MEng in Biomedical Engineering – EEE Pathway
3rd Year MEng Group Project
Supervisor: Dr. Patrick Degenaar

A Revolutionary System for Talking to Neurons with Light!

Interim Report

12th January 2009

Submitted By:
Zheng Da Clinton Goh
CID 00472780

Contents

1. Project Specification

We aim to develop a system which allows users to stimulate neurons with light. This is done through in-vitro optical stimulation of Channelrhodopsin-2 (ChR2) transfected neural cells [1, 2] cultured on a Multi-Electrode Array (MEA). Optical stimulation is carried out by a micro-LED array which is capable of targeting single neurons in a neural network [3, 4]. Our system will provide us with a new and novel way to communicate to neurons with light.

One of our objectives is to develop a spike sorting algorithm which achieves real time response recognition of stimulated neurons to known stimuli of our system. The spike sorting algorithm must be able to detect action potentials and accurately classify detected spikes across all electrodes of the MEA. The algorithm must also be able to provide the user with spatiotemporal information of excited neurons. By applying spike sorting to our system, we hope to develop a method for researchers to gain greater insight into neural interactions of a neural network.

2. Introduction: What is Spike Sorting?

Analysis of spike data recorded by a microelectrode is a technically challenging process. Besides picking up background noise, electrodes also record spikes from more than one neuron simultaneously. This poses a challenge of assigning spikes to neurons. Spike sorting addresses this problem by implementing a three step algorithm to classify spikes recorded on a multi-electrode array (MEA). This process is illustrated in Fig 1.

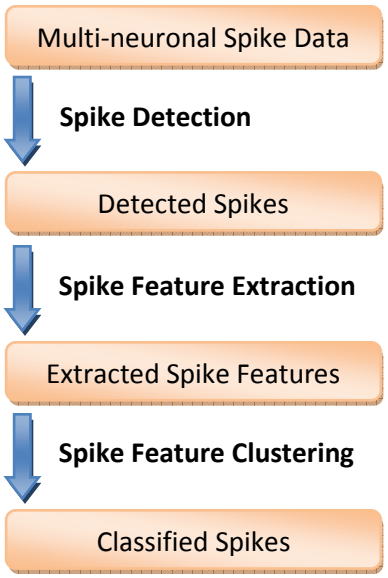


Fig 1: Spike Sorting Algorithm

The first step of spike detection involves filtering recorded spike data to remove high frequency noise and preserve low frequency action potentials. Spikes are then detected using a voltage threshold or a slope threshold on the signal recording.

The next step involves the extraction of spike features from the set of detected spikes. This can be done using various methods which include feature analysis, Principle Components Analysis (PCA) and Independent Components Analysis (ICA).

The final step clusters spikes features and groups detected spikes into various classes. This can be done using various clustering algorithms, such as k-means clustering or Bayesian clustering. The final result of this basic three step algorithm is the classification of spikes according to their waveform features. We can then assign spikes and identify spike patterns of specific neurons. Recording neuron activity using multiple electrodes will also allow us to estimate the location of spiking neurons in the neural network. The spatiotemporal information obtained can be used for many applications which

include transferring an image from the micro-LED array to a neural cell culture, and the identification of neural interactions within a neural network.

3 Spike Detection

The aim of spike detection is to extract data points which form an action potential, based on a set of user defined criteria. Spike detection methods include the voltage threshold method which uses a minimum amplitude threshold to detect action potentials, and the slope threshold method which detects spikes according to its slope, in addition to its amplitude. Before spikes can be detected, each recording is filtered to remove high frequency background noise and preserve low frequency spikes.

3.1 Filtering Multi-neuronal Spike Data

In addition to action potentials detected on an MEA, electrodes also pick up high frequency noise. In order to remove unwanted noise for spike detection, we first pass signal recordings through a band-pass filter. There are two types of digital filters which can be used for this purpose: the Finite Impulse Response (FIR) and the Infinite Impulse Response (IIR) filter. For our spike sorting algorithm, we have implemented an IIR, 2^{nd} order band-pass elliptical filter in MATLAB.

Although IIR filters are not as stable as FIR filters and they introduce a non-linear phase delay, IIR filters require a small number of taps and have a lower computational load, ensuring that the spike sorting algorithm remains computationally efficient. This facilitates future work in moving from offline to online spike sorting. We have also chosen the elliptic filter as it has a fast transition in gain between the pass-band and stop-band, effectively filtering out unwanted frequencies.

As with all filters, signal distortion may occur especially if signal frequencies lie near the cut-off frequencies. This may result in shifted spike times and spike waveform deformation. By setting fixed high and low pass cut-off frequencies across all channels, we ensure constant amplitude and phase errors, hence preserving essential information such as relative spike shapes. At the same time, the user has to carefully choose cut-off frequencies so as not to cause excessive signal distortion. A bandwidth of 300 Hz to 3 kHz is recommended if a broadband amplifier is used [5].

3.2 Voltage Threshold Method

The voltage threshold method detects data points in a signal recording which lie between a minimum and a maximum threshold value. Many spike sorting algorithms use this method for detecting spikes as it is easy to implement and discards background noise effectively [5, 6, 7, 8]. The minimum amplitude threshold discards low amplitude background noise, while the maximum amplitude threshold discards large amplitude artefacts.

Since the MEA records extracellular potentials, a recording of an action potential will show a prominent negative peak. Hence, a negative minimum voltage threshold (Fig 2) is implemented to detect neuronal spikes.

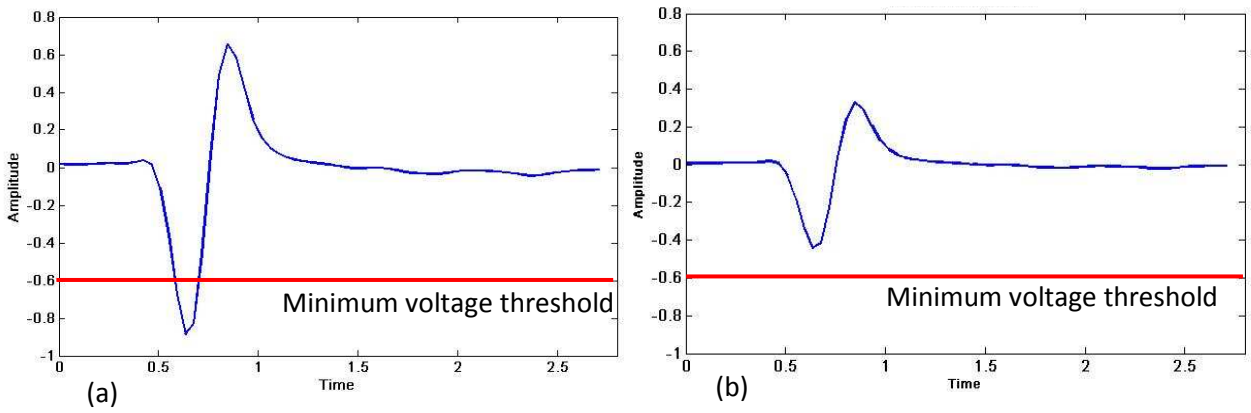


Fig 2: Spike detection using the voltage threshold method. (a) Detected spike with amplitude exceeding minimum voltage threshold. (b) Spike amplitude is too small for detection

3.3 Slope Threshold Method

The slope threshold method (Fig 3) detects action potentials based on the slope of the spike in addition to a minimum amplitude [5]. This method requires the user to define the minimum amplitude, and minimum and maximum slope for which a set of data points must have in order to be detected as a spike. In order to avoid analysing all points in the data set, the minimum amplitude criteria is first applied, followed by the minimum and maximum slope threshold.

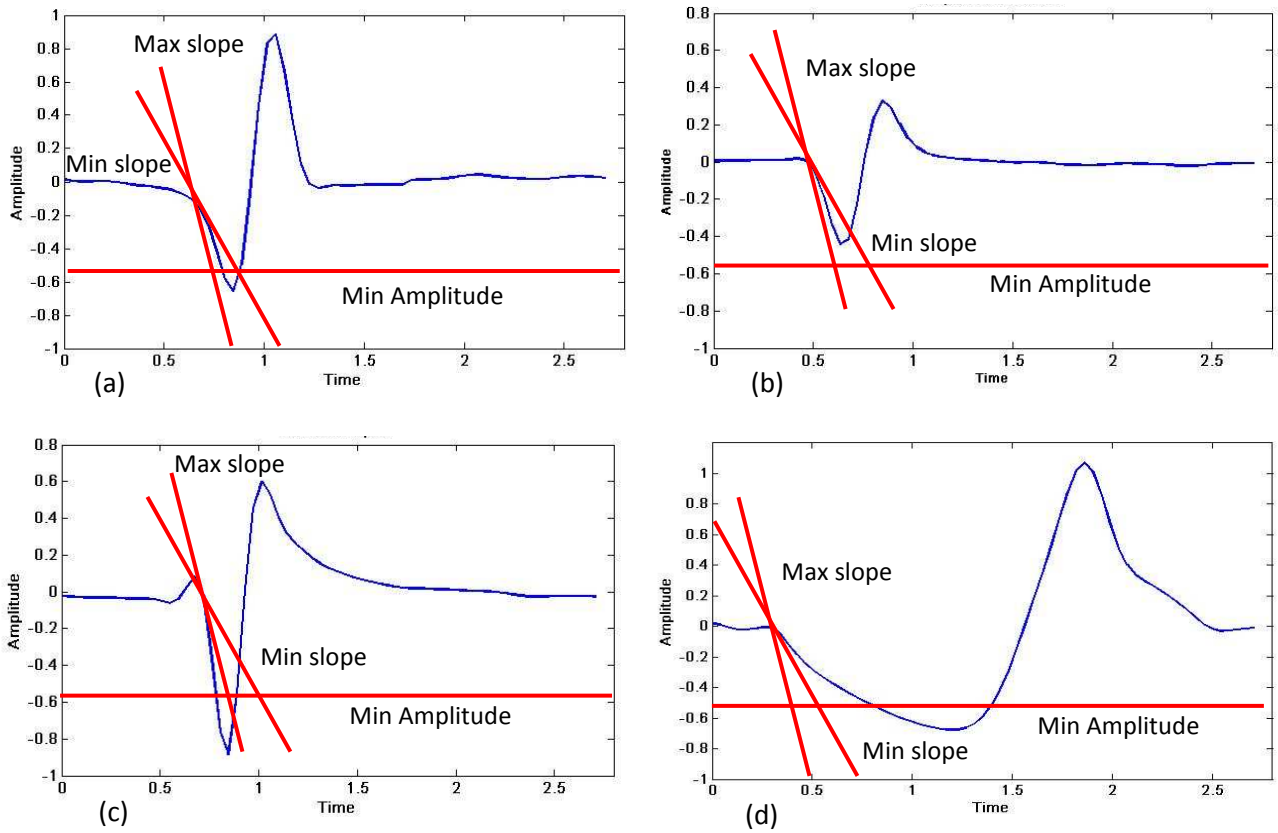


Fig 3: Spike detection using the slope threshold method. (a) Detected spike using the threshold method. (b) Spike amplitude is too small. (c) Spike slope is too high. (d) Spike slope is too low.

3.4 Comparison of Spike Detection Methods

In order to detect action potentials only, the minimum amplitude set using the voltage threshold method must be large enough so as to discard any background noise. This method assumes that the amplitude of recorded action potentials is much greater than that of the background noise level. This is true if the spiking neuron is located near the electrode. However, the amplitude of action potentials decreases exponentially as a function of the distance from the recording electrode to the spiking neuron [8]. If the stimulated neuron is located further away from the electrode, then low amplitude spikes may not be detected by the voltage threshold method. One way to solve this problem is through the use of multiple electrodes. Each time a spike is detected on a single electrode, we can visualise recorded waveforms across all channels of the MEA. This ensures that information is not lost even though spikes are not detected on other electrodes. Another solution is provided by the slope threshold method which adds an extra dimension to the detection criteria of spikes.

The slope threshold method allows greater flexibility in setting the minimum amplitude threshold as it uses the added criteria of minimum and maximum slope. The minimum amplitude can now be set to a lower value, resulting in more data points being detected. At the same time, slope threshold values allow the user to discard background noise and detect spikes of a certain shape only. However, the user has to have prior knowledge of spike shapes in order to specify appropriate minimum and maximum slope threshold values. In the case where such prior knowledge is unavailable, inappropriate threshold values may result in the loss of important spike information.

3.5 Algorithm Implementation and Preliminary Results

We have chosen to adopt the voltage threshold method for our spike detection process. Although low amplitude spikes may not be detected on some electrodes, the use of an MEA ensures that electrodes which are close to the spiking neuron will record large amplitude spikes. The voltage threshold algorithm is also easy to implement and computationally less costly than the slope threshold method. On the other hand, we have insufficient information on the spike shapes of ChR2 transfected SH-SY5Y cells currently used in our system. This means that we are unable to determine appropriate slope thresholds for the detection of spikes.

The voltage threshold method was implemented in MATLAB for a single electrode recording which can be extended to each channel of the MEA. In order to test the implemented algorithm, we applied our voltage threshold algorithm to Quiroga's test data [6]. Fig 4 shows the results of aligned and stored spikes obtained from spike detection.

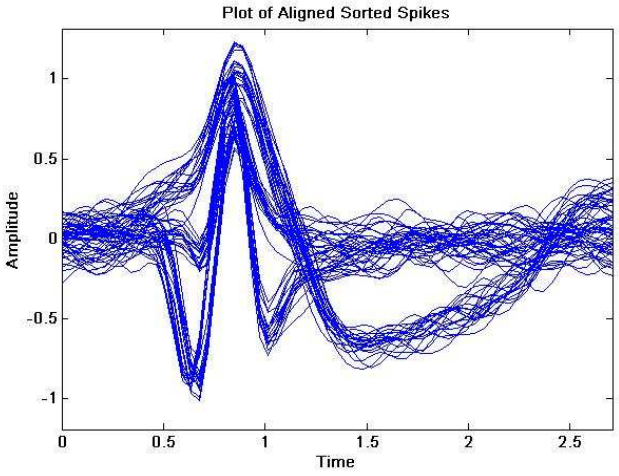


Fig 4: Plot of aligned and detected spikes stored for spike feature extraction

4. Spike Feature Extraction

Neurons typically produce action potentials with a characteristic shape [9]. If we assume that the neuron's spike shape is time invariant, we can characterise the spike shape of each neuron by extracting spike features. However, this assumption may not hold as neuron adaptation and evolution of spike shapes may occur during the stimulation period. Overlapping spikes may also be recorded when two or more action potentials reach an electrode at the same time. Various spike feature extraction algorithms deal with these problems differently, and will be discussed in this section.

4.1 Feature Analysis

A simple method of analysing spike shapes is through the direct measurement of certain spike features. These features illustrated in Fig 5, include the peak-to-peak voltage amplitude, spike width, and positive and negative spike gradient.

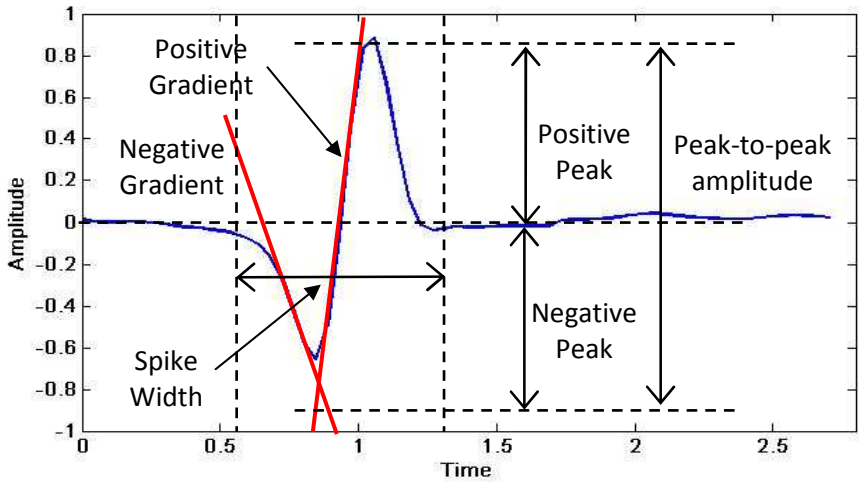


Fig 5: Measurement of spike features. Useful spike features which characterise a spike shape include positive and negative gradients, spike width, positive and negative peaks, and peak-to-peak amplitude.

Measured spike features can be used to discriminate between spikes from different neurons. This method may be used on a single electrode or across multiple electrodes. Fig 6 shows a scatter plot of positive peak amplitude against negative peak amplitude of spikes detected on a single electrode. There is a clear clustering of three different spike shapes which implies the recording of spike patterns from three different neurons. When working with multiple recordings from an MEA, spike features across neighbouring electrodes may be compared to identify spikes from a single neuron being recorded on different electrodes.

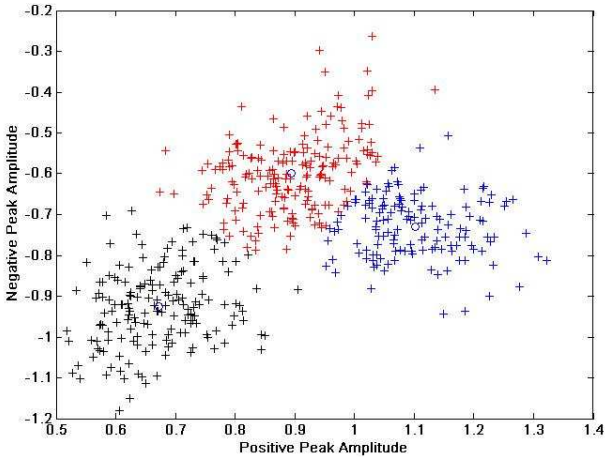


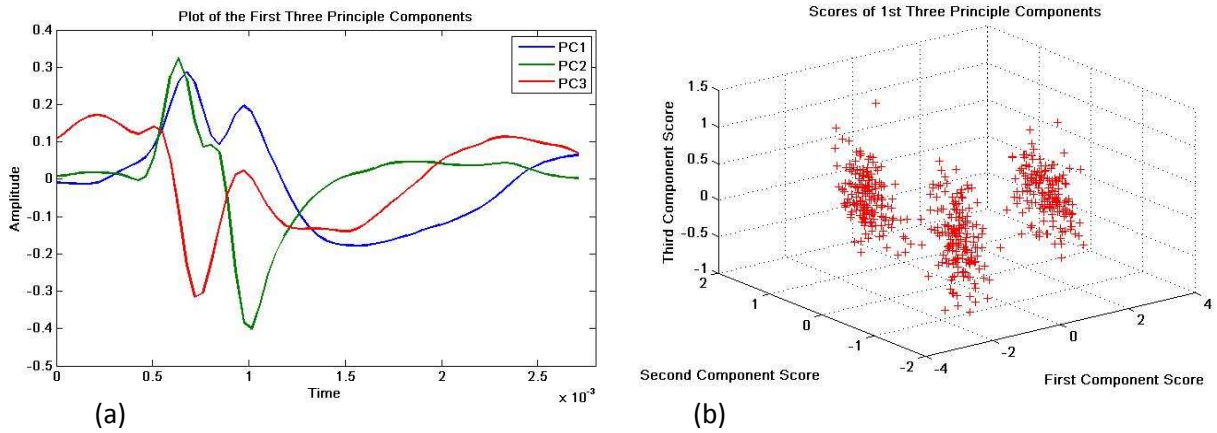
Fig 6: Feature analysis applied to a single electrode. 3 clusters are observed, each corresponding to a different neuron.

4.2 Principle Components Analysis

Principle Components Analysis (PCA) is commonly used in spike sorting algorithms to automatically extract spike features for spike classification [7, 9]. The aim of PCA is to compute an ordered set of orthogonal basis vectors which can be linearly combined to describe each detected spike fully. This method is based on the assumption that the largest variation in a set of data contains the dynamics of interest [9, 10]. A brief mathematical description of PCA can be found in Appendix A. Spike features are captured in the score for each principle component, which is determined by:

$$s_i = \sum_t c_i(t)\,x(t) \qquad \text{Equation 1}$$

where $x(t)$ is the detected spike, and $c_i(t)$ is the i^{th} principle component. In practice, scores of the first three components are used to identify clusters as they account for approximately 76% of the variations in the data. Subsequent components may represent variability in the data due to background noise [9]. The results of PCA are illustrated in Fig 7.



(a) (b)

Fig 7: Results of PCA applied on detected spikes. (a) The first three principle components. (b) A scatter plot of the scores from the first three components.

4.3 Independent Components Analysis

Independent Components Analysis (ICA) is a method which is commonly used to deal with spike data recorded by multiple electrodes [9, 11, 12, 13]. Like PCA, it is based on a set of assumptions and uses data transformation to identify independent sources of activity in recorded signal mixtures. ICA is similar to blind source separation: it recovers n independent signals that have been mixed into n channels by an unknown mixing process. The ICA concept is illustrated in Fig 8 and a brief mathematical description of ICA can be found in Appendix B.

The ICA algorithm may be implemented via maximum likelihood estimation, entropy maximisation or maximisation of non-Gaussianity [14]. An efficient and widely used ICA algorithm is the FastICA algorithm developed by Hyvärinen which utilises non-Gaussianity maximisation [15]. In order to illustrate the capabilities of ICA, a set of sinusoidal and sawtooth signal sources were generated and

linearly mixed in MATLAB. FastICA was then applied to these mixtures and the independent components were obtained, accurately estimating signal sources up to the multiplicative sign (Fig 9).

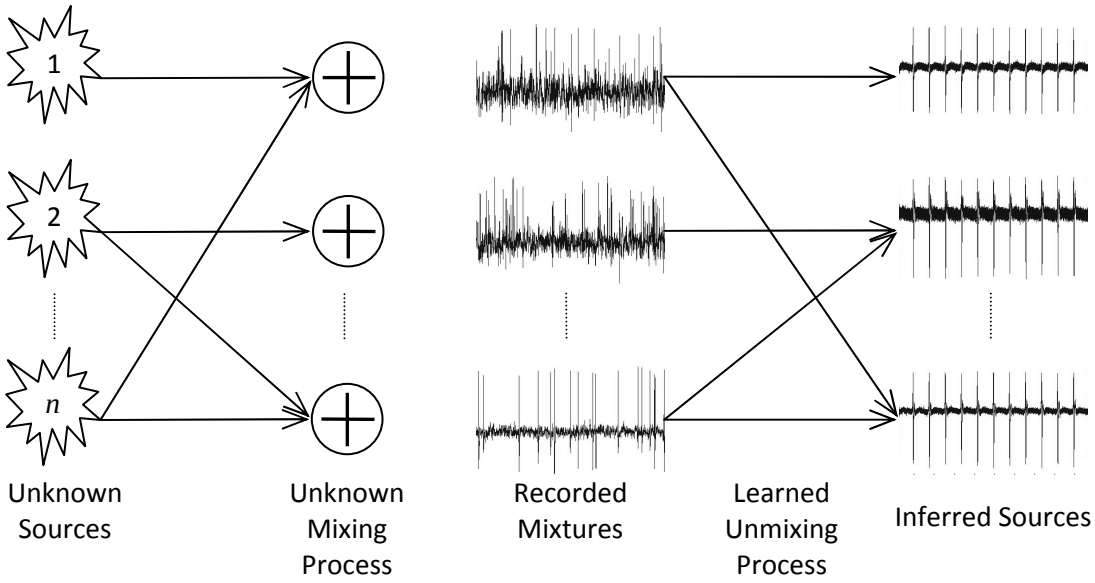


Fig 8: A model of Independent Components Analysis. *n* unknown sources or spiking neurons are mixed linearly by an unknown mixing process and are recorded on *n* electrodes of the MEA. The unmixing process is found through ICA which transforms the recorded mixtures into independent signals

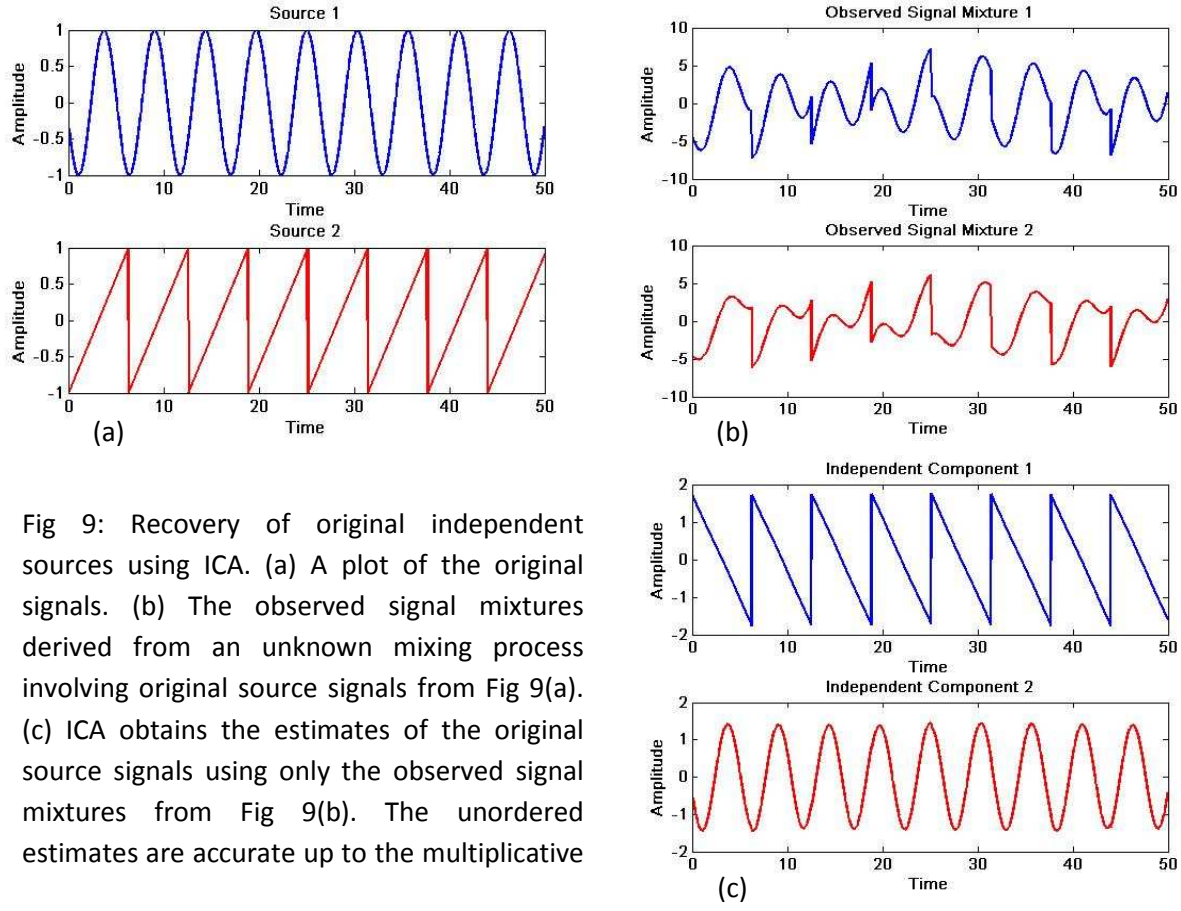


Fig 9: Recovery of original independent sources using ICA. (a) A plot of the original signals. (b) The observed signal mixtures derived from an unknown mixing process involving original source signals from Fig 9(a). (c) ICA obtains the estimates of the original source signals using only the observed signal mixtures from Fig 9(b). The unordered estimates are accurate up to the multiplicative

Unlike other spike feature extraction algorithms, ICA automatically solves the problem of overlapping spikes since it assumes that observations are linear mixtures of source signals. However, this powerful method is not free from constraints. Two other assumptions made by ICA are that the number of sources must be equal to, or less than the number of observed signal mixtures, and that the mixing process is instantaneous. The first assumption is violated when using a single electrode, but may produce results which contain more information than other algorithms such as PCA when applied to MEA recordings. The second assumption may be violated when a time delay is introduced in spike patterns from a neuron located far away from the electrode. Hence, ICA has to be treated with caution in order to make full use of its effects on raw data.

4.4 Comparison of Spike Feature Extraction Methods

Of all the above methods, ICA appears to address our problem very well: ICA allows us to recover the original spiking patterns of individual neurons from a mixture of signals observed on recording electrodes on the MEA. To obtain spatial information from the independent components (ICs), the normalised scalar product between the electrode signals and the ICs can be computed, and its magnitude used to estimate the neuron's position with respect to the recording electrode [12].

Although promising, ICA is flawed as the number of neurons has to be equal or less than the number of electrodes. Although methods which deal with overcomplete ICA problems have been developed [13, 14], few of them have been tested and verified to be capable of accurately recovering original spike patterns fully. Another assumption which ICA makes is the instantaneous mixture of source signals. This may not be true as spikes from neurons which are far away from the recording electrode may experience a time delay during signal recording. Lastly, our system does not necessarily require a blind source separation technique like ICA to carry out spike sorting. This is due to the fact that our system may be calibrated by stimulating single neurons. This provides us with prior knowledge of the system's calibrated neurons, turning the problem into that of partial blind source separation.

PCA on the other hand, is an effective tool for discriminating spike shapes recorded on a single electrode or from each channel in an MEA. One constraint of PCA is that it requires spikes to be aligned, disregarding any temporal information such as spike times and time delays. This problem is solved by recording spike times during spike detection, and assigning spike classes to these times. Besides this, PCA does not solve the problem of overlapping spikes as overlapped spike shapes will be incorrectly clustered or discarded as outliers during the clustering process. PCA also fails to recognise spike adaptation and spike shape evolution where the waveforms are not stationary. In such cases, PCA may produce two different clusters for the same neuron.

PCA also fails when extracting spike features across multiple electrodes. Since spike amplitudes are dependent on the distance between the neuron and electrode [8], spikes from the same neuron which are recorded by two or more electrodes may exhibit different spike shapes. Applying PCA to such data may result in the formation of two or more clusters (Fig 10) and the generation of false positives in our system. Hence, PCA alone remains only useful for spike sorting on a single channel.

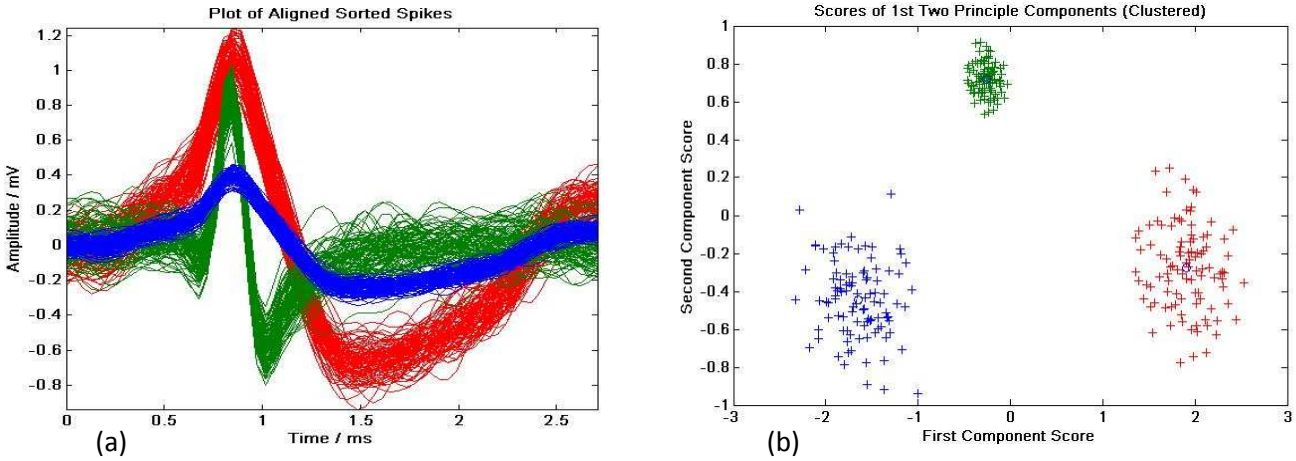


Fig 10: Generation of false positives due to changing spike shape. (a) A plot of three spike waveforms, two of which (blue and red) come from the same neuron. The blue and green spikes are recorded by an electrode nearer to the neuron while the red spikes are recorded by one further away. (b) A plot of the first two principle components obtained from the aligned spikes shows three distinct clusters. This implies that the blue and red spikes come from two different neurons, resulting in the generation of false positives.

PCA however, can be combined with feature analysis to provide an added dimension of clustering across neighbouring electrodes. Although spike amplitude is dependent on the distance between the spiking neuron and recording electrode [8], there are certain features which remain independent of the neuron's spatial location within the MEA. These features include the peak-to-peak amplitude ratio and spike width, which we may use to re-classify clustered spikes detected across multiple electrodes.

The above method is not free from the limitations of PCA. It is unable to properly classify overlapping spikes and may give false positives when presented with time varying spike shapes. Despite these shortcomings, this method gives us greater insight into the neuron's position as we can now visualise spikes from the same neuron which appear on multiple channels. Based on the spike amplitudes recorded by neighbouring electrodes, we can identify the neuron's spatial location with respect to recording electrodes on the MEA. An example describing the variation of spike shapes on neighbouring electrodes is shown in Fig 11.

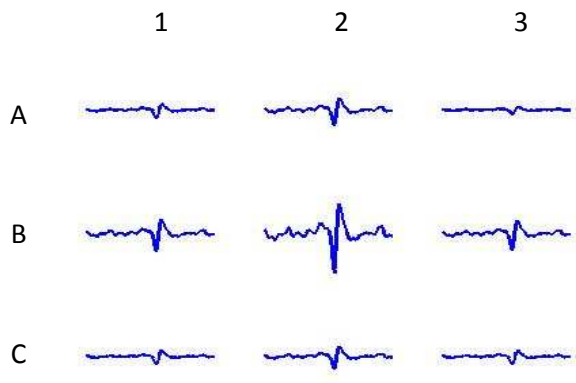


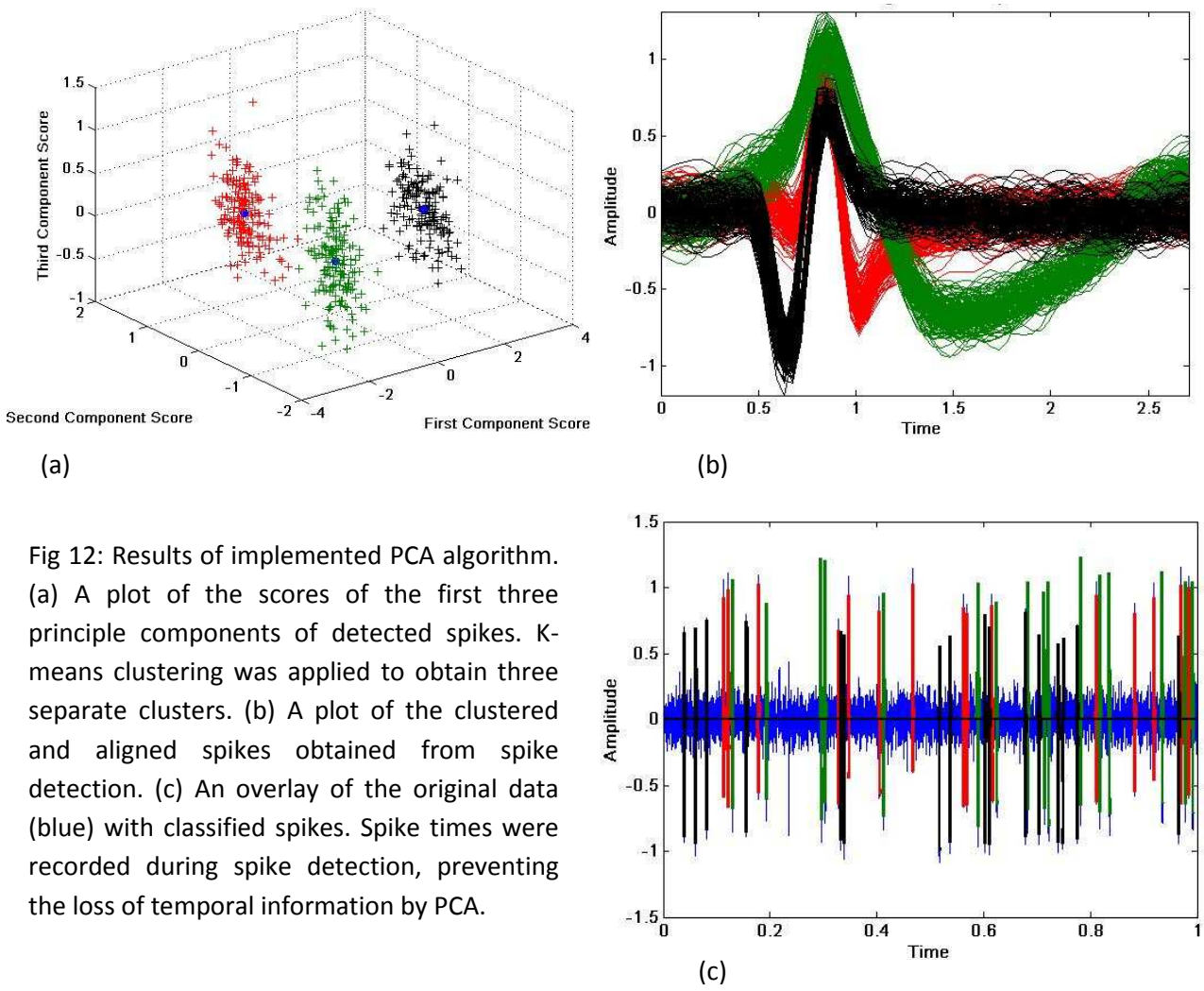
Fig 11: Spike waveforms on a 3x3 electrode array. The spiking neuron is located nearest to electrode B2 which shows the greatest spike amplitude compared to the remaining channels.

4.5 Algorithm Implementation and Preliminary Results

We have adopted the method which combines PCA and feature analysis for spike feature extraction across multiple channels as described above. Although PCA suffers from several drawbacks, it is easy to implement and remains one of the most commonly used methods in spike sorting. When

combined with feature analysis, we will be able to extract relative peak amplitudes across all channels. By modelling spike amplitude and phase difference as a function of neuron to electrode distance, we can estimate the neuron's position within the neural network. This method clearly achieves our aim of extracting spatiotemporal information from neural activity recorded on a MEA.

Our spike feature extraction algorithm was implemented in MATLAB, and works by first applying PCA to each MEA channel. This allows us to run clustering algorithms on the principle component scores, and to discriminate spike shapes being recorded on each channel. The results of single channel PCA with the use of k-means clustering applied to Quiroga's test data [6] are illustrated in Fig 12.



(a) (b)

(c)

Fig 12: Results of implemented PCA algorithm. (a) A plot of the scores of the first three principle components of detected spikes. K-means clustering was applied to obtain three separate clusters. (b) A plot of the clustered and aligned spikes obtained from spike detection. (c) An overlay of the original data (blue) with classified spikes. Spike times were recorded during spike detection, preventing the loss of temporal information by PCA.

As part of future work, we plan to implement the remaining parts of the algorithm as described in the following paragraphs. The next part of the algorithm involves the extraction of spikes from each cluster on a selected electrode, and comparing the spike features to those of another cluster on a neighbouring electrode. If this comparison produces a single cluster, then we can conclude that the spikes originate from the same neuron. After re-classification of spikes with neighbouring electrodes, we can proceed to extract spatial information of the neuron for a given spike class.

Extracting spatial information is done by visualising and comparing relative spike amplitudes across all channels in the MEA. First, we locate the primary electrode which records the largest peak-to-peak amplitude for a given spike class. This allows us to identify the electrode which is nearest to the neuron of interest. If we assume that spike recordings contain a phase delay which is a function of the distance between the spiking neuron and recording electrode, then we can find the timing differences between spikes recorded on all other channels with respect to the primary electrode.

The next step involves identifying all spike times for the particular spike class on the primary electrode. We then cut out a user defined time window of signal recordings across all channels for each of these spike times. The average waveforms across all channels are calculated and waveforms which differ from the average by more than two times the standard deviation are discarded [8]. We can then use the residual average waveforms, fit them to an exponential model based on the electrode positions, and estimate the neuron's position within the MEA.

5. Spike Feature Clustering

By plotting spike features, we are able to visually identify clusters which correspond to spikes from a particular neuron. However, to manually assign each spike to its cluster is a very time consuming process. There are several algorithms which perform this task automatically with varying amounts of supervision by the user. We will be discussing some of these algorithms in this section.

5.1 K-means Clustering

K-means clustering is an algorithm which classifies data into k number of clusters, based on a set of features common to each data element. In this case, we may use principle component scores, or spike features such as peak-to-peak amplitude ratio and spike width for our clustering process.

After the user has specified the number of clusters, the algorithm uses an iterative process which groups spike features based on the minimum Euclidean distance from each data element to the cluster centroids. The aim is to minimise the total variance within each cluster. The results of k-means clustering can be seen in Fig 12(a) where spikes have been grouped into three clusters based on the scores of the first three principle components.

5.2 Bayesian Clustering

The powerful statistical method of Bayesian analysis can be extended to the clustering problem. In this case, each cluster is modelled with a multivariate Gaussian, centred on the cluster [9]. Clustering is done by calculating the probability that a data point belongs to each of the clusters and this is achieved by using Bayes' Theorem:

$$p\left(c_k \mid x, \theta_{1:K}\right) = \frac{p\left(x \mid c_k, \theta_k\right) p\left(c_k\right)}{\sum_k p\left(x \mid c_k, \theta_k\right) p\left(c_k\right)} \qquad \text{Equation 2}$$

where x is the data and c_k is a particular class described by its parameters $\theta_k = \{\mu_k, \Sigma_k\}$; μ_k and Σ_k are the mean and covariance matrix for the class respectively. The posterior $p(c_k \mid x, \theta_{1:K})$ is the conditional probability of each of the clusters, given the data and the set of parameters for all clusters. The likelihood $p(x \mid c_k, \theta_k)$ is the conditional probability of the data given a particular class, while the prior $p(c_k)$ is the probability of the k^{th} class, which corresponds to the relative firing frequencies of spikes between classes [9]. The evidence $\sum_k p(x \mid c_k, \theta_k)\, p(c_k)$ is the probability of the data, found by marginalising the likelihood over all classes. Class parameters are determined by maximising the likelihood of the data:

$$L(\theta) = \prod_{i=1}^{N} p(x_i \mid c_k, \theta_{1:K}) \qquad \text{Equation 3}$$

This clustering method uses probability to define cluster boundaries, hence allowing us to quantify the certainty of the classification. The AutoClass package [16] implements the above algorithm for unsupervised clustering tasks and is capable of choosing the number of classes automatically, which is not possible with the k-means clustering algorithm.

5.4 Comparison of Spike Feature Clustering Methods

The main challenge of any clustering algorithm lies in choosing the number of clusters. The k-means clustering algorithm requires the user to specify the number of clusters. This is a huge disadvantage as the spike sorting algorithm requires supervision, thus preventing us from implementing it online. At the same time, the number of clusters chosen by the user affects the outcome of the clustering process greatly. A poor choice of cluster numbers may lead to incorrect spike clusters. By using the minimum Euclidean distance as the basic criteria for clustering, the algorithm ignores the distribution of data within the cluster. This results in the failure to recognise overlapping clusters and clusters whose shapes differ from a spherical distribution [9]. In addition, omission of outliers has to be carried out manually as the k-means clustering assumes that all data points belong to a cluster. Despite these drawbacks, this algorithm is easy to implement and has low computational complexity.

Bayesian clustering is superior to k-means clustering as it is able to choose the number of clusters automatically. It is also able to recognise overlapping clusters and can omit outliers automatically by assigning a large background class with low cluster weight [9]. However, this clustering method is difficult to implement and can be computationally costly when dealing with large amounts of data [13]. In order to ensure the efficiency of our spike sorting algorithm, lengthy computation periods may be avoided by running the spike sorting algorithm on shorter segments of signal recordings at the expense of reduced data size.

5.5 Algorithm Implementation and Preliminary Results

Currently, we have implemented a k-means clustering algorithm for offline spike sorting as it is easy to implement on MATLAB. Actual recordings made on our system have yet to be carried out, thus we are unable to assess the performance of this simple clustering algorithm. However, as we move from offline spike sorting to online spike sorting, we must implement a clustering algorithm which can automatically choose clusters and remove outliers without supervision. As part of future work, we aim to implement the AutoClass algorithm in MATLAB for the purpose of spike feature clustering. Another round of assessment will then be carried out to determine the amount of computational time required for AutoClass to run on MATLAB as ANSI C tends to provide a more efficient computation platform. The results of the implemented k-means clustering algorithm are illustrated in Fig 6, 10(b) and 12(a).

6. Testing and Validation

In order to assess the validity of our spike sorting algorithm, we plan to develop a testing platform in MATLAB which can be used to simulate spikes of a neural network. The platform should allow the user to input n number of neurons and to specify each neuron's location within the MEA. Once these parameters have been specified, the testing platform will generate n number of spikes with different spike shapes, each allocated to n neurons.

The user can then choose the desired stimulation pattern. Parameters for stimulation include stimulus time, duration, frequency and location. The testing platform will use these input parameters to generate spike waveforms for each stimulated neuron across all channels using the generated spike shapes. We intend to model the spike amplitude and phase delay as a function of the distance between the excited neuron and recording electrode. Two or more spiking neurons which are simultaneously recorded by a single electrode will result in spike overlap.

The user may also include neural interactions between neuron pairs with the specification of interaction strength. Based on the strength of interaction, the platform will use probabilistic methods to determine if the depolarising wave travelling along the axon of a stimulated neuron will result in the eliciting an action potential in another neuron.

What we will obtain from the platform is a set of simulated waveform observations across all channels of the MEA. We can run our spike sorting algorithm on the set of test data generated by our platform. The estimated locations of the detected neurons, spike times and spike classes are then compared to that generated by the simulator. Errors which result from this comparison of spatiotemporal information can then be assessed and our spike sorting algorithm can be validated.

7. Project Gantt Chart

Week	1	2	3	4	5	6	7	8	9	10	11
Date	6/10/08 to 10/10/08	13/10/08 to 17/10/08	20/10/08 to 24/10/08	27/10/08 to 31/10/08	3/11/08 to 7/11/08	10/11/08 to 14/11/08	17/11/08 to 21/11/08	24/11/08 to 28/11/08	1/12/08 to 5/12/08	8/12/08 to 12/12/08	15/12/08 to 19/12/08
Project Task Description (Autumn Term)	Introduction to Projects and Choosing 3rd Year MEng Group Projects										
	Project Briefing, Background Reading on Project										
	Allocation of Roles, Drafting of Project Plan, Background Reading on Spike Sorting										
	Lab Induction, Spike Sorting Plan										
	Test and Understand Quiroga's Spike Sorting GUI with Test Data and Raw Data										
	Understand and Implement Spike Detection in MATLAB										
	Understand and Implement PCA in MATLAB										
	Understand and Implement k-means Clustering in MATLAB										
	Implement Single Channel Spike Sorting in MATLAB GUI										
	Understand the Concepts of ICA										
	Review of Development of Spike Sorting Algorithm and Preparation of Interim Report										
Week	1	2	3	4	5	6	7	8	9	10	11
Date	12/1/09 to 16/1/09	19/1/09 to 23/1/09	26/1/09 to 30/1/09	2/2/09 to 6/2/09	9/2/09 to 13/2/09	16/2/09 to 20/2/09	23/2/09 to 27/2/09	2/3/09 To 6/3/09	9/3/09 to 13/3/09	16/3/09 to 20/3/09	23/3/09 to 27/3/09
Project Task Description (Spring Term)	Implement PCA with Feature Analysis for Multiple Channel Comparison in MATLAB										
	Implement Automatic Clustering Algorithm in MATLAB										
	Develop Spike Generator for Testing Platform in MATLAB										
	Develop Signal Recording Generator for Testing Platform in MATLAB										
	Validate and Test Spike Sorting with PCA & Feature Analysis										
	Assess Errors and Update Spike Sorting Algorithm										
	Apply Spike Sorting Algorithm for Offline Raw Data										
	Upgrade Algorithm for Online Spike Sorting										
	Perform Online Spike Sorting on Raw Data										
	Review of Implementation of Spike Sorting Algorithm, Preparation of Final Report										

Legend: Completed To be Completed

8. References

1. Kramer RH, Chambers JJ, and Trauner D. Photochemical tools for remote control of ion channels in excitable cells. Nat Chem Biol 1: 360–365, 2005.
2. Zhang F, Wang LP, Boyden ES, Deisseroth K (2006) Channelrhodopsin-2 and optical control of excitable cells. Nat Methods 3:785–792.
3. Nikolic, K.; Grossman, N.; Yan, H.; Drakakis, E.; Toumazou, C.; Degenaar, P., A Non-Invasive Retinal Prosthesis - Testing the Concept, Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE , vol., no., pp.6364-6367, 22-26 Aug. 2007
4. Poher V, Grossman N, Kennedy GT, Nikolic K, Zhang HX, Gong Z et al. Micro-LED arrays: a tool for two-dimensional neuron stimulation. *J Phys D Appl Phys* 2008; **41**: 94014–94022.
5. Multi Channel Systems MCS GmbH, 2008. MC_Rack User Manual.
6. Quiroga, R. Q., Nadasdy, Z., and Ben-Shaul, Y. 2004. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. Neural Comput. 16, 8 (Aug. 2004), 1661-1687.
7. Pu-Ming Zhang; Jin-Yong Wu; Yi Zhou; Pei-Ji Liang; Jing-Qi Yuan, "Spike sorting in multi-channel extracellular recordings of retinas," Neural Networks and Signal Processing, 2003. Proceedings of the 2003 International Conference on , vol.1, no., pp. 712-715 Vol.1, 14-17 Dec. 2003
8. Segev R, Goodhouse J, Puchalla J, Berry MJ (2004) Recording spikes from a large fraction of the ganglion cells in a retinal patch. Nat Neurosci 7:1154–1161.
9. Lewicki, M., 1998. A review of methods for spike sorting: the detection and classification of neural action potentials. Network. Comput. Neural Syst. **9**, pp. R53–R78.
10. Shlens J., 2005. A Tutorial on Pricipal Component Analysis, www.snl.salk.edu/~shlens/.
11. Glen D. Brown, Satoshi Yamada, Terrence J. Sejnowski, Independent component analysis at the neural cocktail party, Trends in Neurosciences, Volume 24, Issue 1, 1 January 2001, Pages 54-63, ISSN 0166-2236.
12. Amir Madany Mamlouk, Hannah Sharp, Kerstin M.L. Menne, Ulrich G. Hofmann, Thomas Martinetz, Unsupervised spike sorting with ICA and its evaluation using GENESIS simulations, Neurocomputing, Volumes 65-66, Computational Neuroscience: Trends in Research 2005, June 2005, Pages 275-282
13. Susumu Takahashi, Yuichiro Anzai, Yoshio Sakurai, A new approach to spike sorting for multi-neuronal activities recorded with a tetrode--how ICA can be practical, Neuroscience Research, Volume 46, Issue 3, July 2003, Pages 265-272, ISSN 0168-0102.
14. Mitianoudis N., "Audio Source Separation using Independent Component Analysis", Queen Mary, University of London, PhD thesis.
15. A. Hyvarinen, E. Oja, Independent component analysis: algorithms and applications, Neural Networks, Volume 13, Issues 4-5, June 2000, Pages 411-430, ISSN 0893-6080.
16. P. Cheeseman and J. Strutz. Bayesian Classification (AutoClass): Theory and Results. In Advances in Knowledge Discovery and Data Mining, AAI/MIT Press, USA, 1996.
17. Jeffrey Erman , Martin Arlitt , Anirban Mahanti, Traffic classification using clustering algorithms, Proceedings of the 2006 SIGCOMM workshop on Mining network data, p.281-286, September 11-15, 2006 Pisa, Italy

Appendix A: A Mathematical Description of PCA

The aim of PCA is to compute an ordered set of orthogonal basis vectors which can be linearly combined to describe each detected spike fully. To find the set of ordered orthogonal basis vectors or principle components from a set of detected and aligned spikes, we have to re-represent the original data set as follows:

$$\mathbf{PX} = \mathbf{Y} \qquad \text{Equation A1}$$

where $\mathbf{P}$ is the set of ordered principle components, $\mathbf{X}$ is the set of detected spikes and $\mathbf{Y}$ is a re-representation of this data set. We then optimise $\mathbf{Y}$ by maximising its variance and minimising its redundancy [tut] by diagonalising its covariance matrix $\mathbf{C_Y}$:

$$\begin{aligned} \mathbf{C_Y} &= \frac{1}{n-1}\mathbf{Y}\mathbf{Y}^T \\ &= \frac{1}{n-1}\mathbf{P}\left(\mathbf{X}\mathbf{X}^T\right)\mathbf{P}^T \end{aligned} \qquad \text{Equation A2}$$

It is now evident from Equation 4 that in order to diagonalise $\mathbf{C_Y}$, matrix $\mathbf{P}$ is chosen to be transpose of the eigenvectors of $\mathbf{X}\mathbf{X}^T$. Hence, the principle components are obtained as a set of eigenvectors of the covariance matrix of $\mathbf{X}$. The scale factor for each principle component can then be determined as follows:

$$s_i = \sum_t c_i(t)\,x(t) \qquad \text{Equation A3}$$

where $x(t)$ is the detected spike, and $c_i(t)$ is the ith principle component.

Appendix B: A Mathematical Description of ICA

The goal of ICA is to recover the original spike trains of individual neurons from a mixture of signals detected by electrodes on the MEA. Each recorded signal mixture can be expressed as a linear combination of original independent signals, and the system of *n* recorded mixtures is described in the following equation:

$$\mathbf{X} = \mathbf{AS} \qquad \text{Equation B1}$$

where each row of $\mathbf{X}$ is a recorded signal mixture on each electrode on the MEA, each row of $\mathbf{S}$ is an original spike train of a neuron, and $\mathbf{A}$ is an unknown mixing matrix which generates $\mathbf{X}$ from $\mathbf{S}$. Similarly, we are able to define the unmixing process where the unmixing matrix $\mathbf{W}$ recovers *n* independent sources from *n* recorded signal mixtures.

$$\mathbf{S} = \mathbf{WX} \qquad \text{Equation B2}$$

In order to estimate both the unknown mixing matrix $\mathbf{A}$ and independent sources $\mathbf{S}$ from the observed mixture $\mathbf{X}$, ICA makes several assumptions. ICA first assumes that *n* recorded signals are a linear mixture of statistically independent components which belong to spike patterns from *n* neurons. If two random variables x_1 and x_2 are statistically independent, then the joint probability density function of both x_1 and x_2 is the product of each of the probability density functions of the random variables:

$$p(x_1, x_2) = p(x_1)\, p(x_2) \qquad \text{Equation B3}$$

A measure of statistical independence is described by the Central Limit Theorem, which states that the sum of several independent random variables tends towards a Gaussian distribution. Hence, by maximising the non-gaussianity of $\mathbf{WX}$, we are able to obtain the independent components of the system. There are several ways to measure non-gaussianity, such as kurtosis and negentropy which are described in detail by Hyvärinen et al. [15].