Data Analysis and Model Classification
Mini-project 1: Description and Assignment

Ruslan Aydarkhanov Bastien Orset Julien Rechenmann
Ricardo Chavarriaga José del R. Millán

September 25, 2017

Objective

As we saw in class, there exists a class of problems, where the labels of individual samples is unknown and classes must be guessed from similarities in the data samples. In this mini-project we will familiarize with an unsupervised learning technique (clustering) and we will see how we can assess its quality with validation and cross-validation.

Introduction

Traditional neuroscience involves the study of population cells where neural activity is recorded. Neural prosthesis field may involved the recording of hundred of electrodes. To be able to decode neural activity, it is necessary to follow several processing steps:

Spike detection The raw signal is usually filtered between 300 and 3000 Hz to get rid of low-frequency oscillations and high-frequency noise. Then a threshold is defined heuristically (usually a multiple of the background noise standard deviation), and every activity exceeding this limit, is labeled a spike.

Epoching Then a window of around 2 ms length is extracted around the peak, which contains the full spike time course.

Feature extraction Before clustering, the feature space should be restricted to obtain the best results. One of the traditional methods to do that is principal component analysis. It is a common process to reduce the feature space dimensions.

Clustering Now every spike needs to be assigned to a certain neuron. The basic assumption is, that spikes from the same neuron will always have the same shape in the recorded signal. The difficulty is, that there are no labels available, and even the number of distinguishable neurons is not known. A widespread method is to perform k-means clustering.

Firing rate computation Once every spike is assigned to a certain neuron, the firing rate of each neuron in e.g. 100 ms windows can be computed.

Performing this whole cascade of signal processing and analysis would not only be extremely laborious, but would also exceed the contents of this course. Therefore we will look only into the unsupervised learning part: the k-means clustering.

Dataset description

From an electrode implanted in a non-human primate motor cortex, the surrounding electrical activity was recorded for 30 seconds with a sampling frequency of 30 KHz. The spike detection and epoching has already be done. Therefore you are supplied with a large number of different detected spikes, each of

them extracted in a 2 ms time window with 64 time points. You will find the data in the `spikes.mat` file on the moodle, stored in the variable `spikes` and their projections after principle component analysis in the variable `spikesPCA`.

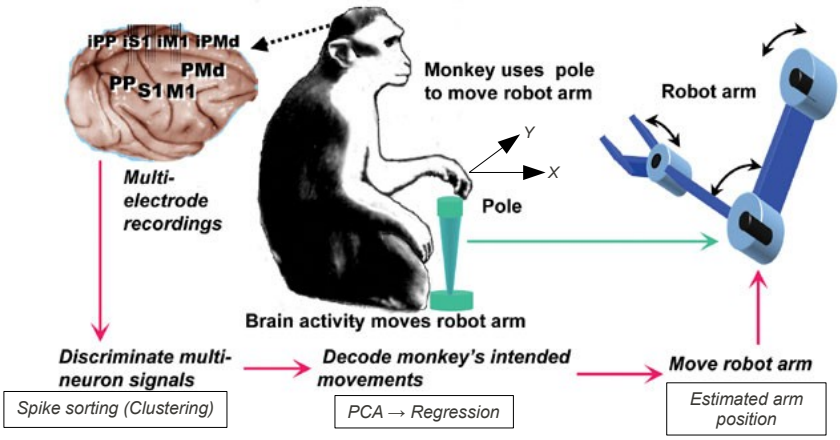

Figure 1: Experimental setup. Steps for data analysis are also shown.

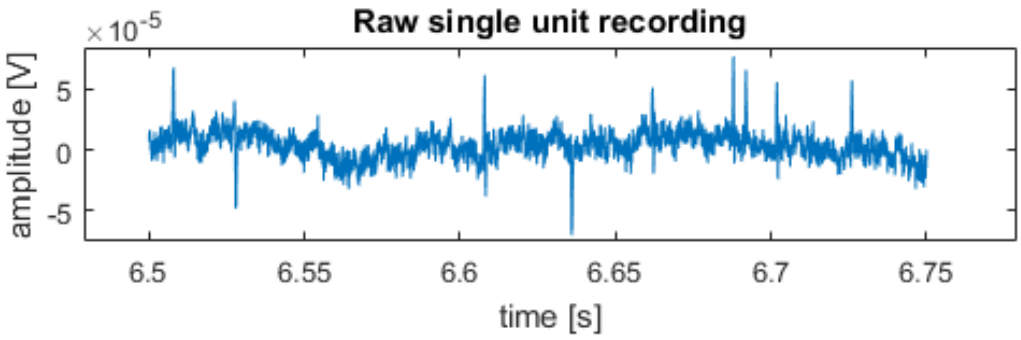

Figure 2: Short time window from the raw recording of electrode 9 of the implanted array. The sharp spikes of firing neurons are clearly visible.

k-means clustering

Based on the similarity of the features, we want to cluster the samples into classes (= different neurons in this case). One simple yet effective method is the *k-means* clustering algorithm.

The algorithm assigns a set of samples $s_1, s_2, \ldots, s_n$ to k different clusters $C_1, C_2, \ldots, C_k$ so that the within-cluster sum of square distances is minimized:

$$\underset{C}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{s \in C_i} \|s - \mu_i\|^2 \qquad \text{with} \qquad \mu_i = \text{mean of } C_i$$

The algorithm walks through the following steps:

1. **Initialization** Determines the starting centroids of the clusters. This can be done in many ways. The easiest is to randomly pick from the observations.
2. **Assignment** Every sample gets assigned to the closest centroid according to the chosen norm (hence nearest neighbor).
3. **Update centroids** Update the centroids to be the mean of the assigned samples.

Steps 2 and 3 are repeated until there is no change in assignment anymore.

Training, testing and validation

Evaluating model quality is difficult since we want to know how good it will predict new unknown data. With a large dataset, we can simply separate it in three different subsets (figure 3). A typical model quality evaluation involves:

1. Partitioning a sample of data into complementary subsets: training and testing. Training subset can be then separated in two different subsets: training and validation (for hyper-parameters estimation) sets.
2. Performing the analysis on training subset
3. Validating the analysis and the hyper-parameters on the other validation subset
4. Estimating the quality of the final model on the testing set

Figure 3: Scheme of separating dataset for training, validating and testing.

Cross validation Cross validation is a model validation used to assess how your model can be generalized to another dataset. The goal of cross-validation is to estimate the expected level of fit of a model to a data set that is independent of the data that were used to train the model. It can be used to estimate any quantitative measure of fit that is appropriate for the data and model. To reduce variability, multiple rounds of cross-validation are performed using different partitions, and the validation results are combined (e.g. averaged) over the rounds to estimate a final predictive model.

Main Challenges (Checklist)

- Spike visualization
- k-Means clustering training and testing.
- Impact of dataset size on clustering quality
- Cross-validation to assess clustering quality

The Report

At the end of the first mini-project you are requested to hand in a report, motivating the choices you made and discussing the results you obtained from your implementation. This report should cover **all aspects** of your analysis. You should demonstrate the understanding of spike sorting utility. We expect an integral consistent text rather than unrelated parts reflecting the guide sheets. The guide sheets are there to help you familiarize with different aspects of the data analysis. Try to combine the knowledge you acquired during the course. Please stick to the following structure:

Introduction In a few sentences (1-2 paragraphs), state the research question and the goal of the report. Give a **brief** description of the methods you have used (max. one page).

Methods Describe step by step and explain in detail what you did exactly, including

- If you have any hyper-parameters, how do you choose them (e.g. initialization of k-means clustering)?

- Make sure it is always clear on which set (training set, testing set, ...) you perform the various steps (e.g. optimization, evaluation, ...). Use the correct nomenclature.
- How do you assess the performance of your model and why?

Results This section is uniquely reserved for the results you obtained. Of course, it is not necessary to show the outcome of every single thing you have tried for this mini-project. So choose well what data to show and what not to show. You should be creative and pragmatic about how to display your data.

Discussion Use this chapter to briefly sum up your results and discuss about following points:

- What are the particularities of your dataset?
- Which methods did you use and why?
- Anything especially difficult or clever you did, and why?
- Any points where the methods did not suffice, and you would be in need of more advanced concepts.

Note: Your report **must not exceed 6 pages** (excluding the title page)! Every additional page will not be considered for grading.

Report writing: how to's

- There are different ways how you can present your results: you could just write them in the continuous text, make a table, or visualize them in a plot.
- Graphs are an extremely powerful tool; they can make it easier to grasp results and their implications. But to unleash their full potential, the reader has to understand **everything** the graph is showing. Therefore, make sure your graphs **always** contain
  - a title (`title`)
  - axis labels for all axes (`xlabel/ylabel`)
  - meaningful axis units. You can specify them manually, if necessary, with `axis`
  - a legend, to show which color / linestyle corresponds to what data (`legend`)
  - a caption in the report, describing what the graph shows, and why this is important.

  Below you can find an example of how to present a figure so to maximize content and readability. (The data presented here is not related to any mini-project.)
- Be aware that a result is not significant until tested for significance (e.g. by using a t-test)! So if you state that something is significant, **always** put the test statistic (e.g. p-value and chosen level of significance).
- Use exact language wherever possible! Describe dynamics and uncertainties always in a scientific way and **with numbers**.

  Examples from previous years:
  - Since there is some scattering in the y axis, we could call it 'moderate' negative correlation. - *Yes you can, but you shouldn't. Why not just report the exact value of the correlation?*
  - If I repeat the function more than 10 times the confusion matrix varies only slightly and the clustering result is always the same, it is always centered on -1.5. - *It would be so much more helpful, if you just report the mean and standard deviation of the elements across the 10 confusion matrices.*
  - As a result, the resulting centroids are not the same, but within the margin of error. - *There is no such thing as a standard "margin of error". Please simply state the mean and standard deviation.*

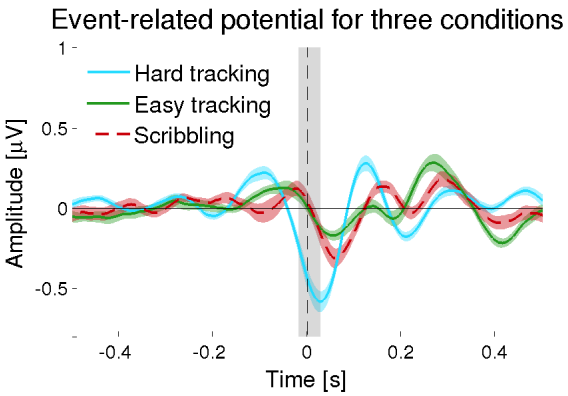

Figure 4: The event-related potential (ERP) is modulated by the difficulty of the task. In hard tracking conditions subjects elicit on average higher amplitude ERPs. The thick lines are the subject means, and the shaded areas indicate the standard error of the means. The gray-shaded bar marks the time of significant difference between the hard and both other tasks ($p = 0.0084$).

Submission

The **deadline** for the report submission is **October 15, 2017** before **23:59**. Make sure your report is in .pdf format and name it according to following convention: `Miniproject1_Group`⟨*groupnumber*⟩`.pdf`, e.g. *Miniproject1_Group99.pdf*. Please also attach any MATLAB code used to obtain the results in a .zip file. The code will not be graded and is looked into only in cases where results or plots seem dubious to find the reason (bug in the code or conceptual error).

Your finished report has to be uploaded to the moodle **only by one person per group**. The submission function will be activated approximately one week before the deadline. **Don't forget to click 'Submit' in the end!**