Principal Components Analysis of Spike Train Data: Towards an Architecture for the CARMEN Project

Hugo Hiden, Georgios Pitsilis, Paul Watson

Department of Computing Science
Claremont Tower
University of Newcastle upon Tyne
NE1 7RU
h.g.hiden@ncl.ac.uk

Abstract

Within the field of Neurological research, the processing and analysis of spike train data is a critical task. This data is obtained from electrodes planted at specific locations within a sample which return electrical information under a range of operations. The data sets collected by such analysis are substantial in size, as they represent voltage measurements sampled at 10 – 20 kHz. In addition, multiple electrodes will frequently by used, which dramatically increases the size of the collected data sets. In order to manage these data sets, scientists are frequently forced to make compromises in terms of the quantity of data they can store at any given time and also the analysis routines that can be applied to this data. This paper presents early work from the CARMEN (http://www.carmen.org.uk) project which aims to improve the analysis of neurological data by providing shared data repositories and services that can be used by Neuroscientists.

1. Introduction

The collection and processing of brain activity data is a challenging problem. Because of the difficulty in accurately locating probes, recorded signals are frequently corrupted by noise and unwanted signals from adjacent neurons which are not of interest to the experimenters. This results in an additional data processing step in order to separate signals from the neurons of interest. The quantity of data collected from parallel probes at high sample rates means that this processing step is typically hindered by the memory and storage constraints of personal computers. The CARMEN project aims to overcome these limitations by using Grid technologies to provided enhanced storage and data analysis services which will be used within the neuroscience community. This paper will introduce, via a simple neurological data analysis example, the concept of an Active Information Repository (AIR). The AIR is an entity that allows the co-location of raw data and associated analysis tools. Raw data is stored within the AIR using the Storage Request Broker (SRB) (Pailthorpe and Bordes, 2000), which allows multiple AIRs to be connected to share raw data. Services are provided within the AIR to perform data manipulation and anslysis tasks, and these services are orchestrated using Taverna workflows (Oinn *et al*, 2004) which enable scientists and end users to easily customise their work without requiring time consuming programming work.

2. Spike Train Data Processing

Spike train data is typically gathered using an electrode or clusters of electrodes inserted into a piece of brain tissue. Because of the close spacing of brain cells, it is usually the case that data collected from each electrode contains electrical information from a number of neurons within its vicinity. A typical sample of this data is shown below in Figure 1.

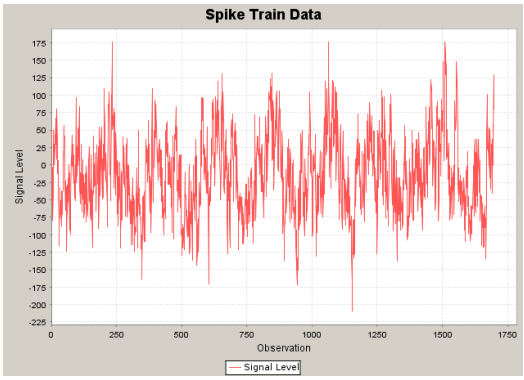

Figure 1: Raw Spike Train Data

The data shown in Figure 1 represents many thousand firings, therefore the first task to be performed in the analysis of these results is to separate out individual spikes from the raw data.

Within the raw spike train data, individual spikes will follow the form shown in Figure 2.

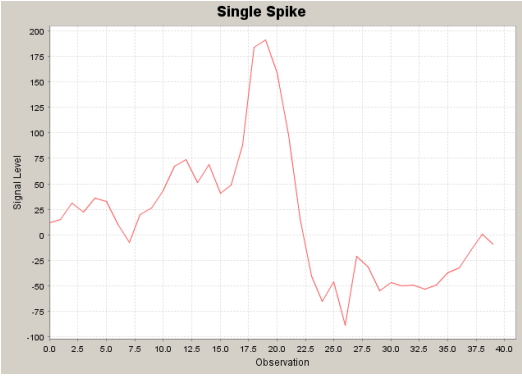

Figure 2: Single Separated Spike

In order to identify individual spike for further analysis, a common approach is to consider all data above a certain threshold value to be caused by the firing of a neuron. When data is detected above this threshold value this data, along with a specified portion of the data before and after the values above the threshold is extracted from the raw spike waveform values. This manipulation has been applied in Figure 2, and is evident in the data before and after the actual spike around observation number 18. In the case of Figure 2 a threshold value of 150 was used for the detection of spikes. A representative collection of spikes is shown below in Figure 3.

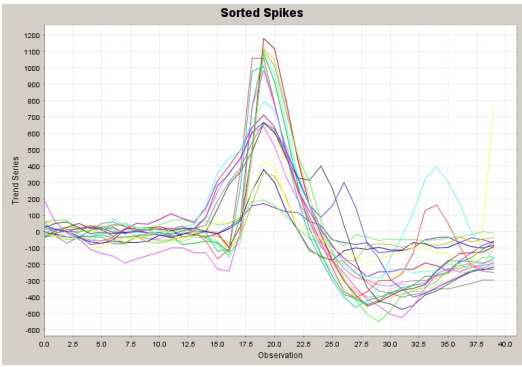

Figure 3: Multiple Sorted Spikes

Figure 3 also demonstrates the presence of a number of different spike types within the spike train. The clearest indication of this can be seen around the main firing time at observation number 18. There are three distinct levels of activity: One at a signal level in the region of 900, one in the region of 600 and one in the region of 300. Differences in signal profile can also be seen in the tail-off period after observation number 25. Whilst it may be fairly straightforward to classify the spike visually, such an approach does not scale when thousands of spikes are collected over the course of an experiment and an automated method is needed. A commonly adopted approach is to apply the multivariate statistical technique of Principal Components Analysis (PCA) to the spike data in order to identify the different firing patterns (for example, see Wheeler, 1999). PCA (Pearson, 1901) is a dimensionality reduction technique that operates by identifying the orthogonal planes of greatest variation within a set of input (**X**) data. These orthogonal planes are referred to a Principal Components (PCs) and are linear combinations of the individual columns within the input data set. In order to use PCA with the sorted spike data shown in Figure 3, the spike data (for '*s*' spikes) is stored in the following format:

$$\boldsymbol{X}=\begin{pmatrix} Obs_{1,1} & Obs_{1,2} & \cdots & Obs_{1,n} \\ Obs_{2,1} & Obs_{2,2} & \cdots & Obs_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ Obs_{s,1} & Obs_{s,2} & \cdots & Obs_{s,n} \end{pmatrix}$$

Equation 1: Data Representation

Where: $Obs_{i,j}$ refers to the i^{th} observation of signal level for the j^{th} spike.

The PCs for the data, which are linear combinations of the individual observations are represented as follows:

$$\boldsymbol{PC_1}=L_{1,1}Obs_{1,1}\cdots L_{n,n}Obs_{1,n}$$

Equation 2: Principal Component Calculation

Where: $\boldsymbol{L_n}$ refers to the weighting placed upon an individual column of data for each component (referred to as the *loading*).

Principal Components are calculated such that the first Component accounts for the maximum possible plane of variation in the original data, the second represents a plane orthogonal to the first accounting for the next largest plane of variance and so on. It is frequently the case that data sets exhibiting a high degree of co-linearity can be almost fully represented using the first few Components. In the case of the spike data shown in Figure 3, the majority of the variation can be accounted for by the first 2 Principal Components (Figure 4).

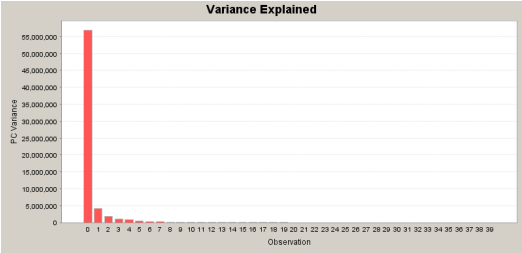

Figure 4: PC Variance

In effect the application of PCA to the spike data has compressed a 40 dimensional data set to a 2 dimensional one. This is represented by Figure 5, which shows the first two Principal Components on a single chart.

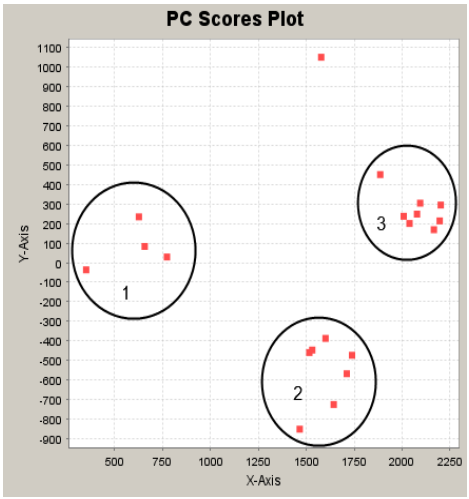

Figure 5: Scores Plot

From Figure 5 it can be seen the Principal Components cluster into three distinct regions, which represent the three different firing patterns identified in Figure 3.
An examination of the weightings assigned to the various observations throughout the lifetime of a spike (40 readings as shown in Figure 3) can be seen below in Figure 6.

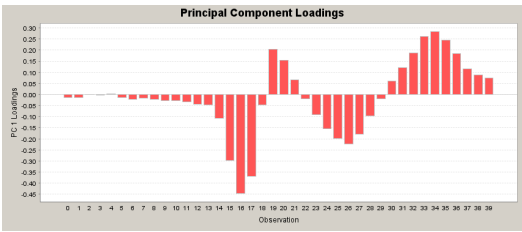

Figure 6: Loadings for first PC

Figure 6 illustrates that the PCA algorithm has attached little importance to the initial portion of the spike data (up to observation 13). An examination of Figure 3 shows this to be a period of relatively low activity with little to distinguish between individual spikes. In addition, PCA has assigned high weightings to the sections shown below in Figure 7, which correspond to the regions of the spikes that exhibit the greatest differences between the different neuron firings.

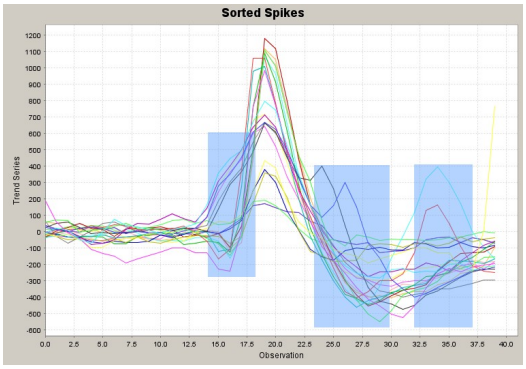

Figure 7: Areas of high weighting

Once a set of PCA weightings of the form shown in Equation 2 have been calculated on a set of spike data (the model above was calculated using 20 spikes), these weighting can be used to classify additional data as it is collected. This is especially beneficial as it means that the full set of spike data does not have to be loaded into memory in order to calculate a model. Figure 8 shows the classification of a larger set of 291 spikes using the PCA model calculated for the reduced spike set.

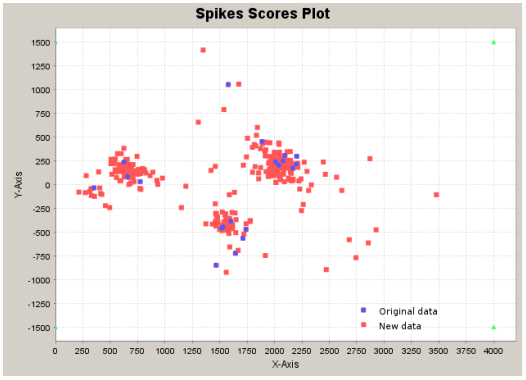

Figure 8: PCA Applied to new data

3. The Active Information Repository

Although Section 1 introduced a PCA spike classification technique which follows a fixed procedure, the CARMEN project aims to provide neuroscientists with a "toolbox" of services which they can use to process spike data, query historical results and perform interactive data processing and visualisation via the Active Information Repository (AIR) architecture below (Figure 9).

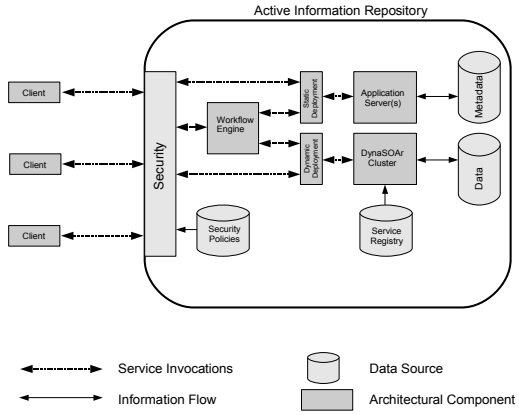

Figure 9: Active Information Repository

The AIR consists of a number of individual components, running across multiple physical machines. The following sections present a description of the various pieces of the AIR. Some of these aspects are already present, and have been used to generate the results shown in this paper, whilst others are either under development or discussion.

3.1. Data Storage

Raw Data is stored within the CARMEN project using the Storage Request Broker (SRB) software developed at the University of Queensland. This has been selected because it has demonstrated itself capable of managing extremely large file systems (in the Petabyte scale), with the possibility for federation across multiple sites. This federation ability allows multiple AIRs to share the raw data gathered by the various project partners. The currently deployed configuration includes a pair of federated SRBs running at York and Newcastle Universities. It is anticipated that, as the project progresses and more partners start to generate data, more local SRBs will be added to this configuration.

3.2. The individual Services

To produce these results, three separate services were used:

Spike detection service: This service takes as an input a URI reference to a raw experimental data file that has already been uploaded to the SRB. This data is then read from the SRB using the Java SRB access client, and a standard threshold based spike detection algorithm is applied. The resultant spike data is also placed into the SRB, and the service returns another URI pointing to this results file.

Principal Components Analysis Service: This service performs Principal Components Analysis on a set of data contained within the SRB and referenced using a URI which is the single input parameter for this service. The resulting Principal Component Scores are placed into a standard relational database. As part of the insertion into the database, an identifier string is generated which can be used to retrieve the Principal Component Scores. The generated identifier is then returned as the output from this service.

Scores Plot Service: This service produces a .gif file containing a Principal Component Scores plot (similar to that shown in Figure 5). This service takes the database identifier returned by the Principal Components Analysis Service and generates a .gif image containing a scores plot. The data for this plot is returned as a Base 64 encoded byte array by the web service.

3.3. Workflow Execution Engine

Because the ultimate aim of the Information Repository is to provide a resource that scientists can easily interact with in the normal course of their investigations, individual services are co-ordinated using a Workflow engine that can be configured graphically. At this stage, two systems are under investigation: the Taverna system (Oinn *et al, 2004*) and the BPEL (Thomas *et al*, 2006) implementation provided by the OMII. Regardless of the final workflow engine choice, workflows will be hosted within the Information Repositories and will be configured to access the services developed during the CARMEN project that have been specifically tailored to the analysis of neurological data.

3.4. Dynamic Service Deployment

Because one of the main aims in the development of the AIR concept has been to produce a system that can be used autonomously by scientists to manipulate and analyse data from many different classes of experiment, it is important for there to be flexibility in terms of the services that are contained within it. In addition, the fact that there are multiple AIRs, each with its own set of experimental data, in is important that services can be made available across AIRs to support localised data analysis and manipulation. These requirements have led us to include a dynamic service deployment element within the architecture for the AIRs. The dynamic service deployment features of the AIR have been built upon results from the DynaSOAr project (Watson *et al*, 2006). DynaSOAr is a prototype software platform for the dynamic deployment of Web Service resources within clusters of

Apache Axis / Tomcat based application servers. Services are uploaded to a central code repository and then deployed onto one of a pool of Tomcat servers when they are first invoked. Services are then deployed and undeployed dynamically depending on the load placed upon individual services. To support all of the functionality required by the AIR, additions are currently being made to DynaSOAr to take account of the physical location of the various data resources being operated upon as part of the service deployment criteria.

3.5. Metadata Requirements

As the volume of data and the number of available analysis services increases, the importance of providing a sufficiently rich metadata layer becomes significant. This metadata is required to describe both the scientific data and also the various services that are available plus the various runtime dependencies that are required in order to execute them. Although some software components that are being included within the AIR have a metadata associated with them (for example, DynaSOAr uses the Grimoires (Moreau *et al*, 2004) repository to maintain service descriptions, which is already metadata enabled), these need to be made available in a integrated fashion.

3.6.Security

As the project progresses, the CARMEN AIRs will be expected to support many different researchers working within a wide range of disciplines. In order to maintain confidence in the system it is essential to design a comprehensive and reliable security infrastructure. Research managers must be able to define security policies for all of the information and services present within the repositories. Given the distributed nature of the AIR architecture, these policies must be understood by all of the participating entities. The CARMEN project makes use of the security architecture defined in Newcastle by the GOLD project, which developed a sytem based upon the eXtensible Access Control Markup Language (XACML) (Sun Microsystems, 2004) that was applied to collaborative chemical process development. These policies are developed using a graphical tool, and uploaded to the AIRs. Enforcement is carried out by Policy Enforcement Points (PEPS) within the AIRs.

4. Conclusions

This paper has presented some aspects of the initial architecture of the data analysis services and infrastructure developed by the CARMEN project. Clearly, CARMEN is an ongoing project and not all of the aspects of the architecture presented here are finalised, although this paper has presented s framework which has already been used to generate some initial results and has been demonstrated to neuroscientists to gather feedback.

Whilst this set of services represents a fairly simple and, arguably, contrived use case, they have been used to develop and validate the Active Information Repository concept. An expanded range of more sophisticated spike detection and processing services will be developed by the neuroscientists working on the CARMEN project and will be included within the Active Information Repository as the project progresses.

5. References

Jerry Thomas, Doug Todd, Harish Gaur, Lawrence Pravin, Arun Poduval, The Hoa Nguyen, Jeremy Bolie, Yves Coene, Stany Blanvalet, Markus Zirn, Matjaz Juric, Sean Carey, Michael Cardella, Kevin Geminiuc, Praveen Ramachandran. (2006): "*BPEL Cookbook: Best Practices for SOA-based integration and composite applications development*" ISBN 1904811337, 2006

Sun Microsystems (2004): "*Sun's XACML Implementation.*" http://sunxacml.sourceforge.net/, 2004

Luc Moreau, Simon Miles, Juri Papay, Keith Decker, and Terry Payne. (2003): "*Publishing Semantic Descriptions of Services.*" Technical report, Global Grid Forum, 2003.

Pailthorpe, B. and N. Bordes. (2000): "*Scalable volume visualisation, ranging from astrophysics, through brain mapping, to oceanography.*" In B. Ernquist, L. Johnsson, M Hammill and F. Short (eds), Simulation and Visualization on The Grid, Lecture Notes in Computational Science, pp. 122-134. Springer-Verlag (2000).

Tom Oinn, Matthew Addis, Justin Ferris, Darren Marvin, Martin Senger, Mark Greenwood, Tim Carver, Kevin Glover, Matthew R. Pocock, Anil Wipat and Peter Li. (2004): "*Taverna: A tool for the composition and enactment of bioinformatics workflows*" Bioinformatics Journal 20(17) pp 3045-3054, 2004, doi:10.1093/bioinformatics/bth361.

Pearson, K. (1901). "*On Lines and Planes of Closest Fit to Systems of Points in Space*". Philosophical Magazine 2 (6): 559–572.

Watson, P., Fowler, C., Kubicek, C., Mukherjee, A., Colquhoun, J., Hewitt, M. and Parastatidis, S. (2006): "*Dynamically Deploying Web Services on a Grid using Dynasoa*r" In Proceedings of the Ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, ISORC 2006, 24-26 April 2006, Gyeongju, Korea Lee, S. , Brinkschulte, U., Thuraisingham, B. and Pettit, R.G. (eds) pp. 151-158 IEEE Computer Society 2006

B.C. Wheeler. (1999): "*Automatic discrimination of single units*" in: M.A.L. Nicolelis (Ed.), Methods for neural ensemble recordings, 1999, pp. 61–78.