ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/269294789

Spike Train kernels for multiple neuron recordings

Conference Paper *in* Proceedings - ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing · May 2014

DOI: 10.1109/ICASSP.2014.6854754

CITATION
1

READS
35

1 author:

Taro Tezuka
University of Tsukuba

62 PUBLICATIONS **351** CITATIONS

SEE PROFILE

All in-text references underlined in blue are linked to publications on ResearchGate, letting you access and read them immediately.

Available from: Taro Tezuka
Retrieved on: 09 May 2016

SPIKE TRAIN KERNELS FOR MULTIPLE NEURON RECORDINGS

Taro Tezuka

Faculty of Library, Information and Media Science
University of Tsukuba
1-2 Kasuga, Tsukuba, Japan 305-0821
tezuka@slis.tsukuba.ac.jp

ABSTRACT

There is a growing interest in analyzing multineuron spike trains, which are spike timing data obtained from multiple neurons in the brain. Kernel methods have been successful in clustering and classification of single-neuron spike trains. We extend these methods to multineuron spike trains. Among various possible extensions, the *mixture kernel* was found to be most effective. The optimum parameter obtained from training this kernel was close to a biologically plausible value, suggesting that our approach is effective for seeking an appropriate model for the activity of a set of neurons.

Index Terms— Spike train, multiple neuron, multineuron, kernel methods, distance measure

1. INTRODUCTION

A spike train is a sequence of time points indicating that a neuron fired action potentials [7, 21]. It is often assumed that most if not all information conveyed by one neuron to another is expressed in a spike train [20]. Spike trains are also important for applications such as brain machine interfaces (BMI), since the aim of such systems is to decode information conveyed by neurons and make direct communication possible between the brain and machines.

One of the most common goals of BMI is to distinguish the subject's intent from different possibilities. In machine learning, this is called a classification task, and it can be performed when a distance is defined among observed data, in this case, spike trains. Thus, much work has been devoted to defining appropriate distances between spike trains [24, 12, 16, 5]. It has been pointed out, however, that rather than looking at distances between spike trains, considering their kernels could be more productive [17, 18]. This is because kernels can be used for various other tasks in machine learning, including regression, clustering and dimension reduction.

Recently, recordings of spike trains coming from multiple neurons are increasingly becoming common. Such data can be obtained either through multiple site recordings [1] or spike sorting [19]. The existing work, however, has only dealt with either distances between multiple neuron spike trains [4, 11, 10, 14, 1] or kernels on single neuron spike trains [17, 18]. The aim of this paper is to propose kernels for spike trains taken from multiple neurons. Specifically, we propose *multineuron spike train kernels*, which are kernels defined over the sets of spike trains obtained from multiple neurons. Like the original spike train kernels (single-neuron spike train kernels), these can be used in different machine learning tasks, including classification, clustering, dimension reduction, and prediction.

2. RELATED WORK

Various distances have been proposed for measuring the similarity between spike trains, including the Victor-Purpura distance [25], van Rossum distance [23], and ISI (Inter Spike Interval) distance [13]. There are also probabilistic models such as the one proposed by Dauwels et al. [6]. The Victor-Purpura distance is often called the edit distance, since it defines the distance between spike trains t and s as the minimum cost of creating s from t by a sequence of basic edit procedures [25]. In the van Rossum distance, spike trains are converted into continuous functions by convolution, and the squared difference of functions are integrated to give the distance [23].

Recently, there have been various proposals that define distances between multineuron spike trains. One example is the multineuron van Rossum distance proposed by Houghton and Sen [11, 10]. Kreuz et al. extended the ISI (inter-spike interval) distance to include multineuron spike trains [14]. Aronov et al. defined multineuron distances that are extensions of the Victor-Purpura distance [1]. Brown et al. gives an overview of different approaches in defining multineuron spike train distances [4].

3. METHOD

3.1. Spike train kernel

The memoryless cross intensity kernel (mCI kernel) k_λ defined below is a single-neuron spike train kernel proposed by Antonio Paiva [17, 18]. Our aim in this paper is to extend this scheme to multineuron spike trains.

$$k_\lambda(\xi, \zeta) = \int dt \lambda_\xi(t) \lambda_\zeta(t) \tag{1}$$

λ_ξ and λ_ζ are intensity functions for spike trains ξ and ζ, respectively. It is common practice to express the intensity function by convolving the spike train and a *smoothing function* h, as in $\lambda(t) = \sum_{n=1}^{N} h(t - t^n)$. Here, t^n is the timing of the nth spike and N is the total number of spikes in the train. In our experiments, we used a Gaussian function for h, since spike timings are assumed to have stochastic fluctuation, which could be modeled using a Gaussian. This is a very common choice in the analysis of spike trains.

3.2. Kernels on multineuron spike trains

Consider a case where spike trains are recorded at M sites, or channels. Let $\mathcal{S}$ be the set of all possible spike trains. We express a spike train taken from a neuron m by x_m and use x to express a tuple $(x_1, ..., x_M)$. In other words, $x \in \mathcal{S}^M = \mathcal{S} \times ... \times \mathcal{S}$, where $\mathcal{S}$ appears M times on the right hand side. We call x_m a component of x.

The most general way to define a kernel in this space is to use the multineuron spike trains x and y directly as variables, without imposing any structure. Such a general kernel can be expressed as $k(x, y)$. In order to make parameter estimation more tractable, however, we would like to impose some structure on it. Since we assumed that x and y have components, we can define a kernel on a pair of their components, x_m and y_n. Such a kernel can be expressed as $k_{mn}(x_m, y_n)$. This type of kernel is more restrictive than the general form $k(x, y)$. For example, we can no longer consider a kernel that is dependent on more than one component for a single variable, such as $k(x_m + x_{m'}, y_n + y_{n'})$, where $m \neq m'$ or $n \neq n'$. We now define a kernel $\tilde{k}$ whose components are kernels k_{mn}.

$$\tilde{k} = (k_{1,1}, k_{1,2}, ..., k_{1,M}, k_{2,1}, ..., k_{M,M})^T \tag{2}$$

Since we assume that the value of a kernel is a real number, the codomain of $\tilde{k}$ is $\mathbb{R}^{M^2}$. Let $f : \mathbb{R}^{M^2} \to \mathbb{R}$ be a function defined by repetitive application of operations that the positive definite kernels have the closure property with [22]. For example, f can be defined using the sum and product, since the sum and product of positive definite kernels are positive definite. A positive coefficient multivariate polynomial of the components of $\tilde{k}$ is one example of f. Using f, we can construct a new kernel k_f as follows.

$$k_f(x, y) = f(\tilde{k}) \tag{3}$$

If we let f be any function consisting of operations with the closure property, it is difficult to determine a single function from a finite sized training data set. We therefore would like to restrict the form of f to make the parameter estimation tractable. In other words, we would like to think of a model restricting the degree of freedom of f. In the following subsection, we give specific examples of multineuron spike train kernels defined by restricting f to specific forms.

3.3. Specific examples of multineuron kernels

Mixture kernel: One of the most natural ways to define a kernel on multineuron spike trains is to use a linear combination, or mixture, of the components of $\tilde{k}$.

$$k_P(x, y) = \sum_{m=1}^{M} \sum_{n=1}^{M} P_{mn} k_{mn}(x_m, y_n) \tag{4}$$

Here, P_{mn} is a real scalar. We use P to denote a matrix whose (m, n)th-entry is P_{mn}. x_m and y_n are single neuron spike trains, and k_{mn} is a positive definite kernel for a pair of single neuron spike trains. In the following experiments, we used the memoryless cross intensity (mCI) kernel for k_{mn}, but other kernels can be chosen as well. We will call k_P the *mixture kernel*. The off-diagonal entries of P indicate the interaction between different neurons. If P is a diagonal matrix, it means that spike trains from different neurons should be considered separately. When the diagonal entries and off-diagonal entries are all 1, it does not matter which neuron fires, so the neurons are interchangeable.

Derived polynomial kernel: A common way to extend a given kernel is to use the *derived polynomial kernel* [22]. In the definition below, r is an arbitrary real number, and p is a positive integer, and k is an arbitrary kernel.

$$k_{r,p}(x, y) = (k(x, y) + r)^p \tag{5}$$

In the experiment, we used a mixture kernel k_P for k, but other kernels could be used as well. Increasing p results in a stronger non-linear effect. For a set value of p, increasing r reduces the non-linearity. This is because when $(k(x, y) + r)^p$ was expanded and the expression was considered as a polynomial of $k(x, y)$, the higher order terms would have relatively smaller coefficients compared with the lower order terms.

R-convolution kernel for tuples: Another kernel that can be constructed from component-wise kernels is the *R-convolution kernel* for tuples, defined below [8]. For $k_{m,m}$, we can use a positive definite single-neuron kernel such as the mCI kernel.

$$k_M(x, y) = \prod_{m=1}^{M} k_{m,m}(x_m, y_m) \tag{6}$$

This model is less flexible than the previously mentioned ones because it lacks parameters to be adjusted.

4. EVALUATION

In this section, we first describe the evaluation criterion, and then compare the three proposed kernels defined in the previous section. We also compare the best performing multineuron spike train kernel with the single-neuron spike train kernel. Finally, we compare our proposal with a conventional multineuron distance measure. For implementation, we used Matlab.

Since the multineuron spike train kernel is a newly proposed concept in this paper, we ought to evaluate it by converting it to a distance. This will make a comparison with the existing multineuron distance measures possible. From a positive definite kernel k, we can obtain a pseudo-distance d, i.e., $d(x, y) = \sqrt{k(x, x) - 2k(x, y) + k(y, y)}$ [17]. A good pseudo-distance is one that takes on a large value between elements belonging to different classes and takes on a small value for elements belonging to a same class. In such case, the classifier could effectively assign each element to different classes. In this respect, pseudo-distances can be evaluated in a same way as distances.

4.1. Evaluation using Fisher's LDA

A classifier is a program that takes an observed data element and returns a class, to which the data element is assumed to belong to. In Fisher's linear discriminant analysis (FLDA), one seeks a weight vector w that maximizes the objective function $J(w) = \frac{w^T S_B w}{w^T S_W w}$ [3]. S_B is called the between-classes variance and S_W is called the within-class variance. The reason for using this objective function $J(w)$ is to maximize the ratio of the between-classes variance $w^T S_B w$ and the within-class variance $w^T S_W w$ when sample vectors $\{v_i\}$ are projected to a one-dimensional space using w. In this paper, since we are evaluating distances, they are always non-negative real numbers. In other words, sample vectors are present in the 1-dimensional real vector space $\mathbb{R}$. There is no need to project them using w. Therefore, w can be considered to be a scalar constant and can be removed from the objective function. As a result, we get $J = \frac{S_B}{S_W}$, where $S_B = (\mu_2 - \mu_1)^2$ and $S_W = S_1 + S_2$. Here, μ_1 and μ_2 are vectors representing the sample mean for elements in class 1 and class 2, respectively. S_1 represents the variance of elements belonging to class 1 and S_2 is that for class 2. Although J is no longer a function of w, it depends on the distance d. Indeed, $(\mu_2 - \mu_1)^2$, S_1, and S_2 depend on the distance being used. Since J is the measure for the "separateness" between classes, we use it to compare different distances. We would like to find the distance that maximizes J.

We use spike train recordings to evaluate distances in the following way. Let Q be the set of conditions and X be the set of all trials. In the multineuron case, a trial consists of spike train recordings taken from multiple sites. Let $q \in Q$ represent a condition. If the set of trials recorded under condition q was represented by X_q, we can get a set of distances $\{d(x, y)\}_{x,y \in X_q}$. This set makes a class to be used in FLDA. We can also obtain a set $\{d(x, y)\}_{x \in X_q, y \in X \setminus X_q}$, which will make another class for FLDA. We use these two classes to calculate the objective function J. Algorithm 1 illustrates the evaluation procedure.

Algorithm 1 Evaluation of distances

```
for each condition $q$ in $Q$
    for each data $x \in X_q$
        for each data $y \in X_q \setminus \{x\}$
            create a sample at $d(x, y)$ with the class label 1
        for each data $y \in X \setminus X_q$
            create a sample at $d(x, y)$ with the class label 2
calculate the objective function $J$ of FLDA using samples
```

4.2. Data set

We used multineuron spike train data described in Aronov et al. [1], which is available publicly [1]. It is recorded at the V1 (primary visual field) of a macaque monkey when spatial sinusoidal grating waves were presented. The recordings were made at two sites, i.e. $M = 2$. The experiment was performed under four conditions, each corresponding to a specific spatial phase, namely 0, $\frac{\pi}{2}$, π, and $\frac{3\pi}{2}$. For each condition, 64 trials were carried out, each trial lasting 1 second.

4.3. The effect of σ_h on the mixture kernel

To evaluate the effect of the parameter, we changed $\sigma_h \in [0, 1]$, which is the sole parameter for the Gaussian smoothing function h. The unit for σ_h is in seconds. For P, we used the identity matrix. Figure 1 shows the results.

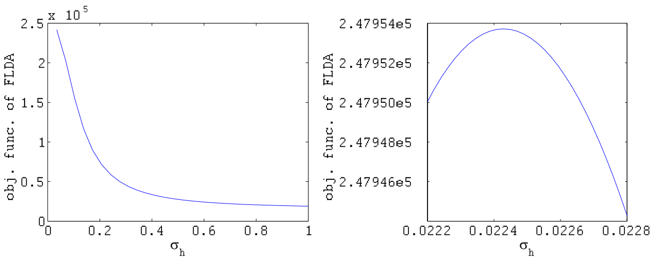

Fig. 1. Changing σ_h of the mixture kernel

$J(\sigma_h)$ was maximized near $\sigma_h = 0.0224$, i.e. at 22.4 ms. Surprisingly, this value is close, at least in the order of magnitude, to the time constant of the smoothing function used in the synapse model proposed by Conor Houghton, which used $\tau = 12$ ms [9, 12]. Moreover, Narayan et al. applied the van Rossum distance to spike recordings from zebra finches and obtained the highest performance when the time constant τ of the decaying exponential kernel was set to 12.8 ms [15]. In other words, although we did not put any biological considerations into our model, it nonetheless resulted in biologically

[1]Neuroanalysis.org, `http://neuroanalysis.org`

plausible values. This indicates that some of the basic parameters in the modeling of neurons could be revealed using multineuron spike train kernels.

4.4. The effect of P on the mixture kernel

We changed P to see its effect. Since there is no prior information about each recording site, we only considered the "fair" case, where all of the diagonal entries are equal to each other and the off-diagonal entries are equal to each other as well. The diagonal entries were set to 1, and the off-diagonal entries were set to a. In other words, $P = (1-a)I + a\mathbf{1}\mathbf{1}^T$. In the experiment, we searched for the value of $a \in [-1, 1]$ where $J(a)$ is maximized. σ_h was set to 0.0224, since this was the best value obtained in the previous experiment. The result indicated in Figure 2 shows that the objective function $J(a)$ is maximized when a is near zero, although slightly negative. It may be due to two spike trains carrying different information, therefore summing the kernels decreases $J(a)$.

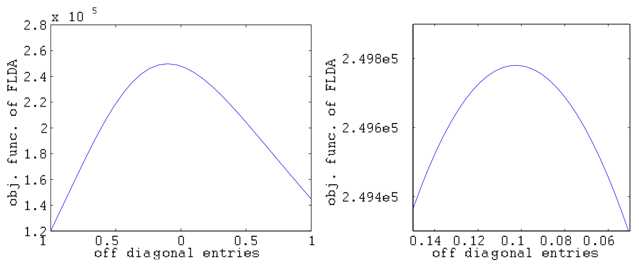

Fig. 2. Changing a of the mixture kernel

4.5. Effect of r on the deduced polynomial kernel

For $r \in [0, 1000]$, $J(r)$ was monotonically increasing and was maximized at $r = 1000$. As can be seen from Equation 5, the large value of r means the lower order terms of the polynomial are relatively larger than the higher order terms. This indicates that the higher order terms are less important, and the mixture kernel is sufficient at least for this data set. We would like to test other possible polynomial kernels in future work.

4.6. Effect of σ_h on the R-convolution kernel

For the R-convolution kernel for tuples, the maximum value $J = 1.364 \times 10^{-6}$ was obtained when $\sigma_h = 0.0630$. For the mixture kernel, we obtained a higher value, namely $J = 2.480 \times 10^{-5}$. This indicates that the mixture of interactions is a better model than their product.

4.7. Comparison of single-neuron and multineuron mCI

Next, we compared the performance of the multineuron mCI with that of the single-neuron mCI. In Figure 3, the solid line is the result for the multineuron mCI, based on the mixture kernel. The dotted line and broken line are for the single-neuron mCI, taken from each of the two recording sites. It is clear that J is much higher for the multineuron mCI.

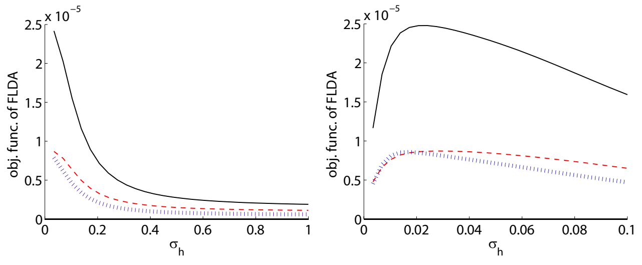

Fig. 3. Single-neuron mCI and multineuron mCI

4.8. Comparison with other distance measure

Since the mixture kernel performed best among proposed kernels, we compared it with a conventional method, the multineuron van Rossum distance. To calculate the latter, we used a program implemented by Thomas Kreuz [2][10]. When we used their original parameter settings, namely $rsd\_tc = 0.005$ and $cosalpha = 0.5$, we got $J = 2.995 \times 10^{-6}$. For the mixture kernel, which obtained $J = 2.480 \times 10^{-5}$, turned out to perform better in terms of separating the spike trains obtained under different conditions.

5. CONCLUSION

We proposed multineuron spike train kernels and compared their performance with an existing measure on multineuron spike train distance. The results showed that the mixture kernel performs the best. We also obtained optimal parameters that correspond to biologically plausible values. Such parameters can be used for large-scale simulations of realistic neural networks. Selecting a kernel corresponds to choosing a model of neural coding. Since models that best describes the activity of neurons are what many theoretical neuroscience researchers are seeking, it indicates the value of pursuing the research on multineuron spike train kernels.

In future work, we plan to evaluate the performance of the kernel more directly, without converting it into a distance. This will make it impossible to compare with existing distance measures, but its absolute performance can still be measured. Also, we plan to test our method using other data sources, both real and simulated.

Acknowledgment This work was supported in part by JSPS KAKENHI Grant Numbers 21700121, 25280110, and 25540159.

[2]THOMAS KREUZ online, `http://wwwold.fi.isc.cnr.it/users/thomas.kreuz/sourcecode.html`

6. REFERENCES

[1] Dmitriy Aronov, Daniel S. Reich, Ferenc Mechler and Jonathan D. Victor, Neural coding of spatial phase in V1 of the macaque monkey, Journal of Neurophysiology, Vol.89, pp.3304-3327, 2003.

[2] Christian Berg, Jens Peter Reus Christensen and Paul Ressel, Harmonic Analysis on Semigroups, Springer, 1984.

[3] Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[4] Emery N. Brown, Robert E. Kass and Partha P. Mitra, Multiple neural spike train data analysis: state-of-the-art and future challenges, Nature Neuroscience, Vol.7, No.5, pp.456-461, 2004.

[5] Daniel Chicharro, Thomas Kreuz and Ralph G. Andrejak, What can spike train distances tell us about the neural code?, Journal of Neuroscience Methods, Vol.199, pp.146-165, 2011.

[6] Justin Dauwels, François Vialatte, Theophane Weber and Andrzej Cichocki, On similarity measures for spike trains, in Proceeding of the 15th International Conference on Advances in Neuro-Information Processing, pp.177-185, 2009.

[7] Peter Dayan and L. F. Abbott, Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems, MIT Press, 2001.

[8] David Haussler, Convolution kernels on discrete structures, UCSC-CRL-99-10, 1999.

[9] Conor Houghton, Studying spike trains using a van Rossum metric with a synapse-like filter, Journal of Computational Neuroscience, Vol. 26, No.1, pp.149-155, 2009.

[10] Conor Houghton and Thomas Kreuz, On the efficient calculation of van Rossum distances, Network, Vol.23, No.1-2, pp.48-58, 2012.

[11] Conor Houghton and Kamal Sen, A new multi-neuron spike-train metric, Neural Computation 2008, Vol.20, No.6, pp.1495-1511, 2008.

[12] Conor Houghton and Jonathan Victor, Measuring representational distances - the spike-train metrics approach, in Nikolaus Kriegeskorte and Gabriel Kreiman (Eds), Understanding Visual Population Codes: Towards a Common Multivariate Framework for Cell Recording and Functional Imaging, MIT Press, 2011.

[13] Thomas Kreuz, Julie S. Haas, Alice Morelli, Henry D.I. Abarbanel and Antonio Politi, Measuring spike train synchrony, Journal of Neuroscience Methods, Vol.165 pp.151-161, 2007.

[14] Thomas Kreuz, Daniel Chicharro, Ralph G. Andrejak, Julie S. Haas and Henry D.I. Abarbanel, Measuring multiple spike train synchrony, Journal of Neuroscience Methods, Vol.183, No.2, pp.287-299, 2009.

[15] Rajiv Narayan, Gilberto Graña and Kamal Sen, Distinct Time Scales in Cortical Discrimination of Natural Sounds in Songbirds, Journal of Neurophysiology, Vol. 96, pp.252-258, 2006.

[16] Richard Naud, Felipe Gerhard, Skander Mensi and Wulfram Gerstner, Improved similarity measures for small sets of spike trains, Neural Computation, Vol.23, pp.3016-3069, 2011.

[17] Antonio R.C. Paiva, Il Park and Jose C. Principe, A reproducing kernel Hilbert space framework for spike train signal processing, Neural Computation, Vol.21, No.2, pp.424-449, 2009.

[18] Antonio R.C. Paiva, Il Park and Jose C. Principe, Inner products for representation and learning in the spike train domain, in Karim G. Oweiss (Ed), Statistical Signal Processing for Neuroscience and Neurotechnology, Academic Press, 2010.

[19] Rodrigo Quian Quiroga, Zoltan Nadasdy and Yorum Ben-Shaul, Unsupervised spike sorting with wavelets and superparamagnetic clustering, Neural Computation, Vol.16, pp.1661-1687, 2004.

[20] Fred Rieke, David Warland, Rob de Ruyter van Steveninck and William Bialek, Spikes: Exploring the Neural Code, MIT Press, 1997.

[21] Larry Squire, Darwin Berg, Floyd E. Bloom, Sascha du Lac, Anirvan Ghosh and Nicholas C. Spitzer, Fundamental Neuroscience (4th Ed), Academic Press, 2012.

[22] John Shawe-Taylor and Nello Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, 2004.

[23] M.C.W. van Rossum, A novel spike distance, Neural Computation, Vol.13, pp.751-763, 2001.

[24] Jonathan D. Victor, Spike train metrics, Current Opinion in Neurobiology, Vol.15, pp.585-592, 2005.

[25] Jonathan D. Victor and Keith P. Purpura, Spike metrics, in Nikolaus Kriegeskorte and Gabriel Kreiman (Eds), Understanding Visual Population Codes: Towards a Common Multivariate Framework for Cell Recording and Functional Imaging, MIT Press, 2011.