a way to represent data in a new space (that may or may not be of a lower dimensionality). Thus, dimensional reduction techniques are the subset of all possible representational algorithms capable of being used for dimensional reduction. However, each of the representational algorithms to be discussed can also be used for other tasks besides dimensional reduction. Consequently, labeling these techniques "dimensional reduction algorithms" is not strictly correct. Nevertheless, since representational algorithms will be used exclusively for dimensional reduction in this document, the two terms will be used interchangeably in the context of this paper.

level to focus on improving first is not obvious. For instance, using a new algorithm for dimensional reduction (such as Independent Component Analysis) could improve the overall yield of cells from the system. But is a modest increase in cell yield worthwhile if the users of the system still have to spend hours of their time verifying and improving the results from the clustering algorithm? In planning for this project, I decided that focusing on the representation of the data was justifiable for a number of reasons. First, the actual upper bound performance of an ideal spike sorting system on our data is unknown. By running the current system 24 times (varying the parameter set on each iteration) and then combining the results, it is possible to approximate the number of cells lost due to the parametric nature of the current system. This analysis, referred to as the "union analysis," is effective at isolating many more new cells by varying parameters in the thresholding and clustering algorithms (J. Gauthier, unpublished data). However, deficits in the representation of the data due to nature of PCA cannot be revealed through this process. Consequently, it is not clear how much a new representational algorithm can improve the yield of isolated cells. Over the course of the project I evaluated the performance of many new representational techniques and compared the results to PCA (with each technique integrated into the current spike sorting system) as well as the output from the union analysis—with the assumption that the union analysis offers a lower bound approximation for the output from an improved spike sorting system.

Aim 1: Compared the efficacy of the current representational algorithm (PCA) to a number of different techniques. If a new technique demonstrated promise, it was integrated with the other components of the standard analysis procedure. Then, it

was rigorously compared to both a single iteration of the standard analysis procedure as well as to the union analysis.

Another reason for concentrating on the representation of the data rather than clustering is the fact that a great deal of time has already been invested in developing a software tool to quickly evaluate and improve the output from the clustering algorithm. Over the past year, many features were be added to this software package (known as Cell-Finder). The hope is that these new features have already decreased the amount of time that users of the spike sorting system must spend in validating the results from clustering. Consequently, now focusing on improving the yield of the system is further justified.

Aim 2: The current semi-automatic spike sorting software ("Cell-Finder") used widely by the laboratory was augmented with a variety of new time saving features.

3. Methods

A more in depth discussion of the methods used in the laboratory can be seen elsewhere (Chichilnisky and Kalmar, 2002; Litke et al., 2004). This section is intended only to provide sufficient background for the methods that will be referenced later in this document. Additionally, these various algorithms and methods tested will be introduced and discussed later in this document.

3.1 Experimental Apparatus

The most commonly used arrays contain 512 electrodes that cover a total area of 1.2 mm^2. The electrode spacing on these arrays is 60μm and the electrodes are arranged in a rectangular pattern. However, arrays with other sizes and electrode spacings exist as well—such as the larger 519 electrode array that has hexagonal geometry. Patches of living retinal tissue (with the pigment epithelium typically removed[6]) are placed onto the array such that the retinal ganglion cell (RGC) layer directly contacts the electrode array. The tissue is continually perfused with physiological solution to maintain its health for up to 15 hours.

Above the electrode array is a microscope lens that focuses images from a nearby computer monitor onto the tissue. Various stimuli can be shown on the monitor; however, a white noise stimulus is the principal tool for mapping the receptive fields[7] of the RGC.

3.2 Cell Class Identification

After spike sorting has been completed, each cluster must be validated according to some objective measure. In addition to the statistics implemented in "Cell-Finder,"

[6] It is interesting to note that the tissue responds similarly when the pigment epithelium (RPE) is left on the tissue. Preparations with the RPE left on the tissue have the advantage of being able to withstand greater exposure to light since bleached photoreceptors can regenerate.

[7] Receptive fields correspond to areas in visual space to which a particular cell responds.

there are three important methods used to determine the quality of a particular representation of a cell in data. The first method is the autocorrelation function (ACF). By cross-correlating the data surrounding individual spikes corresponding to a candidate cell with each other and creating a histogram, a distinct shape emerges. A certain number of bins should have zero activity in them—as that time interval corresponds to the refractory period for the cell. Consequently, one can determine if the candidate cell is "contaminated" with noise or activity from other cells by examining the ACF and seeing if a well defined refractory period exists. Throughout the paper, this technique (referred to as "contamination") was used to validate most of the algorithms evaluated.

Another technique that is commonly used to evaluate the quality of spike sorting is the generation of receptive fields (RFs) for entire morphological classes of cells. By examining the quantity and quality of receptive fields present in a dataset, we have a solid means of comparing the efficacy of different algorithms that is independent of the inner workings of the techniques themselves. For example, comparing the net output of unique, high quality (low contamination and high spike rate) receptive fields allows for the comparison of a new representational algorithm, integrated with the other components of the current analysis procedure, with the "union analysis" result (which was based primarily on manipulating clustering parameters). Such a comparison was conducted and can be seen in Section 5.4.

3.3 Spike Triggered Average (STA) Computation and Receptive Field Fitting

Receptive fields were calculated by computing the Spike Triggered Average (STA). After the presentation of a "white noise" stimulus as described in (Chichilnisky, 2001), the STA can be used to determine the receptive fields for the cells present in that preparation. This process involves pulling a frame from the white noise stimulus movie

50 ms before each spike corresponding to a particular cell. If the movie frames that were extracted are averaged, the resultant image shows the receptive field (see Figure 6). An ellipse corresponding to the cross section of a Gaussian can then be fit to the data. As a consequence of mapping hundreds of receptive fields in a single dataset, morphological classes can be separated from each other. This is possible due to significant receptive field differences between ON and OFF parasol, ON and OFF midget, and other RGC types (Chichilnisky and Kalmar, 2002). Groups of receptive fields for individual classes form "mosaics" such that interdigitation and overlap between neighboring cells is minimal. Many receptive field mosaics can be seen in Figure 2.

3.4 Software Development

While the spike sorting system is currently implemented in Java, the analyses described here were conducted in MATLAB (The MathWorks, Natick, MA). This allowed for the creation of a framework wherein each component in the spike sorting system could be easily interchanged for different algorithms. Unfortunately, this approach resulted in reduced performance compared to development in a different programming environment. However, some of these performance issues were remedied by calling functions already implemented in Java from within MATLAB whenever possible.

4. Overview and Literature Review

4.1 Standard Analysis Procedure Overview

The current solution to systematic deficits in the spike sorting system is to manually view individual electrodes and evaluate the performance of the spike sorting on individual electrodes. This is accomplished using a tool known as Cell-Finder developed by Jonathon Shlens and myself. Cell-Finder automatically classifies clusters into categories based on their quality. The user can then use this information to determine which electrodes must be reclustered by hand[8]. Unfortunately, Cell-Finder does not actually correct any of the pathological issues with the current spike sorting system. It merely provides a means for the user to intervene and attempt to manually correct the shortcomings of the system. As a result, in order to actually improve the efficacy of the standard analysis procedure, it must be changed in some way.

Therefore, the standard analysis procedure will now be described prior to examining methods for improving the system (see Figure 1 for an overview of the system). Starting with raw electrode traces, one must quickly extract candidate action potentials from vast amounts of data. This is accomplished by applying a threshold to the data (usually at 3-4σ). Each time a sample is recorded with a magnitude greater than the predefined threshold value, it is labeled as a threshold crossing event. Each threshold crossing event is then turned into a vector comprised of samples drawn from around it to create a candidate spike vector. Additionally, due to the high electrode density in our arrays, a cell detected on one electrode is likely also detected on neighboring electrodes. Consequently, six other candidate spike vectors are created by examining a time interval

[8] The features added to this software will be further examined in Section 5.

around the threshold crossing event on all neighbor electrodes. After this, we have seven representations of the same time window across seven electrodes. These representations are concatenated into a single high dimensional candidate spike vector[9]. This process is accomplished in the "thresholding level" of analysis shown in Figure 1.

Once these vectors have been constructed for each threshold crossing event, Principal Component Analysis (PCA) is used to find a projection of the data suitable for clustering (this is the "representational level" of analysis). This approach is usually successful and is rather fast due to the nature of PCA. However, there are various problems with the representations generated by PCA in some situations (these will be discussed in Section 4.2). Next, a highly parametric algorithm guesses the number of clusters present on the electrode in preparation for the actual clustering of the data (explained in detail in Section 4.4). Following this, the data is clustered with a mixture of Gaussians model using the Expectation Maximization algorithm. These two steps, estimating the number of classes and performing clustering, comprise the "clustering level" of analysis (from Figure 1). In each component of this system, there are clear advantages and disadvantages. It is likely that a more optimal system can be constructed such that both computational cost and error can be minimized. Consequently, a number of other possible methods for each stage of the spike sorting process will be discussed.

At the thresholding level, it seems as though few improvements can be made. Currently, the standard deviation is calculated for each electrode. The threshold is then set to some multiple of this sigma value (usually 3 or 4) that has been obtained by testing various values across datasets. While some real data is certainly lost in this process,

[9] Each candidate spike vector is of length (window_length^2 + 1) $*$ neighbor_electrodes, where window_length typically = 13, and neighbor_electrodes is 6.

analyzing all of the noise data in an attempt to find more spike events would be computationally infeasible. However, it is important to note that the sigma value chosen has a strong effect on how well the rest of the system performs.

This was demonstrated in the union analysis first performed by Jeffrey Gauthier. The current spike sorting system was run 24 times on the same dataset. After this was completed, receptive fields were generated and duplicate cells were discarded. The net yield of unique cells was compared to a single iteration of the system using a common parameter set (see Figure 2). In the analysis, three different threshold values were chosen—each threshold was applied to eight iterations of the analysis. The highest threshold (indicating that the largest amount of candidate spikes was discarded) yielded the most unique cells from the analysis. This perhaps is indicative of the fact that physiological noise from nearby cells may interfere with PCA and clustering. However, even if that is not true, it certainly indicates that thresholding is an important component of the analysis to consider improving.

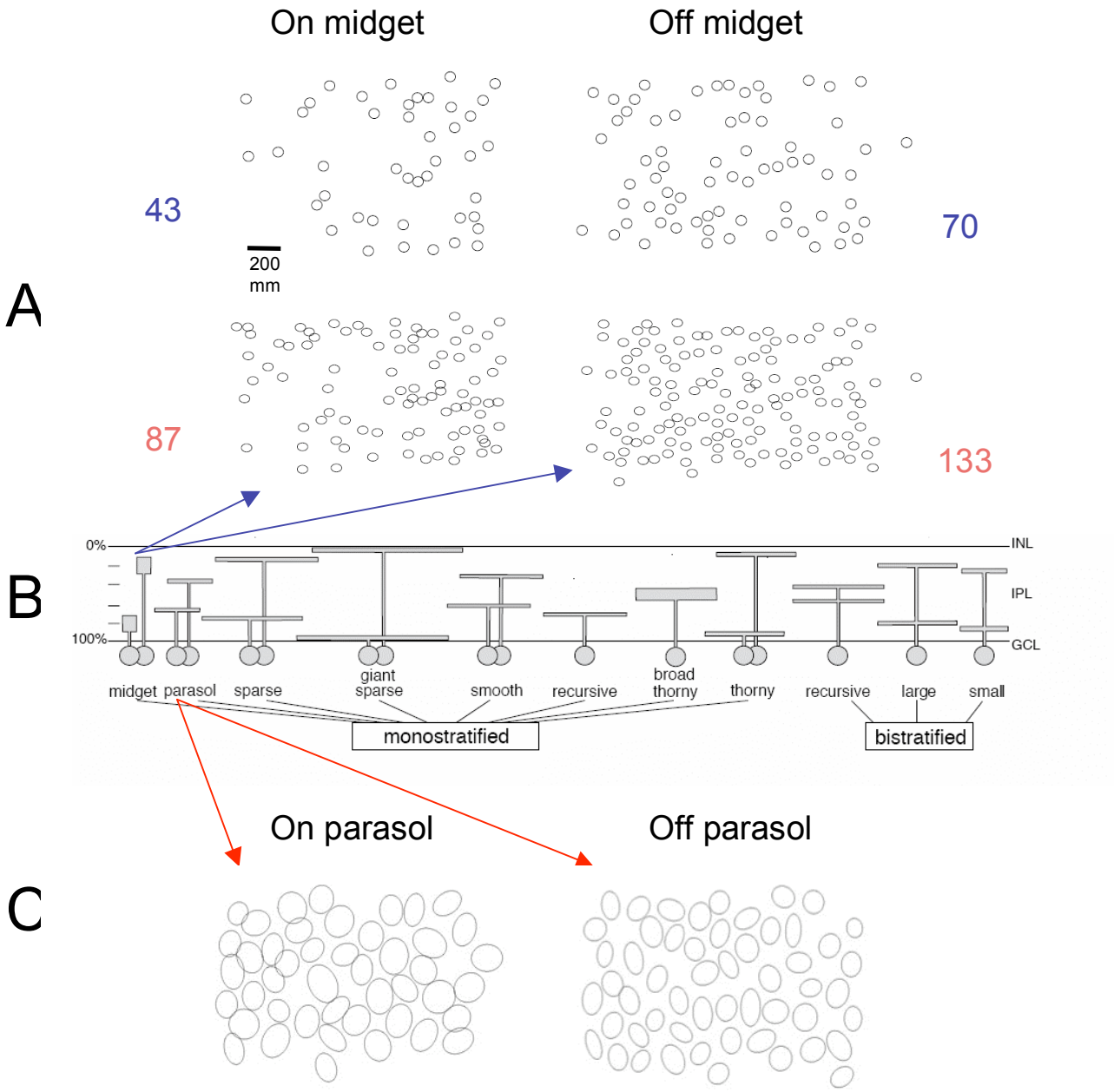

Figure 2– Two mosaics of midget receptive fields (represented by Gaussian fits) can be seen above. The first two mosaics represent the output of the standard clustering procedure run a single time (A, top). The lower two mosaics (with 87 and 133 more cells, respectively, than the top mosaics) show an estimate of the upper bound for spike sorting performance. They were created by running the standard analysis procedure 24 times with various parameter sets (provided by Gauthier, unpublished data). B shows the known morphological classes of Retinal Ganglion Cells (RGC) (adapted from Dacey and Packer, 2003). C: the parasol mosaics can be seen for the same dataset can

be seen when the analysis was run a single time. Union analysis reveals only 2 more ON cells and 3 more OFF cells.

4.2 Approaches to Improving Spike Sorting

There are many unique approaches to thresholding that may be worth implementing in the future. For instance, Rebrik and colleagues demonstrated an interesting technique (Rebrik et al., 1999). Instead of calculating a fixed threshold for use on each electrode, they created a thresholding surface based on the covariance between the different electrodes of the tetrode. Thus, more salient parts of the data would be more likely to cross the threshold than less important components (as measured by the cross-channel covariance matrix). Using this technique on our setup would likely provide some benefit, however, it is not clear whether or not the benefits will be worth the computational cost—especially due to the large size of our arrays and the fact that the SNR of our system is much higher than is typically seen in tetrode recordings. Nevertheless, this is certainly worth investigating further in light of the results from the union analysis.

After the raw data has been thresholded and individual candidate spikes have been isolated, the data is in a very high dimensional space (as described in the previous section). The algorithm used in our system, Principal Component Analysis (PCA), is a popular technique as it is straightforward to implement and is non-parametric. Unfortunately, PCA does not find independent components from source data: it merely minimizes the covariance. Since a zero covariance does not imply independence, PCA is unable to uncover many third order redundancies. Brown and colleagues discuss a three dimensional data set wherein the third dimension is simply the exclusive-or of the first two dimensions. PCA fails at reducing the dimensionality of this system as the

covariance between the third dimension and the first two dimensions is zero (Brown et al., 2001).

As a consequence of this shortcoming in PCA, the algorithm is unable to isolate many cells adequately—especially when cells are bursting or firing synchronously with other cells. When PCA fails to separate the signal components satisfactorily, it is very difficult to create distinct clusters that represent individual neurons in the clustering stage of spike sorting, regardless of the clustering strategy used. As a result, investigating a more effective means to represent the data in low-dimensional space is likely the level at which the greatest improvements in spike sorting will be seen. Additionally, since there is no way to manually compensate for these deficits (unlike in clustering), most of the cells lost due to problems with PCA cannot be recovered even with the union analysis.

PCA chooses the first component along the axis with the greatest variance in the dataset, as clustering is usually most successful among datasets where the data is maximally spread out (Sahani, 1999). The other components found by PCA are then orthogonal to this first dimension and are sorted by variability. Unfortunately, in some instances, the component with the maximum variance in a dataset does not correspond to the component that maximally separates clusters. For instance, imagine a two dimensional dataset composed of two long, thin, distinct, clusters that lie parallel to each other in space. In this example, a line drawn in the parallel to two clusters would represent the greatest variance present in the dataset. However, a line perpendicular to the line of greatest variance, in this specific instance, would best discriminate the data. Forming a data projection along this “linear discriminant” would yield a representation of the data that is much easier to cluster. Unfortunately, without knowing the correct cluster

assignments *a priori* it would be impossible to actually compute the linear discriminant by any means. Such a calculation of the multi-class extension of Fisher's linear discriminant, known as Linear Discriminant Analysis (LDA), would require knowledge of the within-class and between-class covariance matrices.

4.3 Candidate Representational Techniques

Fortunately, an approximation of the result of LDA can be obtained using an estimate of the within-class covariance matrix (Σ_W) to find the "class-whitened" total covariance matrix for the dataset ($\Sigma_{T,\ white}$). By diagonalizing $\Sigma_{T,\ white}$ the principal components of this new class-whitened space can be obtained (Sahani, 1999). In most circumstances, by sampling from the noise distribution[10] and finding the covariance matrix, an accurate approximation of Σ_W can be created. After creating this estimate, $\Sigma_{T,\ white}$ can be created by applying the whitening filter:

$(\Sigma_{W,\ estimated})^{-1/2}\ \Sigma_B\ (\Sigma_{W,\ estimated})^{-1/2} = \Sigma_{T,\ white}$ (where Σ_B is obtained by finding the covariance matrix of the candidate spike vectors). This technique will likely be effective as long as two assumptions are met. First, background noise must be assumed to be additive such that $\Sigma_{W,\ estimated} + \Sigma_B = \Sigma_T$ (where Σ_B is the between-class covariance). This means that the noise signal is assumed to be additive at all points in the recording—even during spike events. Consequently, we must also assume that the noise distribution is independent from the spike data distribution. If this second assumption is not true, and the noise distribution (represented by our estimate of Σ_W) is actually correlated with spike events, then the whitening operation could degrade the visibility of the signal components we are trying to isolate. For purposes of clarity, this "whiten and diagonalize"

[10] In terms of our spike sorting system, we can sample from background noise by randomly choosing time intervals that are adequately far away from above-threshold spike events.

approximation of the result of LDA will be referred to Noise Whitened PCA (NWPCA). This technique was described by Maneesh Sahani in his thesis (Sahani, 1999) as well as in previous publications by others (see Ripley, 1996). However, it is infrequently mentioned in spike sorting literature. The results from integrating this algorithm into the current spike sorting system and comparing the system to the standard analysis procedure (which includes standard PCA) can be seen in Section 5. In general, the spike sorting system isolated more cells successfully when using NWPCA to represent the data. For instance, in one dataset, 25 OFF midget cells were found that the standard analysis procedure did not reveal.

Since PCA is a very computationally efficient algorithm, and is already implemented in the current spike sorting system, Noise Whitened PCA was an interesting algorithm to investigate. If the algorithm showed even a modest improvement over standard PCA a strong argument could be made for full scale implementation—especially since the computational cost is only marginally higher than standard PCA. However, since Noise Whitened PCA is still fundamentally PCA, it is inherently susceptible to similar problems. Consequently, completely different approaches (such as Independent Component Analysis) that work based on different assumptions needed to be evaluated as well.

The Independent Component Analysis (ICA) algorithm works by minimizing mutual information between dimensions and can thus recognize many third order redundancies (such as the exclusive-or example given earlier). ICA also offers many other advantages over PCA, such as the ability to separate signals from synchronously firing cells, as well as the ability to identify spikes of a stereotyped waveform that are

different sizes from each other (such as when a cell is bursting) as being from the same cell (Brown et al., 2001). Unfortunately, ICA is more complicated to implement and does not always converge on the same "independent components"—much unlike how PCA functions. ICA is also very sensitive to noise, and fails if the number of signals in the dataset outnumbers the number of detectors (this is known as the completeness problem). Many of these issues are addressed in greater detail elsewhere (Sahani, 1999).

Despite these caveats, the potential advantages of ICA are very great. Indeed, Brown and colleagues described the performance of ICA as a "success" when it was used on data provided by our lab (Brown et al., 2001). Additionally, there are many freely available implementations of ICA such as the popular "Fast ICA" package for MATLAB[11]. As a consequence of this, it is rather straightforward to gauge how useful ICA is as a candidate component of our spike sorting system.

The same individuals that developed "FastICA" created a MATLAB package called "Icasso" to automatically separate the IC components that represent actual data from the components that are merely artifacts from the ICA algorithm itself (Himberg et al., 2004). This is accomplished by running ICA iteratively with a different parameter sets on each iteration. After this process has been completed, each IC component that was generated is plotted in space based on mutual similarity between components. Agglomerative clustering is performed subsequently. Dense clusters then represent IC components that are similar to each other (suggesting that they may be real sources as different parameter sets yielded similar components). This seems to be a powerful technique as it solves the problem of not being able to automatically sort IC components by quality (as can be done with PCA). Additionally, the average waveform of a particular

[11] FastICA can be obtained at http://www.cis.hut.fi/projects/ica/fastica/

cluster (the “centrotype” vector) is automatically calculated. These centrotype vectors are likely better representations of the data than any single IC component from the cluster used to create the centrotype.

4.4 Improving Clustering

Beyond the dimensional reduction stage of the analysis, the final stage of spike sorting to consider is the process of clustering. It is important to emphasize that the difficulty of this step is entirely dependent on the quality of the output from the dimensional reduction algorithm. If the data is not represented in a reasonable way, no clustering algorithm will be able to isolate individual cells. Currently, a mixture of Gaussians model is fitted to the data present on each electrode using the Expectation-Maximization (EM) gradient descent algorithm. This technique works reasonably well as it has a low computational cost and has a probabilistic model behind its output. However, it provides no means for estimating the number of clusters present on an individual electrode.

In order to do this, a rather simple technique known as the “water filling algorithm” was developed by the lab. According to a set of parameters defined for each dataset (and changed infrequently), a histogram is calculated for all the points present on an electrode. Then the number of clusters (and centroid location for each) is defined by the bins in the histogram that exceed a certain threshold. Due to the parametric nature of this technique, as well as its rather primitive nature, most of the problems with the current clustering code can be attributed to this “water filling algorithm.”

One possible technique that could be used instead of mixture of Gaussians involves the use of Markov Chain Monte Carlo (MCMC) methods to reliably estimate the

number of clusters on a given electrode before the algorithm forms clusters (Nguyen et al., 2003). While the specific implementational details will not be discussed here, it seems as though this might be a useful technique for improving clustering. Since the algorithm uses Monte Carlo methods, it can be easily distributed to many computers by simply running a certain number of random walkers on different machines simultaneously. However, the method is rather new and has not yet been tested on our data. When the focus of this work shifts to the clustering level, this method will be considered more closely.

Another clustering technique that could be used in lieu of the current system is spectral clustering (Ng et al., 2001). Instead of estimating the number of clusters present on the electrode directly, the spectral clustering algorithm can simply be run a number of times with different k values (number of clusters) chosen. Each of the resultant sets of clusters is then evaluated based on how dense the clusters present are. The solution where cluster density is maximized can then be chosen as being the best fit to the data. This technique seems promising as it should be straightforward to implement (needing only a "few lines of MATLAB"), and is very efficient computationally (Ng et al., 2001).

5. Findings

5.1 Cell-Finder Development

As first discussed earlier in this document, one of the first attempts to improve the current state of the spike sorting system can be seen in the development of the Cell-Finder software package (Figure 4). Implemented in Java, the original goal of the tool was to provide the user a means of manually examining the performance of clustering on each electrode in the dataset. The user had to manually examine random electrodes of interest to determine whether clustering was acceptable. This was a very time consuming task that needed to be improved. In the past year, a number of new features have been added in an attempt to automate much of this process. This was accomplished by the addition of various statistics that evaluate the quality of individual clusters. These statistics can indicate which clusters are already of high quality, and which ones demand attention from the user.

A number of useful statistics were added, but a few are especially worthy of note. The first of these is the "fragmentation index" statistic that was designed to determine when the clustering algorithm (more specifically, the "water filling algorithm" that guesses the number of clusters present on an electrode) erroneously split datapoints belonging to a single cell into multiple clusters. This is accomplished by computing the contamination index (based on the number of autocorrelation function refractory period violations) between each pair of clusters present on an electrode. If the contamination index of all of the spikes present in a pair of clusters is as low as the contamination index of each individual cluster, it is likely that those two clusters of points are actually from a

single cell. In such situations, Cell-Finder alerts the user who can then quickly use the new “union cluster” feature to combine the two clusters.

Another useful statistic leverages the probabilistic framework built into the mixture of Gaussians model used by the spike sorting system. When the Expectation Maximization (EM) algorithm assigns datapoints to clusters, it does so based on the greatest likelihood value found for each datapoint. This likelihood value indicates that the point belongs to a specific cluster more strongly than any other. In this way, the “best likelihood ratio” statistic was created to determine how well isolated a particular cluster is. In the process of these cluster assignments, the EM algorithm saves out all computed likelihood values. This statistic then simply finds the second highest likelihood value for each cluster and computes log(highest likelihood / second highest likelihood) for each point in the cluster. The mean of this value is then reported to the user. Higher resultant values indicate that a particular cluster is well isolated. When used in concert with other statistics such as the fragmentation index and contamination, the quality of a particular cluster can be strongly quantified.

However, such numbers are not particularly useful at reducing time spent manually analyzing data if the user must manually compare these statistics to each other. As a consequence of this, the “schema” feature was implemented. Schemas allow the user to choose a set of predefined ranges for each statistic calculated by Cell-Finder that is deemed acceptable. Within the acceptable range, there can be up to four sublevels that represent the degree to which a particular cluster is acceptable. These levels are then used to sort the clusters (and associated statistics) in a table, as well as to color code each cluster based on its quality as determined by the schema. Consequently, in order for the

user to determine the quality of a particular cluster he merely has to examine the color of that cluster in the table (Figure 4, B and C).

These new features have been reasonably successful at accelerating the once arduous task of evaluating the efficacy of clustering. However, the fact that such an intricate system is necessary to compensate for deficits in the clustering system indicates there is certainly a need to investigate using new techniques. At this point, let us turn to a discussion of progress made in improving the representational level of the spike sorting system.

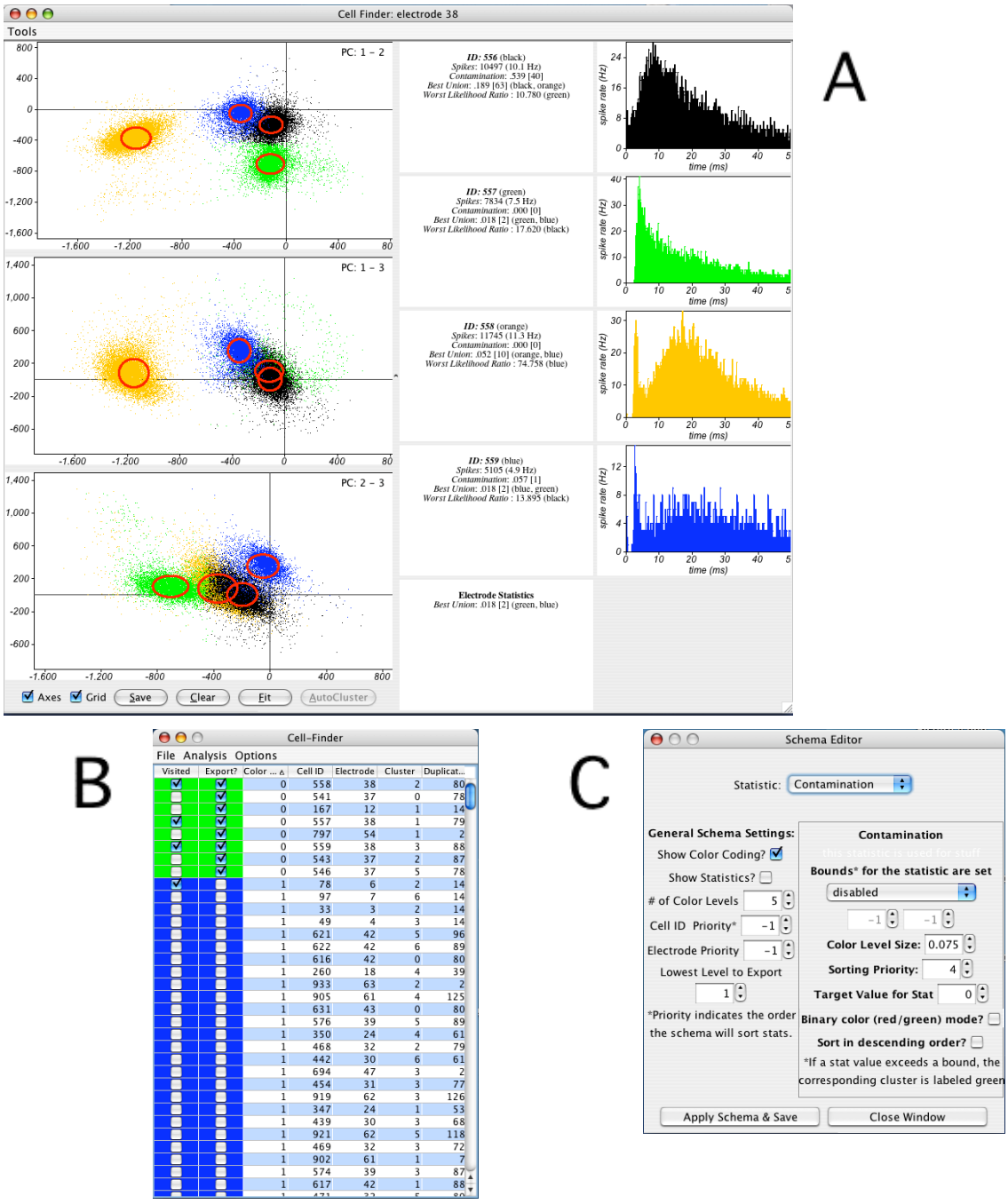

Figure 3 - Cell-Finder software allows for the user to quickly evaluate the performance of the spike sorting algorithm. This is accomplished in multiple steps. First the user can manually examine the representation and clustering of specific electrodes (A). The cluster centroids can be reset manually by the user. Alternatively, the user can browse through a list of every cluster present in the dataset sorted according to various statistics that evaluate cluster quality (B). A combination of thresholds (known as a schema) placed on various statistics can be used to automatically color code each cluster based on the thresholds defined by the user (B and C).

5.2 Kohonen's Self Organizing Map (SOMs) and Other Techniques

In addition to the various representational techniques that were introduced in Section 4, a number of techniques that are not typically used in contemporary spike sorting systems were considered for use. One such technique, Kohonen's Self Organizing Map (SOM) is a commonly used unsupervised learning algorithm that can be used for dimensional reduction. Another laboratory with a much less complicated spike sorting system successfully applied SOM as a means of correcting the output from ICA (Hermle et al., 2004). Intuitively, the behavior of the behavior of the network can be described as a low dimensional manifold (representing the network) that contorts itself to the distribution of datapoints present in higher dimensional space. Each point is then assigned to the nearest network node. In this way, the output from the algorithm is somewhat similar to more complicated manifold learning techniques such as multidimensional scaling (MDS) and various other nonlinear dimensionality reduction techniques such as Isomap (see Tenenbaum et al., 2000).

This technique was tested on our data using a MATLAB implementation of Kohonen maps available directly from Teuvo Kohonen's laboratory.[12] Directly applying the algorithm to our data was relatively straightforward. Candidate spike vectors from an individual electrode were simply provided as input to a 10 by 10 unit network. After the algorithm converged, k means clustering was performed on the network nodes themselves with various k values. Since there are so few units (100) compared to spike vectors (thousands), it is possible to successfully apply a simple algorithm such as k means. Additionally, it is possible to use a subset of the data present on an electrode as a training set for the network, in order to improve computational performance.

[12] The SOM Toolbox for MATLAB can be obtained from http://www.cis.hut.fi/projects/somtoolbox/

The results from applying SOM to a single electrode can be seen in Figure 4. Note that the algorithm located a single well isolated cluster, and only a small amount of structure is present throughout the rest of the network. This can be seen by examining Figure 4 B where the distances from each node to its neighbors can be seen. Since each node has many neighbors, each individual unit is represented as series of hexagons—where each hexagon represents a pairwise distance between neighbors. Darker colors denote shorter distances. Thus, it is possible to see that the units in the top right of Figure 4 B are well isolated from the rest of the network. These units correspond to the yellow cluster found in Figure 4 A.

In order to compare this representation to PCA, the datapoints assigned to the best isolated cluster (the yellow cluster in Figure 4 A), were plotted in terms of the first two principal components of the electrode. Since the yellow cluster was so well isolated, it was hypothesized that it would be represented by a very well isolated cluster in PC space. Indeed, the analysis shows that the points assigned to the yellow cluster by SOM fall within a well isolated cluster in the PC plot (Figure 4 C). Unfortunately, SOM failed to associate as many points with the well isolated cluster as did PCA (both algorithms used the same number of input datapoints). Even if the yellow cluster was enlarged by including units from the green cluster, it is unlikely that a better result would be seen. The black hexagons in Figure 4 A show this: larger hexagons indicate more datapoints assigned to a unit. Since so few points were assigned to the green cluster (see the small black hexagons within the green cluster shown in Figure 4 A), combining the green and yellow clusters would not result in a significant change. Perhaps better results could be obtained if a different parameter set were used for the network. Unfortunately, it is

important to note that performance degrades quickly as the size of the network is increased.

While the potential benefits of using an algorithm such as SOM are great, such an implementation would necessitate broad changes to the infrastructure of the current spike sorting system. For instance, all techniques currently used to evaluate cluster quality (such as those implemented in Cell-Finder) would be inapplicable since a mixture of Gaussians model is not used for clustering. Indeed, it is not even known whether such an implementation would be comparable (or better than) to the current techniques used in terms of computational cost or cell yield. Despite the fact that some optimizations can be made to improve the performance of SOM, it is highly unlikely that the algorithm would ever be nearly as fast as PCA. Consequently, work investigating this technique was not pursued further. However, if improvements in cell yield could be seen using this technique (or similar nonlinear dimensional reduction techniques), creating a secondary spike sorting system devoted to maximizing cell yield at any computational cost could be envisioned.

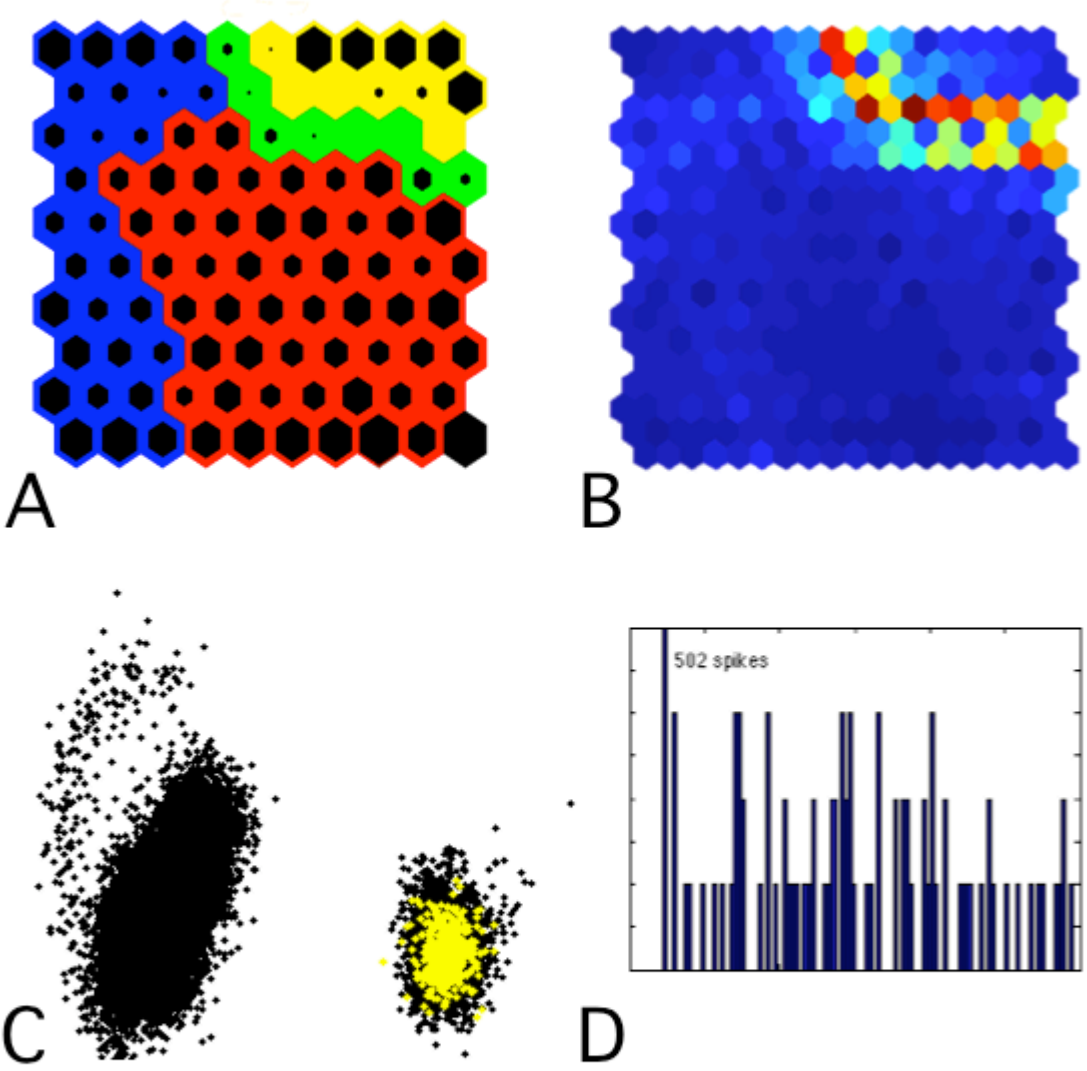

Figure 4 - Results from the application of a Kohonen SOM (implementation provided by Esa Alhoniemi, et al.[13]) to 512-electrode recording data can be seen above. The hit histogram indicating the number of points assigned to each node of the network can be seen in A (larger black hexagons indicate more points assigned to that particular node). The colors indicate the result of K-means clustering on the network, four clusters were chosen in the example shown. Image B shows the pairwise distance between neighboring units in the network. Darker colors indicate shorter distance between adjacent nodes. Since the yellow cluster (from A) is clearly the best isolated (as can be seen in B), the points assigned to the yellow nodes are plotted in PC space (C). The uncontaminated autocorrelation function of the yellow points can be seen in D.

[13] The SOM Toolbox can be obtained from http://www.cis.hut.fi/projects/somtoolbox/

5.3 Iterative ICA

One of the more promising techniques for improving the representation of our data involves the use of ICA. Indeed, this technique was evaluated on our dataset using the FastICA implementation along with the Icasso algorithm introduced in the previous section. As predicted (Brown et al., 2001), this technique was indeed successful at isolating cells from raw electrode data. Unfortunately, the process of running ICA on various parameter sets through all of the candidate spikes present on a particular electrode is a very time consuming task. The end result of this procedure yielded results that were qualitatively similar to what could be obtained using the current system. Perhaps large gains were not seen as the algorithm was applied to various individual electrodes, rather than whole datasets. Nevertheless, preliminary results were not especially encouraging—especially considering the computational cost. For instance, when the iterative ICA technique (described below) was applied to all of the datapoints present on individual electrodes, the process yielded no well isolated cells that were not detected by PCA.

Let us now consider the process used to apply ICA to our data (see Figure 6 for diagram of the process when run on a subset of data). After the data has been thresholded, candidate spike vectors are extracted from the electrode being examined as well as from the six neighbor electrodes. Each of the candidate spike vectors corresponding to an individual source electrode are then combined into single "source" vectors. These seven source vectors are then used as input for the Icasso wrapper to the FastICA algorithm. Icasso was set to run the FastICA algorithm ten times, each time varying the initial conditions of the ICA algorithm. Additionally, the data are "bootstrapped" on each iteration. This means that the data are resampled by a random increment such that the

shape of the data will be forced to change (Himberg et al., 2004). This helps to stop the algorithm from consistently converging in the same local minima—which is a possibility even with random starting parameters.

Once ICA has converged on the tenth iteration, we are left with 7•10 (input components • number of iterations) IC components. Each of these components is then plotted in space and clustered using an agglomerative algorithm implemented within Icasso (the number of clusters was set to 20 prior to beginning analysis; see Figure 5 A, right). Following this, the density of all of the clusters was automatically evaluated and centrotypes corresponding to the centroid vector of each cluster were computed (see Figure 6 A, left). The centrotypes corresponding the densest seven clusters were then manually saved out from this analysis. Only seven components were typically analyzed in further detail because if we assume that the IC component evaluation technique in Icasso works well, and recall that ICA cannot separate more signals than there are input sources, the rest of the centrotypes not likely to be real data.

Each of the saved IC components was then analyzed using PCA. Since each input source was composed of a number of vectors of spike window length, there is a representation of n spikes present (where n represents the total number of spikes present on an electrode) in each IC component. Thus, when PCA is run on an IC component a result similar to Figure 5, part B is seen. There should be two distinct clusters: a large center cluster representing all the spike vectors with magnitudes close to zero, and a second, well-isolated cluster containing all of the spike vectors that hopefully contain real data. While Figure 5 was generated using a subset of data present on an electrode, an uncontaminated ACF can still be seen—in addition to a weak image of its receptive field.

Despite the fact that the results obtained from applying iterative ICA to the dataset were not especially encouraging, a more rigorous evaluation may yield more useful findings. Now that the process of implementing an automated framework for testing this algorithm has been completed, the labor involved in rigorously evaluating this technique would be reasonably small. Consequently, the question that remains is whether the computational cost of applying this approach to our data is justifiable. Since many other groups have demonstrated success with this technique (Brown et al., 2001; Hermle et al., 2004), it seems as though further investigation is warranted at a later date. It is possible that there are tremendous gains possible with ICA that we have simply not seen. Indeed, there are still a number of different variations of the ICA algorithm that need to be evaluated. A comparison of different non-linearity functions, for instance, could be especially useful at demonstrating the efficacy of this technique more thoroughly.

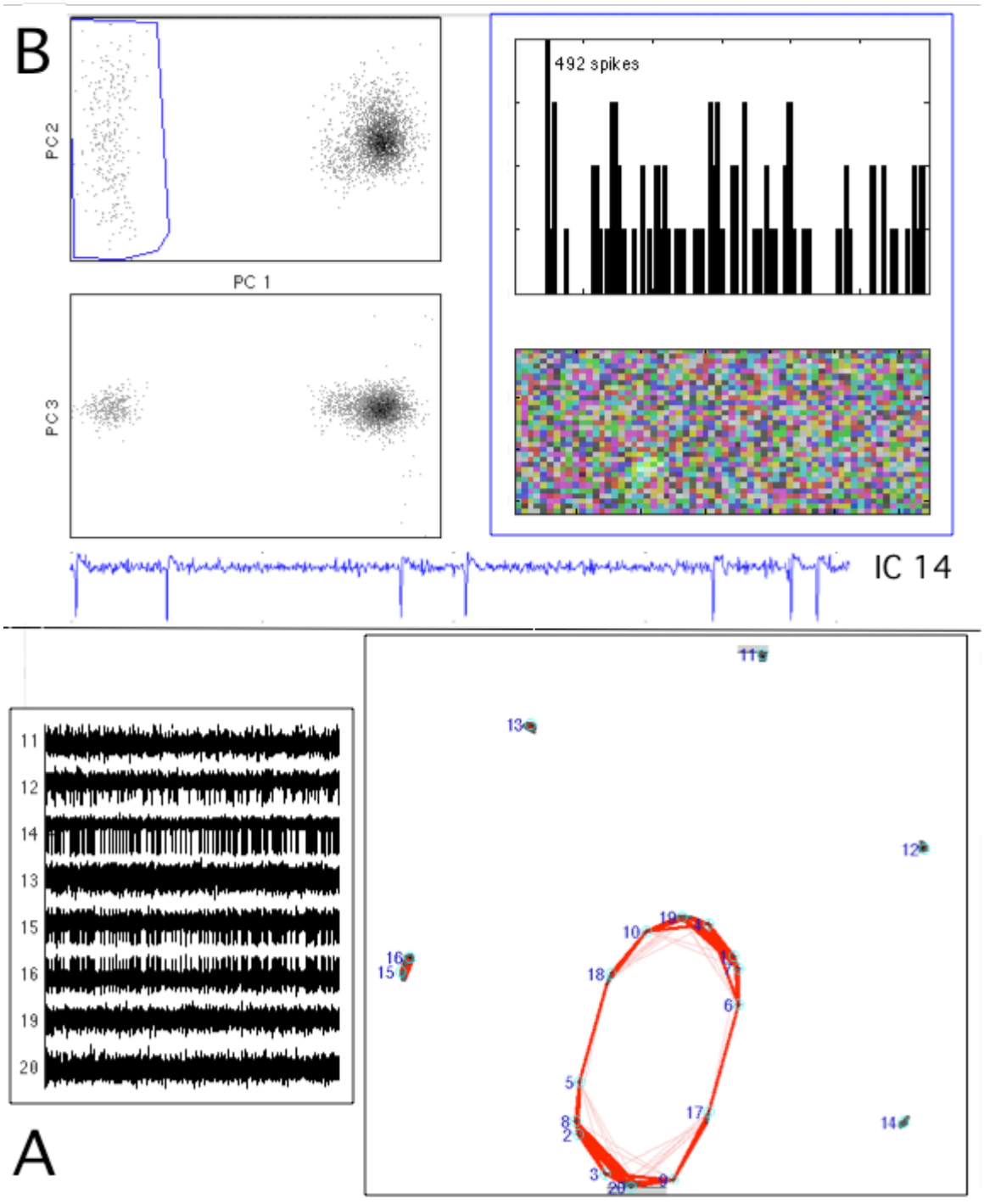

Figure 5 - The results from applying iterative ICA to array data can be seen above. The algorithm implementation used was FastICA with the Icasso package for MATLAB (see Himberg et al., 2004). The bottom half (A) of the figure shows the result from running ICA a number of times with different parameter sets, each time generating seven independent components. Each component was plotted in space (A, right) and clustering was performed. Compact clusters indicate strong similarity between all IC components within the cluster. The mean waveform of cluster 14 plotted in PC space (B, left) can be seen along with its autocorrelation function (B, right top) and receptive field (B, right bottom).

5.4 Rigorous comparison between Noise Whitened PCA (NWPCA) and PCA

The most successful technique evaluated over the course of this project was Noise Whitened PCA (NWPCA). The technique is commonly used to approximate the result of Linear Discriminant Analysis (Ripley, 1996; Sahani, 1999)—although it is not widely applied to spike sorting systems. It is also rather straightforward to compute as it only involves finding the principal components of a whitened covariance matrix created by sampling from the noise distribution (as described in Section 4). The only significant added computational load is the actual sampling from the noise distribution. Since testing for this algorithm was conducted within MATLAB, the process of importing raw electrode data was rather time consuming and hindered the ability to accurately determine how much slower the algorithm is compared to standard PCA. Nevertheless, the computational cost of this technique should be similar standard PCA if implemented efficiently.

In the evaluation of this technique, the two datasets were analyzed using the standard analysis procedure with PCA replaced by NWPCA (hereafter referred to as the "new analysis" procedure). When comparisons between the standard analysis and the new analysis were made, both spike sorting systems used the same thresholding and clustering parameters. The number of cells located by the new analysis procedure far exceeded the amount found by the standard spike sorting system using PCA.[14] NWPCA was especially successful at separating pairs of midget cells from each other. This can be seen in Figure 6. The bottom of the figure shows an electrode in which the standard analysis procedure failed to isolate two midget cells. (Figure 6 B1). These poorly isolated cells are represented in PC space by the green cluster (bounded by a convex hull in the

[14] In one dataset analyzed using the new analysis procedure, 37 midget cells were extracted that were lost in the standard analysis.

bottom of Figure 6 B). If this cluster is examined closely, there only seems to be a single region of high density (shown by dark points) within the cluster—indicating that the spikes representing the two midget cells are likely overlapping in PC space. This suggests that any clustering algorithm would have trouble separating the two cells. However, if one examines the representational output from NWPCA (Figure 6 A), the datapoints from the two cells are separated from each other in space. Consequently, the clustering algorithm was able to extract the two midget cells successfully.

In addition, the mean spike waveform plots (shown to the left of the autocorrelation function plots in the Figure 6) indicate that the largest spike waveforms for the midget cells were detected on the electrode shown in the figure. Consequently, PCA failed to separate the two cells from each other on the electrode where it had the best chance of doing so—indicating that PCA likely fails to find many midget cells like these. This is further supported by Figure 7 B where two mosaics of midget cells were obtained by running the union analysis on a dataset. Since the standard analysis is run 24 times in the union analysis procedure, many cells lost due to the parametric nature of the clustering[15] and thresholding levels of analysis can be recovered. Consequently, any cells missed by the analysis are likely not isolated due to deficits in the representation of the data. The blue ellipses overlaid upon the union analysis mosaic in Figure 7 B show the receptive fields for cells that were found by a single iteration the new analysis procedure. The union analysis was unable to find the cells represented by blue ellipses. This provides strong evidence that NWPCA can represent data better than PCA in many cases.

[15] More specifically, it is the "water filling" algorithm that guesses the number of clusters present on an electrode that is highly parametric. The actual clustering algorithm is nonparametric.

The data in Figure 7 A further support the idea that it was not mere luck or an artifact from clustering that separated the two midget cells in Figure 6. Analysis of a second dataset using the new analysis procedure revealed 11 new ON midget cells and 26 OFF midget cells (Figure 7 A, left shows the NWPCA mosaic; Figure 7 A, right shows the PCA mosaic). While this is indeed a significant improvement over the standard analysis procedure, a rather surprising finding in this dataset is that a single iteration of the standard analysis procedure located 2 ON midget cells and 12 OFF midget cells that the new analysis technique did not find. While the new algorithm still fared well (finding a net gain of 9 and 14 new ON and OFF midget cells, respectively), this result suggests that the current thresholding and clustering parameters must be optimized to work with NWPCA. Currently, the clustering system overclusters NWPCA representations on nearly every electrode and dataset. This strongly indicates that there is a more optimal parameter set to be used. In order to examine the reasons why these cells were missed, their precise location within NWPCA space must be examined in the coming weeks. This will hopefully reveal whether these lost cells were missed due to deficits in representation or clustering.

Currently, more data are being analyzed using this new technique. The implementation within MATLAB is now capable of analyzing an entire dataset without any user intervention. As a consequence, many datasets will be analyzed using this technique very soon. Integrating NWPCA into the standard clustering analysis should also be rather straightforward, especially due to the techniques' similarity to PCA. However, there are a few issues that must be considered. First, it is theoretically possible that the noise datapoints (randomly selected to be far from any spike events) could

somehow be correlated with actual spikes. If this were true, NWPCA would likely generate a worse representation of the data than standard PCA in some cases. While it is not currently known if this occurs, the performance of the algorithm must be examined when using atypical stimuli (such as mouse datasets where a drifting grating was presented to locate direction selective ganglion cells). Additionally, the exact structure of the "noise" distribution used by the NWPCA algorithm must be examined. If the structure of the noise seems to be even somewhat correlated with the structure of the data, an investigation into finding new methods for sampling from the noise will be necessary. Furthermore, the process of integrating this noise finding procedure into the current spike sorting infrastructure will be difficult as it is a completely new step—and unfortunately one that requires access to the massive raw data files. Thus, robustly integrating NWPCA into the standard analysis procedure will be a nontrivial process.

Nevertheless, the results from evaluating NWPCA seen so far are rather exciting. Since this technique is rather simple compared to other nonlinear dimensional reduction techniques, the results suggest that an optimal spike sorting system could likely isolate far more cells than are extracted by current methods. This provides hope for the idea that poorly understood morphological classes may be soon extracted from already existing data.

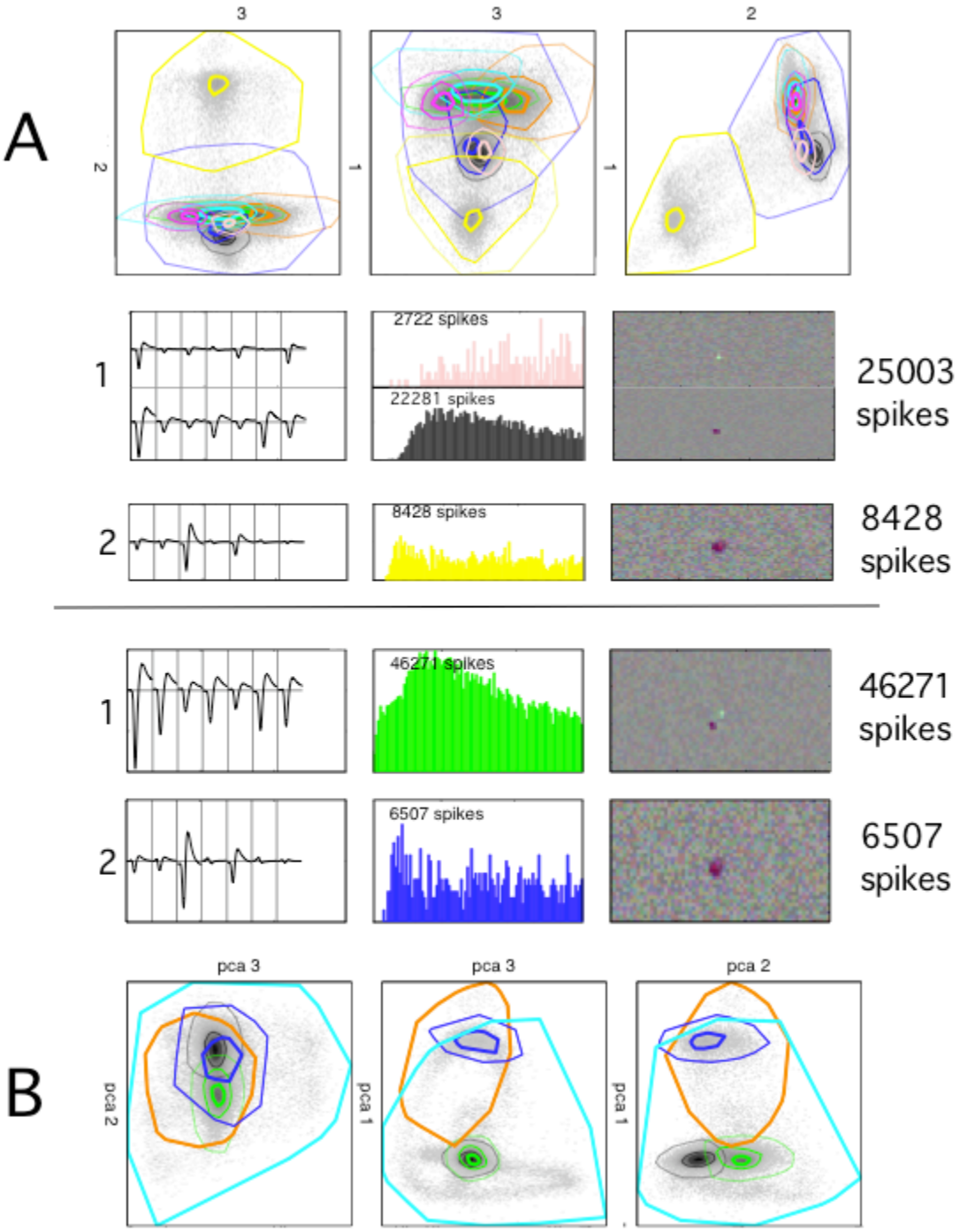

Figure 6 – Results from an analysis comparing Noise Whitened PCA (A) and normal PCA (B). The figure shows the clustering of each electrode. Colored convex hulls and contours show the boundaries and color of each cluster. The clusters that represent an on-parasol cell can be seen in A-2 and B-2. B-1 (normal PCA) shows two contaminated midget cells. A-1 (Noise Whitened PCA) shows the same cells well separated from one another. The plots on the left show the mean spike waveforms across six neighbor electrodes as well as for the center electrode. The first mean waveform is from the center electrode. Note that the amplitude of the signal from the midget cells is largest on the current electrode. Similar results were observed on many other electrodes.

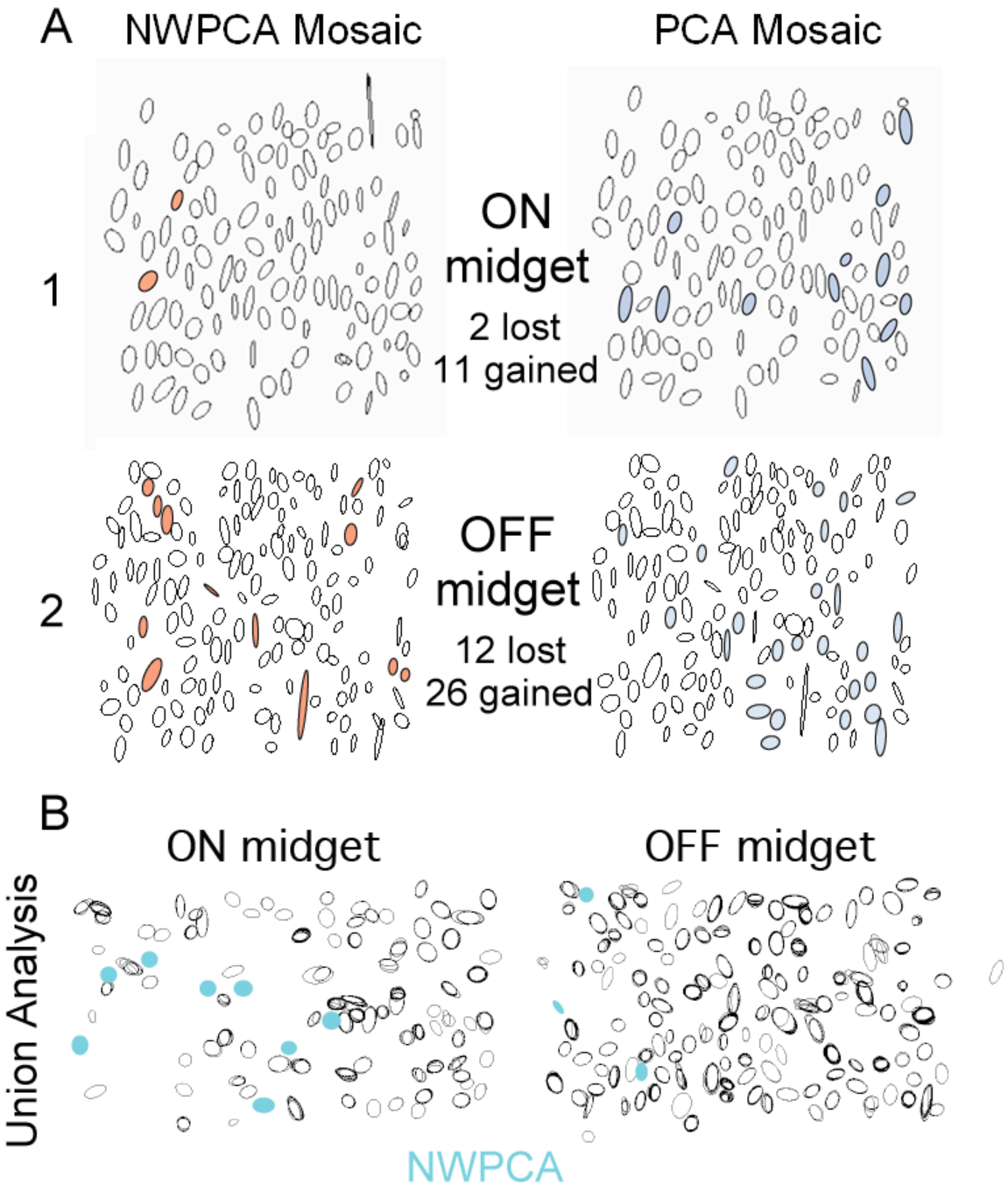

Figure 7 – Parasol receptive field (RF) mosaics obtained from two different datasets (A and B). The mosaics on the left of A were created by using the new analysis procedure (NWPCA generated representation). The right mosaics (in A) used the standard analysis (PCA generated representation). Red ellipses indicate cell receptive fields present in the standard analysis using that were not found in the new analysis. Blue ellipses show cell receptive fields from the left (new analysis) mosaic that were not found in the right (standard analysis) mosaic. In B two midget mosaics were generated by running the standard analysis 21 times and then combining the results into single RF mosaics. When the same dataset was analyzed using the new analysis procedure (for a single iteration), many new cell receptive fields were found (shown by cyan ellipses).

6. Discussion

Over the course of this project, a significant amount of progress has been made towards the goal of improving the representational level of the current spike sorting. While a number of representational techniques were evaluated, the most complete analysis was performed on one of the more simple techniques: NWPCA. However, the results from that algorithm, when integrated to the current spike sorting system, compare very favorably to the standard analysis procedure and even to the union analysis—a result previously thought to represent an estimate of the upper bound for spike sorting on our data. As the analyses here have shown, it seems as though the upper bound for spike sorting performance has yet to be reached. While it would be difficult to search for this upper bound in a manner similar to Harris and colleagues (see Harris et al., 2000), it seems plausible that nonlinear dimensional reduction techniques that have yet to be evaluated will yield many more cells—especially in light of the results from NWPCA, a comparatively simple linear technique.

However, this begs the question of whether maximal cell yield is worth the added computational cost of increasingly complex algorithms. A possible answer to this question is to simply create two parallel systems. A fast system (perhaps using NWPCA) suitable for use during experiments and coarse analysis of data could be constructed in addition to a slower, offline system (Figure 8). This second spike sorting system could make use of more computationally intensive techniques such as iterative ICA, and perhaps a Monte Carlo based clustering solution. In this way, no sacrifices would have to be made for speed or accuracy—one could simply use the system most practical for a particular analysis task.

As an aside, the NWPCA technique discussed could likely be improved further using an idea already partially implemented in our MATLAB spike sorting framework. From within the framework is currently possible for a user to manually select a subset of datapoints (such as a single cluster) present in a projection and run NWPCA (or any other representational algorithm) again on that selected subset of data. Based on evaluating this technique manually, running a representational algorithm on subsets is often especially effective at isolating poorly isolated clusters. This technique is particularly useful when a single cluster is extremely far away the other clusters. In such a circumstance, the remaining clusters are often compacted together tightly—making clustering difficult. By running the representational algorithm a second time on the poorly isolated clusters, better separation can be seen between those datapoints (Sahani, 1999). It is likely that this technique could be automated in a straightforward fashion—perhaps by using the "best likelihood" statistic present in Cell-Finder to determine which clusters are poorly isolated and automatically running NWPCA a second time. This technique could even be integrated into the current analysis framework: each time data is re-represented in space, it could simply be treated as a new "virtual" electrode in the dataset. In this way, the current file format and analysis conventions could be retained.

Additionally, the NWPCA algorithm could be utilized in concert with the union analysis technique in order to compensate for current deficits in the clustering algorithm. However, over the longer term, the threshold and clustering levels of the spike sorting system clearly must be improved using new techniques. For instance, many well studied clustering algorithms, such as spectral clustering (Ng et al., 2001), may yield better results than the current system. Alternatively, multidimensional scaling techniques

similar to Kohonen's SOM, such as Isomap (Tenenbaum et al., 2000), could be used to represent the dataset with far fewer points but still maintain the topological structure of the data. If such a technique were applied, clustering algorithms would need to classify far fewer points in order to obtain the same results found currently. Consequently, the problem of clustering would be greatly simplified and even simple algorithms might work better than the current approach.

As I intend to continue this work for many more months, I do not see this document as an endpoint for the project. While I hope to see significant near-term gains as a consequence of this work, such as the current use of the NWPCA algorithm, the robust implementation of some of the techniques evaluated here will take some months. Indeed, this process of evaluating various techniques has, in some ways, left me with more questions than answers. The fact that I have found many successful approaches to spike sorting these data means that the process of finding an optimal solution will be rather difficult. To some degree it seems as though there is, in reality, no optimal combination of algorithms to solve this problem—there are tradeoffs inherent in each technique. Perhaps by creating multiple specialized spike sorting systems a closer to ideal solution can be obtained. The coming months will reveal if this is indeed the case.

7. Recent Work: Summer 2007

In the past two months, a great deal of progress has been made towards the goal of creating a more robust spike sorting system. Many new techniques at various levels of the spike sorting system have been implemented, and entire datasets can be analyzed utilizing the new techniques. While the current implementation of these algorithms is still within MATLAB, it is still possible to do useful analyses—albeit slower than in a robust, optimized version of the new spike sorting system. However, until the usefulness of these various algorithms can be rigorously characterized, work towards such a robust implementation cannot be justified. Consequently, a great deal of computation is currently underway to evaluate the efficacy of various techniques that have been found. It is hoped that the results from these analyses will be promising, and that work towards a robust implementation of a new spike sorting procedure can begin in the fall. An updated version of Figure 8 explains the current vision of for a new, robust spike sorting system.

7.1 Nonlinear Dimensionality Reduction Techniques Evaluated

Earlier in this work it was hoped that nonlinear dimensionality reduction techniques, such as Isomap and Locally Linear Embeddings (LLE), might be able to represent our data better than linear techniques such as PCA. Unfortunately, new literature suggests that nonlinear dimensionality reduction techniques rarely outperform PCA on real data (van der Maaten, 2007). This is an interesting finding, but sadly one that conforms to tests using various nonlinear techniques to represent the data.

Many of these techniques were tried, ranging from Isomap and LLE to newer variants of these algorithms—such as Conformal Eigenmaps, a variant of LLE. While the results of these tests will not be explained in detail here, none of these techniques yielded low dimensional projections that were significantly better than standard PCA. This is

especially disappointing considering the massive computational cost of these algorithms compared to PCA. Since NWPCA has already shown to successfully outperform standard PCA, it seems as though it remains the best algorithm for dimensional reduction.

7.2 NWPCA Better Understood

As a consequence of the disappointing results from testing many nonlinear techniques, the nature of NWPCA's success was investigated further. Initially, it was assumed that since the noise events used to sample from the raw data only choose random points far away from spike events, the amplitude of the spikes that comprise the noise covariance matrix would be small (see Section 4.3). Surprisingly, this is not the case. Since the algorithm did not check for superthreshold events on neighbor electrodes, they were sometimes included in the randomly sampled noise events—provided that those spikes were below threshold on the center electrode (such events are hereafter referred to as "neighbor spike" events).

This was somewhat interesting, and prompted a question: what happens if these are removed? The answer is that whitening by a noise covariance matrix that does not contain any of these neighbor spike events generates projections that are very similar to standard PCA. Consequently, it seems as though it is the process of whitening by neighbor spike events that is most useful at reducing noninformative variance. This was confirmed by creating a noise covariance matrix that was comprised solely of neighbor spike events. When PC space is whitened by such a matrix, the result is very similar to standard NWPCA—where the noise covariance matrix contains far fewer neighbor noise events. This result suggests that only a few neighbor noise events are necessary to successfully whiten PC space. Indeed, the original NWPCA procedure is validated as it

makes sense to randomly choose noise events. In this way, the number of neighbor noise events that are chosen is dependent on the structure of the dataset being analyzed.

As a result of this finding, it seemed clear that there may be other means of reducing noninformative variance in PC space, thus improving the representation of the data in a manner similar to NWPCA. Such techniques are discussed in the following section.

7.3 Various Variance Reduction Techniques Evaluated

The spike vectors plotted in PC space are comprised of a spike waveform from seven electrodes: six neighbors and the center electrode. This is done with the assumption that all neighbor electrodes contain useful representations of the spike detected on the center electrode. Unfortunately, this assumption is usually wrong. Consequently, choosing to use a subset of neighbor electrodes (instead of all six) can yield better projections from PCA as noninformative components that contribute variance are eliminated. While this is a somewhat useful technique, it is nontrivial to automate this technique in a useful manner as each neighbor electrode might vary in its usefulness over time.

One possible solution that was investigated involved normalizing the variance of all of the neighbor electrodes. The idea behind this is that such a technique might be able to reduce or eliminate high amplitude spikes on neighbor electrodes. Such spikes often create extremely well isolated clusters comprised of few points—relegating the remaining spikes to a very small region of PC space. Consequently, the majority of the datapoints on the electrode cannot be represented adequately by PCA. A simple method to evaluate this idea was implemented: if the variance of a spike waveform from a neighbor electrode exceeded a certain threshold (the “normalization threshold”), it was reduced to one.

When this threshold was chosen correctly, improvements to the representation of the data could often be seen. These improvements look qualitatively similar to the result of applying PCA iteratively to successive subsets of a dataset (mentioned in Section 6). Unfortunately, no promising method for automatically determining the normalization threshold could be determined as the optimal threshold varies from electrode to electrode. As a result, this idea was not pursued much further. Nevertheless, there is an important lesson to be learned: the format of the data sent to be analyzed by the dimensionality reduction algorithm is as important as the algorithm itself. Thus, it makes a great deal of sense to consider new thresholding methods.

7.4 Ellipsoidal Thresholding is A Promising Technique

Currently, the standard spike sorting system only thresholds on the center electrode, that is it looks for events on electrode x(t) that have a greater than τ variance (and less than τ variance x(t-1)). Since our recordings take place on a multielectrode array this makes little sense. Using information from the neighbor electrodes can inform a better thresholding surface. Such a procedure, known as ellipsoidal thresholding, has been proposed (Sahani, 1999).

This procedure involves computing $\Sigma(x_i(t))^2$ where i = 1 to number of neighbor electrodes + the center electrode. A threshold can then be applied to this value rather than to the spike trace of a single electrode. This technique is optimized by whitening the covariance matrix of the raw data prior to computing the power. While work using this technique is very preliminary, results seem promising and will likely result in the detection of many spikes that were previously indistinguishable from noise events.

7.5 Monte Carlo Markov Chain Clustering Methods are Successful

While the current clustering algorithm used (Expectation Maximization used to fit a mixture of Gaussians) is rather fast and somewhat effective, it is far from ideal. The primary reason for this is that if it is initialized poorly (i.e. with the incorrect number of clusters, and with the initial cluster centroids near local maxima), the algorithm will likely converge on local maxima and generate inadequate results. One method of resolving this involves the use of Monte Carlo Markov Chain (MCMC) techniques. Such a method, utilizes a stochastic process to estimate the correct number of clusters for a given probability density function (such as the spike vectors plotted in PC space) (Nguyen et al., 2003). The results from applying such an algorithm, provided by Frank Wood[16] have been promising and generate much better results than the standard clustering algorithm. Some preliminary results can be seen in Figure 9.

7.6 Future Goals and Conclusion

As a consequence of the research done this summer, we stand a great deal closer to the implementation of an entirely new spike sorting system. While work must still be done to improve thresholding, NWPCA and Monte Carlo clustering are already in the process of being tested. Consequently, it is predicted that work towards constructing an optimized, robust version of the new spike sorting system that currently exists in MATLAB will begin soon.

[16] Implementation in MATLAB publicly available from http://www.gatsby.ucl.ac.uk/~fwood/code/crp.zip

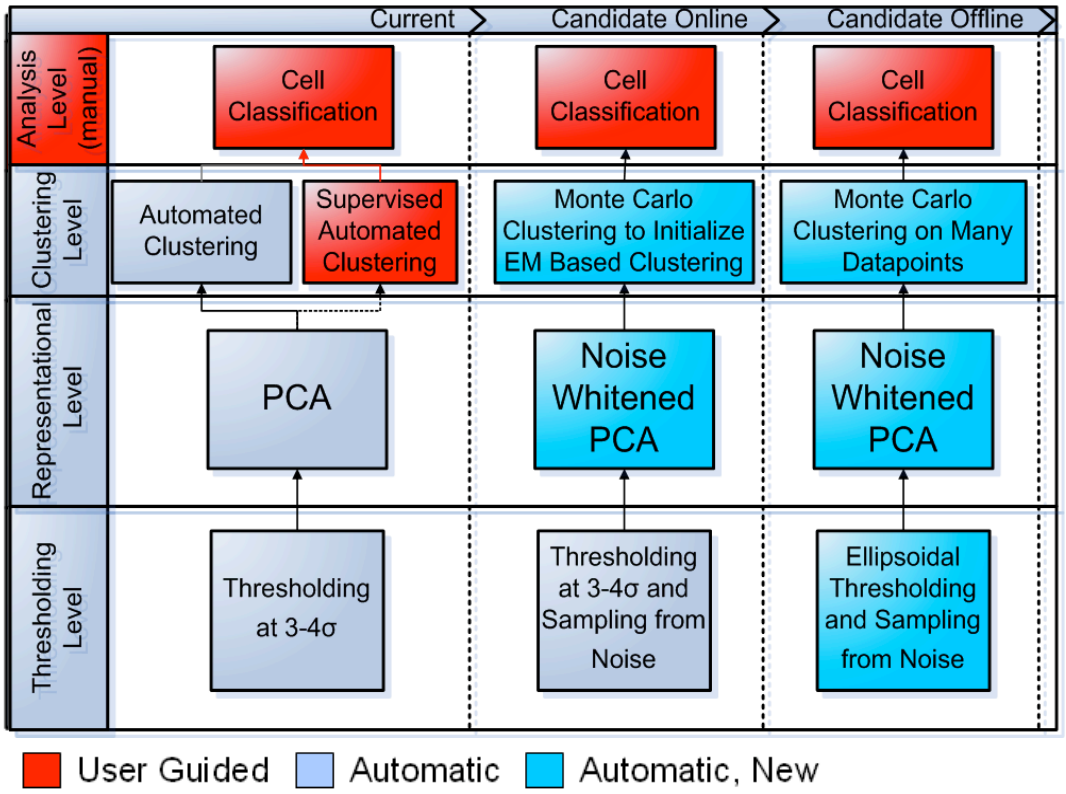

Figure 8 – Current spike sorting system alongside proposed new spike sorting systems: a fast online system, and an offline system designed to maximize cell yield.

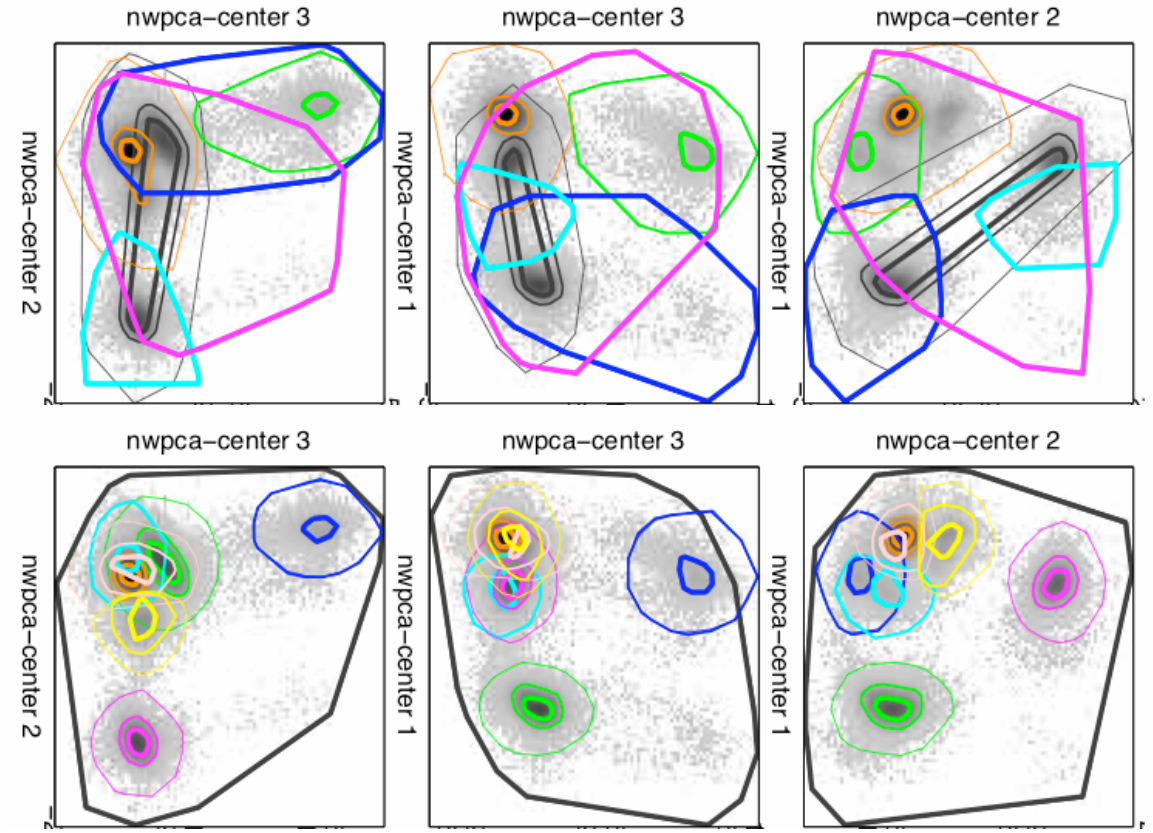

Figure 9 – NWPCA projections with contour plots of cluster boundaries shown. Top: Clustering generated by the Expectation Maximixation (EM) algorithm fitting a mixture of Gaussians to the data. It was initialized by a parametric greedy algorithm. The cluster boundaries do not match regions of density in the plots well. Bottom: Clustering generated by the Monte Carlo Markov Chain clustering algorithm used to initialize the EM algorithm.

7. Acknowledgements

I would like to thank Professor Chichilnisky for his guidance and support throughout my project. Additionally, I thank Jonathon Shlens for his assistance and insights. Jeffery Gauthier was immensely helpful to me when I began my work comparing algorithms in MATLAB—especially with visualizing the results from various techniques. Conversations and assistance from Clare Hulse, Martin Greshner and the rest of SNL-E also helped me to complete this work. Finally, I would like to thank Professor de Sa and Professor Chiba for their advice and help.

References:

Brown GD, Yamada S, Sejnowski TJ (2001) Independent component analysis at the neural cocktail party. Trends Neurosci 24:54-63.

Chichilnisky EJ (2001) A simple white noise analysis of neuronal light responses. Network 12:199-213.

Chichilnisky EJ, Kalmar RS (2002) Functional asymmetries in ON and OFF ganglion cells of primate retina. J Neurosci 22:2737-2747.

Clark A (2001) Mindware : an introduction to the philosophy of cognitive science. New York: Oxford University Press.

Dacey DM, Packer OS (2003) Colour coding in the primate retina: diverse cell types and cone-specific circuitry. Curr Opin Neurobiol 13:421-427.

Harris KD, Henze DA, Csicsvari J, Hirase H, Buzsaki G (2000) Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. J Neurophysiol 84:401-414.

Hermle T, Schwarz C, Bogdan M (2004) Employing ICA and SOM for spike sorting of multielectrode recordings from CNS. J Physiol Paris 98:349-356.

Himberg J, Hyvarinen A, Esposito F (2004) Validating the independent components of neuroimaging time series via clustering and visualization. Neuroimage 22:1214-1222.

Hinton GE, Sejnowski TJ (1999) Unsupervised learning : foundations of neural computation. Cambridge, Mass.: MIT Press.

Hubel DH, Wiesel TN (1968) Receptive fields and functional architecture of monkey striate cortex. J Physiol 195:215-243.

Koch C (2004) The quest for consciousness : a neurobiological approach. Denver, Colo.: Roberts and Co.

Litke AM, Bezayiff N, Chichilnisky EJ, Cunningham W, Dabrowski W, Grillo AA, Grivich M, Grybos P, Hottowy P, Kachiguine S, Kalmar RS, Mathieson K, Petrusca D, Rahman M, Sher A (2004) What does the eye tell the brain?: Development of a system for the large-scale recording of retinal output activity. Nuclear Science, IEEE Transactions on 51:1434-1440.

Ng AY, Jordan MI, Weiss Y (2001) On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems.

Nguyen DP, Frank LM, Brown EN (2003) An application of reversible-jump Markov chain Monte Carlo to spike classification of multi-unit extracellular recordings. Network 14:61-82.

Rebrik SP, Wright BD, Emondi AA, Miller KD (1999) Cross channel correlations in tetrode recordings: implications for spike-sorting. In: Proceedings of Computation and Neural Systems Meeting. Big Sky, MT.

Reperant J, Ward R, Miceli D, Rio JP, Medina M, Kenigfest NB, Vesselkin NP (2006) The centrifugal visual system of vertebrates: a comparative analysis of its functional anatomical organization. Brain Res Rev 52:1-57.

Rieke F (1997) Spikes : exploring the neural code. Cambridge, Mass.: MIT Press.

Ripley BD (1996) Pattern recognition and neural networks. Cambridge ; New York: Cambridge University Press.

Sahani M (1999) Latent variable models for neural data analysis. Pasadena, Calif.: California Institute of Technology.

Segev R, Goodhouse J, Puchalla J, Berry MJ, 2nd (2004) Recording spikes from a large fraction of the ganglion cells in a retinal patch. Nat Neurosci 7:1154-1161.

Shlens J, Field GD, Gauthier JL, Grivich MI, Petrusca D, Sher A, Litke AM, Chichilnisky EJ (2006) The structure of multi-neuron firing patterns in primate retina. J Neurosci 26:8254-8266.

Tenenbaum JB, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290:2319-2323.

van der Maaten LP, EO. van den Herik, HJ. (2007) Dimensionality Reduction: A Comparative Review. In.

von der Malsburg C, Schneider W (1986) A neural cocktail-party processor. Biol Cybern 54:29-40.