Inner products for representation and learning in the spike train domain[*]

António R. C. Paiva, Il Park, and José C. Príncipe

May 11, 2010

Abstract

In many neurophysiological studies and brain-inspired computation paradigms, there is still a need for new spike train analysis and learning algorithms because current methods tend to be limited in terms of the tools they provide and are not easily extended. This chapter presents a general framework to develop spike train machine learning methods by defining inner product operators for spike trains. They build on the mathematical theory of reproducing kernel Hilbert spaces (RKHS) and kernel methods, allowing a multitude of analysis and learning algorithms to be easily developed. The inner products utilize functional representations of spike trains, which we motivate from two perspectives: as a biological-modeling problem, and as a statistical description. The biological-modeling approach highlights the potential biological mechanisms taking place at the neuron level and that are quantified by the inner product. On the other hand, by interpreting the representation from a statistical perspective, one relates to other work in the literature. Moreover, the statistical description characterizes which information can be detected by the spike train inner product. The applications of the given inner products for development of machine learning methods are demonstrated in two problems, showing unsupervised and supervised learning.

[*]Chapter 8 in: "*Statistical Signal Processing for Neuroscience and Neurotechnology*", Karim G. Oweiss ed., Academic Press, 2010.

Contents

1 Introduction

Spike trains are a representation of neural activity. In neurophysiological studies, spike trains are obtained by detecting intra or extra cellularly the action potentials (i.e., "spikes"), but preserving only the time instant at which they occur. By neglecting the stereotypical shape of an action potential, spike trains contain an abstraction of the neurophysiological recordings, preserving only the spike times (i.e., the time at which action potentials occur) [Dayan and Abbott 2001]. Then, a spike train s is simply a sequence of ordered spike times $s = \{t_n \in \mathcal{T} : n = 1, \ldots, N\}$, with spike times in the interval $\mathcal{T} = [0, T]$ corresponding to the duration of the recording period. However, this abstraction poses a major problem in the way spike trains are manipulated. Indeed, put in this way, spike trains can be more readily modeled as realizations of point processes [Snyder 1975]. This means that any information, if present, is contained only in the location of the spike times. This is a very different perspective from the usual (functional) random processes, where the information is coded in the amplitude of the signal. For this reason, the usual operations of filtering, classification, and clustering, are not directly applicable to spike trains.

A commonly used methodology for spike train analysis and processing starts by converting the spike train to its *binned* counterpart and proceed as with functional random processes. The binned spike train is obtained by sliding a window in constant intervals over the spike train and counting the number of spikes within the window [Dayan and Abbott 2001]. The fundamental property of this transformation is that it maps randomness in time (expressed in the spike times) to randomness in amplitude of a discrete-time random process. Hence, after binning the spike trains, statistical analysis and machine learning methods can be applied as usual, making this methodology very straightforward and thus highly attractive. On the other hand, the impact of the *lossy* binning transformation must be taken into account. Note that, by binning, information about the exact spike times is lost, because an implicit time quantization takes place (with quantization step equal to the width of the sliding window). Therefore, any subsequent analysis will have limited accuracy, as demonstrated, for example, by Park et al. [2008] while comparing the usual cross-correlogram to an approach that avoids this quantization. This loss of information is especially concerning in studies focusing on the temporal dynamics because differences in spike timing smaller than the binning step will likely not be detected.

An alternative methodology is to utilize machine learning tools using a statistical characterization of the point process model. In concept, this methodology is very similar to the use of Bayesian methods in machine learning [Jordan 1998, Jensen 2001]. This approach is supported by the extensive characterizations of point processes in the statistical literature [Snyder 1975, Daley and Vere-Jones 1988, Karr 1986, Reiss 1993], but their analysis from realizations (such as spike trains) typically requires knowledge or the assumption of an underlying model [Brown et al. 2001, Eden et al. 2004, Barbieri et al. 2001]. Thus, the result is inaccurate if the model chosen is incomplete (i.e., if it is not capable of fully describing time-dependencies in the spike train). To characterize spike trains from more general models the standard approach has been to utilize generalized linear models (GLM) [Truccolo et al. 2005, Kass et al. 2005], which have many parameters and require large amounts of data for parameter estimation. Hence, GLMs are not appropriate for analysis of spike train data from a single-trial. More fundamentally, statistical approaches do not provide an effective way to handle multiple spike trains. To avoid handling high-dimensional joint distribution, statistical approaches systematically assume the spike trains to be independent which neglects potentially synergistic information [Narayanan et al. 2005, Schneidman et al. 2003]. Finally, using a statistical characterize directly, general machine learning methods are not easily developed and are very computationally expensive unless Gaussian approximations are used [Wang et al. 2009].

A number of other spike train methods are available, such as, frequency methods, Volterra series modeling, and spike train distance measures. However, these approaches are limited in scope and in the class of problems they are capable of tackling, whereas we are interested in developing a framework for general machine learning. Frequency-based methods [Pesaran et al. 2002, Baccalá and Sameshima 1999, Hurtado et al. 2004] can be useful for analysis of the frequency contents, to search for repetitive patterns, and phase synchrony, but they do not directly allow machine learning. Moreover, stationarity must be assumed, at least piecewise, which limits the dynamics which the frequency decomposition can expose. Volterra series [Marmarelis 2004, Song et al. 2007] have been utilized to model neural systems by expressing the system into a series of "kernels" which combine the input spike train(s) at multiple time instants. Intuitively, the Volterra series is the generalization of the Taylor series to systems with memory. The Volterra series decomposition can be utilized to reproduce a neural process or study a complex system by analyzing the (simpler) "kernels" [Marmarelis 2004]. However, it cannot be utilized for general machine learning since, for example, its intrinsic supervised formulation prevent its use for any unsupervised learning method. Another recent approach has been based on spike train measures. Some of the most widely utilized distance measures include Victor-Purpura's distance [Victor and Purpura 1996, 1997] and van Rossum's distance [van Rossum 2001]. A key advantage of using spike train distances is that they avoid the problems with binning by operating with the spike times directly. Moreover, several problems can be posed and solved resorting only to distances. On the other hand, distances do not provide the necessary "mathematical structure" for general machine learning. Most methods require other operations, such as the ability to compute means and projections, which are not possible with distance alone.

In this chapter we present a general framework to develop analysis and machine learning methods for spike trains. The core concept of the framework is the definition of inner product operators for spike trains. By defining inner product operators for spike trains, we build on the mathematical theory of reproducing kernel Hilbert spaces (RKHS) and the ideas from kernel methods, thus allowing a multitude of analysis and learning algorithms to be easily developed. The importance of RKHS theory and the corresponding induced inner products has been shown in a number of problems, such as statistical signal processing and detection [Parzen 1959, Kailath 1971, Kailath and Duttweiler 1972]. Moreover, inner products in RKHS spaces are the fundamental construct behind kernel machine learning [Schölkopf et al. 1999, Vapnik 1995, Wahba 1990]. However, rather that tacitly choosing these kernels as is typically done in kernel methods [Schölkopf et al. 1999], inner products will be defined here from functional representations of spike trains. We motivate the functional representations of spike trains from two perspectives: as a biological-modeling problem, and as a statistical description. The biological-modeling approach highlights the potential biological mechanisms taking place at the neuron level and which are quantified by the inner product. On the other hand, by interpreting the representation from a statistical perspective, one relates to other work in the literature. Hence, for spike trains, both of these perspectives are closely related. One of the most interesting consequences of this observation is that it exposes the advantages and limitations of some neural models.

It must be noted that the applications of this framework are not restricted to neural spike trains. Recently, there has also been a great interest in using spike trains directly for biologically inspired computation paradigms, rather than the more traditional firing rate network models [Dayan and Abbott 2001]. Examples of such paradigms are the liquid-state machine computational models [Maass et al. 2002, Maass and Bishop 1998], spiking neural networks [Bohte et al. 2002, Maass and Bishop 1998] or neuromorphic hardware design [Jabri et al. 1996]. In all of these applications, and regardless of the nature of the process generating the spike trains, there is the need to analyze and decode the information expressed by spike trains.

2 Functional representations of spike trains

Two complementary forms of functional representations of spike trains are presented in this section. By functional representation we are referring to a function of time that characterizes the spike train. These functional representations will be utilized later in defining inner products for spike trains. By first introducing these representations, we aim to (i) show the connection between neuron modeling and statistical approaches, and (ii) make clear how the inner product definitions in section 3 reflect synapse processing dynamics and statistical knowledge.

2.1 Synaptic models

In spite of the 'digital' characteristics embodied in spike trains, neurons are analog entities. Hence, some form of "spike-to-analog" conversion must take place in the synapses which converts spike trains into a functional form. Indeed, a spike propagated and arriving at the axon terminal will trigger the release of neurotransmitters through the synaptic cleft onto to receptors in the dendrite of the efferent neuron [Dowling 1992]. This induces an increase of the dendrite's conductance, which allows the transfer of ions between the surrounding and the dendrite, resulting in the post-synaptic potential. Immediately after, the neurotransmitters unbind and are metabolized, and the difference in potential between the dendrite and the surrounding makes ions flow towards the base potential, terminating the post-synaptic potential.

If the diffusion of neurotransmitters and the induced increase in conductance is assumed instantaneous, this process can be described through a very simple model. A spike arriving at the axon terminal results in the increase in conductance g,

$$g \rightarrow g + \Delta g, \tag{1a}$$

where the constant Δg denotes the increase in conductance. Then, the unbinding of neurotransmitters leads to a decrease in conductance,

$$\tau_s \frac{dg}{dt} = -g, \tag{1b}$$

considering a decrease controlled by a constant membrane time constant τ_s. Similarly, the model can be written in terms of the membrane potential.

For a spike train, the dynamics expressed by equations (1) can be written quite simply. If the input spike train is written as

$$s = \sum_{i=1}^{N} \delta(t - t_i), \tag{2}$$

with $\delta(\cdot)$ the Dirac delta impulse function, then the post-synaptic potential can be written as

$$v(t) = \sum_{i=1}^{N} h(t - t_i), \tag{3}$$

where h is the stereotypical evoked post-synaptic potential. With the parameters above, $h(t) = (\Delta g) \exp[-t/\tau_s] u(t)$, where $u(t)$ denotes the Heaviside step function.

It is interesting to verify that equation (3) is the same as the filtered spike train definition utilized in van Rossum [van Rossum 2001, equation (2.2)]. Then, van Rossum defines the distance between two filtered spike trains $v_i(t)$ and $v_j(t)$ as,

$$D(v_i, v_j) = \sqrt{\frac{1}{\tau_s} \int_0^\infty [v_i(t) - v_j(t)]^2 dt}. \tag{4}$$

That is, van Rossum's distance is merely the Euclidean distance between the evoked post-synaptic potentials according to this simple model.

One of the fundamental limitations of the previous model is that it neglects the limited availability of receptors in the efferent dendrite. When two spikes arrive to a synapse within a short time interval (i.e., such that the conductance has not returned to zero), the second spike will induce a smaller change in the conductance because many of the receptors are already bound to neurotransmitters because of the previous spike. This observation can be inserted in the model by modifying equation (1a) to

$$g \to g + \frac{\Delta g}{g_{\max}} (g_{\max} - g), \tag{5}$$

where $g_{\max}$ is the maximum value that the conductance can attain. The equation expressing the unbinding, equation (1b), remains the same. Equation (5) means that, for small conductance values, the conductance is increased by $\approx \Delta g$, but the increase saturates as g approaches $g_{\max}$. Put differently, equation (5) behaves much like equation (1a) followed by a nonlinear function, such as the hyperbolic tangent, for example. Hence, equation (5) can be approximated as

$$g \to f(g + \Delta g), \tag{6}$$

where f denotes a nonlinear function, appropriately scaled by $g_{\max}$. For example, both $f(x) = g_{\max} \tanh(x/g_{\max})$ or $f(x) = g_{\max}(1 - \exp(-x^2/(2g_{\max}^2)))$ are valid possibilities. In fact, any monotonic increasing function in $\mathbb{R}^+$, that is zero at zero and saturates as the argument goes towards infinity, could potentially be utilized under the appropriate scaling. Actually, nonlinear "sigmoid" functions which verify these requirements have been measured in neurophysiological recordings [Freeman 1975]. Correspondingly, the post-synaptic potential can be written as,

$$v^\dagger(t) = f\left(\sum_{i=1}^{N} h(t - t_i)\right), \tag{7}$$

affected by the same nonlinear function.[1] Note that the argument of the nonlinearity f equals the evoked membrane potential given by the previous model (cf. equation (3)). An example of the evoked membrane potentials by the different models is given in figure 1.

[1]Actually, the above equation is only an approximation to the model characterized by equation (6) since the nonlinearity also shapes the dynamics of the unbinding (equation (1b)), and the actual dynamics will depend on the exact choise of f.

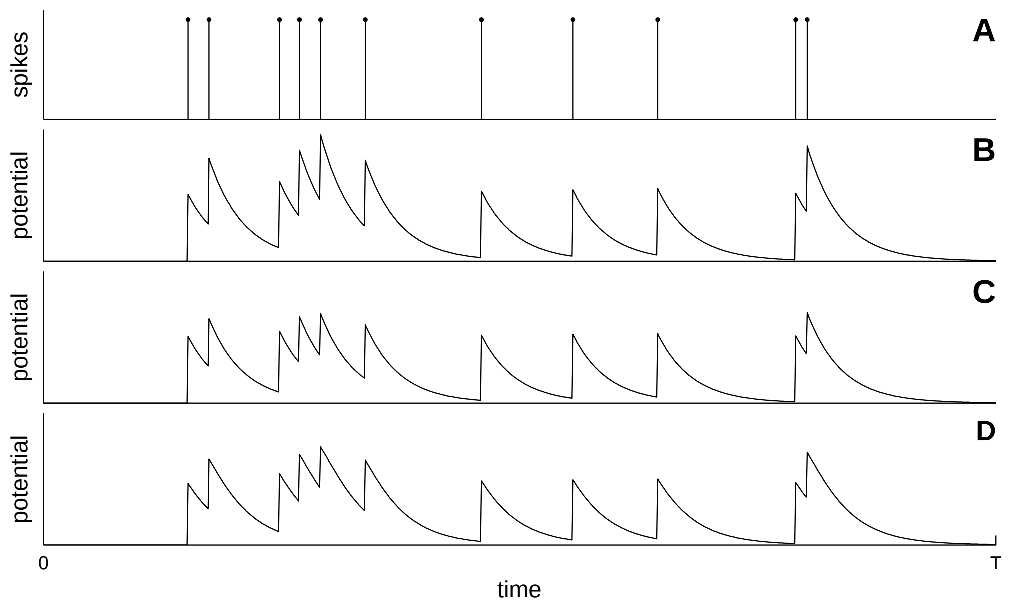


Figure 1: Comparison of evoked membrane potentials with different synapse models. The input spike train to the synapse model is shown in (A). The potentials in (B) through (D) correspond, respectively, to the linear model (equation (3)), nonlinear model (potential corresponding to the conductance given by equation (5)), and nonlinear model through nonlinear transformation (equation (7)). The same scale was used for the y-axis in (B) through (D).

Obviously, more complex synaptic models can be formulated (see for example Gerstner and Kistler [2002] for an extensive review). A common modification is to account for the diffusion time of the neurotransmitters across the synapse and their binding with the dendrite receptors. Similarly, the model can be augmented to account for the depletion of neurotransmitters in the axon terminal [Tsodyks and Markram 1997].

Overall, these synaptic models aim to emulate the filtering process that occurs at the synapses. In its simpler form, such as shown by equation 3, and even if the diffusion time is taken into account, the filter is linear. In modeling the depletion of neurotransmitters and/or receptors however, the filter becomes nonlinear. A very important observation is that nonlinear mechanisms incorporate a dependency on previous spikes, whereas a linear filter expresses only their superposition. In other words, the output of a nonlinear filter can reflect interactions in time between spikes. This observation can be made explicit, for example, if one substitutes f in equation (7) by its Taylor series expansion and develops the expression, which reveals that the post-synaptic potential depends on interactions between the smoothed spikes (see Appendix A for details).

As in van Rossum [2001], functional Euclidean distances can also be defined between the post-synaptic potentials evoked according to any of these models. In spike trains from zebra finch recordings, Houghton [2009] showed an improvement in classification performance when more complex models where used. The improvement was particularly significant when accounting for the depletion of receptors, which suggests the importance of multiple interacting spike trains.

2.2 Intensity estimation

Although spike trains are sets of discrete events in time, the underlying point process has a natural functional representation: the intensity function, or more generally the *conditional* intensity function. The conditional intensity function is a powerful functional descriptor because, rather than only explaining the spike train per se, it aims at describing the statistics of the underlying process which is more closely tied to the information the spike train encodes.

The conditional intensity function is defined for $t \in \mathcal{T} = (0, T]$ as

$$\lambda_s(t|H_t) = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \Pr\left[N(t+\epsilon) - N(t) = 1 | H_t\right], \tag{8}$$

where $N(t)$ is the counting process (i.e., the number of spikes in the interval $(0, t]$), and H_t is the history, up to time t, of the point process realization and any other random variables or processes the point process depends on. The conditional intensity function fully characterizes a point process because it describes the probability of an event (i.e., a spike), given its past.

Estimation of a general conditional intensity function is very difficult, if not impossible, because the definition in equation (8) can accommodate an infinite number of dependencies. Even if this is number is small, one has to estimate how each parameter affects the probability of an event for any time in the future, and how these influences may change due to correlations in the parameters.

Nevertheless, one can attempt to estimate the conditional intensity function by constraining its functional form. An example is the statistical estimation framework recently proposed by Truccolo et al. [2005]. Briefly, the spike train is first represented as a realization of Bernoulli random process (i.e., a discrete-time random process with 0/1 samples), and then a generalized linear model (GLM) is used to fit a conditional intensity function to the spike train. The GLM considers that the logarithm of the conditional intensity function has the form [Truccolo et al. 2005],

$$\log \lambda_{s_i}(\hat{t}_n | H_n^i) = \sum_{m=1}^{q} \theta_m g_m(\nu_m(\hat{t}_n)), \tag{9}$$

where $\hat{t}_n$ is the nth discrete-time instant, $\{g_m\}$ are general linear transformations of independent functions $\{\nu_m(\cdot)\}$, $\{\theta_m\}$ are the parameters of the GLM and q is the number of parameters. The terms $\{g_m(\nu_m(\hat{t}_n))\}$ are called the predictor variables in the GLM framework and, if one considers the conditional intensity to depend linearly on the spiking history, then the $\{g_m\}$ can be simply delays. In general the intensity can depend nonlinearly on the history or external factors such as stimuli. Other functional forms of the log conditional intensity function could also be utilized. Examples are the inhomogeneous Markov interval (IMI) process model proposed by Kass and Ventura [2001], and the point process filter proposed by Brown et al. [2001].

These approaches however have a main drawback: for practical use, the functional form of the conditional intensity function must be heavily constrained to only a few parameters, or long spike trains must be available for estimation of the parameters. The first approach has been utilized by Brown et al. [2001]. Conversely, in the framework proposed by Truccolo et al. [2005], a large number of parameters need to be estimated since q must be larger that the average inter-spike interval.

The estimation of the conditional intensity function is greatly simplified if an (inhomogeneous) Poisson process model is assumed. This is because there is no history dependence. Hence, in this case the conditional intensity function is only a function of time, $\lambda_s(t)$, and is called the intensity function. Several methods exist in the point process literature for intensity function estimation. The obvious approach is to divide the duration of the spike train $\mathcal{S}(\mathcal{T}) = [0, T]$ in small intervals and average the number of spikes in each interval over realizations. In essence, binning aims at exactly this, without the average over realizations. A common alternative is kernel smoothing [Dayan and Abbott 2001, Reiss 1993, Richmond et al. 1990], which is much more data efficient, as demonstrated by Kass et al. [2003]. Given a spike train s comprising of spike times $\{t_m \in \mathcal{T} : m = 1, \ldots, N\}$, the estimated intensity function is

$$\hat{\lambda}_{s_i}(t) = \sum_{m=1}^{N_i} h(t - t_m^i), \tag{10}$$

where h is the smoothing function. This function must be non-negative and integrate to one over the real line (just like a probability density function (pdf)). Commonly used smoothing functions are the Gaussian, Laplacian and α-function, among others. Note that binning is nothing but a special case of this procedure in which h is a rectangular window and the spike times are first quantized according to the width of the rectangular window [Dayan and Abbott 2001].

Comparing equations (10) and (3), it is clear the resemblance between them. However, based on the context they where introduced, those equations provide two seemingly very different perspectives. Equation (3) aims to describe the evoked membrane potential due to a spike. On the other hand, equation (10) suggests this mechanism is equivalent to intensity estimation. Conversely, it is possible to think statistically of the former synapse model as an approximation to the expectation of the dendrite response to a spike train. Perhaps more importantly, the connection between these equations proves that the synapse model described by equation (3) is limited to Poisson spike trains. In other words, this model is memoryless. Thus, potential information contained in interactions between spikes in a spike train will be neglected. In contrast, the model characterized by equation (7) has memory due to the nonlinearity. An mentioned in the previous section and shown in Appendix A, the nonlinear function introduces higher-order correlations in the smoothed spike train obtained from the linear model.

In summary, although spike train's content could be decoded using a statistical characterization, there is a severe trade-off between complexity of the model and inference, and its estimation ability. For these reasons, this is not the goal of this chapter. Instead, we build a general framework for machine learning using inner products. A fundamental reason to define inner products as spike train operators is that, we do not need an *explicit* statistical characterization. Instead, we need only to ensure that the inner product detects the *implicit* statistical similarities between two spike trains.

3 Inner products for spike trains

As mentioned in the introduction, because spike trains are realizations of point processes, it is not straightforward to devise machine learning algorithms to operate with spike trains. In this section we show that inner products are fundamental operators for signal processing, and their definition for spike trains provides the necessary structure for machine learning.

Using a few signal processing and machine learning applications as examples, let us demonstrate why are inner products so important (Figure 2). In filtering, the output is the convolution

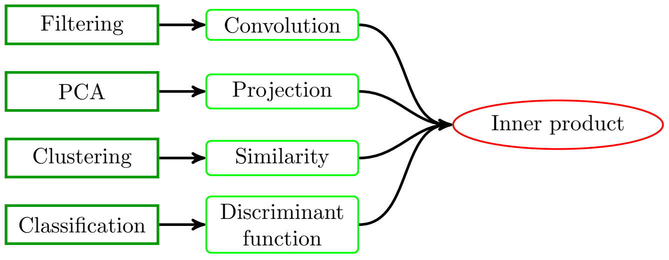


Figure 2: Inner products are fundamental operators for signal processing and machine learning.

of the input with an impulse response; for principal component analysis (PCA), one needs to be able to project the data; for clustering, the concept of similarity (or dissimilarity) between points is needed; and for classification, it is necessary to define discriminant functions that separate the classes. In all of these applications, the operators needed are either implemented directly by an inner product or can be constructed with an inner product. Convolution implements an inner product at each time instant between a mirrored and shifted version of the input function and the systems' impulse response. Projection is by definition an inner product between two objects. An inner product is also a similarity measure, and dissimilarity measures (such as distances) can be defined given an inner product. Discriminant functions cannot be obtained directly with an inner product, but they can be nonlinearly combined to approximate it to the desired precision, as happens for example in neural networks where the linear projections in the processing elements are implemented by inner products.

Many more examples can be shown to rely entirely on inner products, or that can be formulated and solved using only inner products. In fact, using inner products to solve general machine learning problems is the enabling idea behind kernel methods [Schölkopf et al. 1999]. There are many studies in the literature that show how this idea can be utilized to statistical signal processing [Parzen 1959, Kailath 1971, Kailath and Weinert 1975, Kailath and Duttweiler 1972], smoothing and spline methods [Wahba 1990], classification [Vapnik 1995, Schölkopf et al. 1999], and many more. Similarly, what we propose is to establish such a general framework for spike train signal processing and pattern recognition by defining inner product operators.

On the other hand, there has been much work recently on spike train distance measures [Victor 2002, van Rossum 2001, Houghton and Sen 2008]. Hence, a natural question is: Are spike train metrics sufficient for general machine learning with spike trains? The answer is no. The reason for this is that most of machine learning methods require the mathematical structure of a vector space, which is not provided by a metric, but implicitly attained with the definition of an inner product. For example, it is not possible to cluster spike trains using a k-means-like algorithm with only a distance because there is no way to compute the center points (the "means" in k-means). However, computing the mean is trivial in a inner product (vector) space. Moreover, the definition of spike train inner products naturally induces at least two spike train metrics, as shown later. Clearly, it is possible to define distances without having an inner product associated with it (see, for example, Victor and Purpura [1996], Kreuz et al. [2007], Christen et al. [2006]). Yet, in addition to limitations just discussed, these approaches tend to obscure the assumed point process model.

3.1 Defining inner products for spike trains

In this section we discuss how to define inner products for spike trains, and the advantages and limitations of each approach. There are two main approaches to define inner products for spike trains. One possibility is to start by defining an inner product for spike times, and then utilize the linearity in the inner product space and the fact that spike trains are sets of spike times to extend the inner product to spike trains [Carnell and Richardson 2005, Schrauwen and Campenhout 2007, Paiva et al. 2009]. An alternative methodology is to define inner products on functional representations of spike trains [Paiva et al. 2009]. Both of these methodologies are discussed next.

3.1.1 Inner products for spike times

In the first approach there are two key design elements: the definition or choice of the inner product between spike times, and how these are combined to obtain the final spike train inner product. Often, the inner product for spike times is simply set by choosing a symmetric and positive definite kernel to operate on spike times [Paiva et al. 2009, Carnell and Richardson 2005, Schrauwen and Campenhout 2007]. Using RKHS theory (see appendix C for a brief introduction), it immediately follows that a kernel obeying these conditions will induce a reproducing kernel Hilbert space, for which the kernel represents the evaluation of the inner product [Aronszajn 1950]. Conceptually, these kernels operate much in the same way as the kernels operating on data samples in machine learning [Schölkopf et al. 1999]. Then, depending on the application, the spike time kernels can be combined differently to capture information that matches ones' understanding or a priori knowledge about the problem. For example, defining the inner product between two spikes as,

$$\langle \delta(t-t_m), \delta(t-t_n) \rangle = \exp\left(-\frac{|t_m - t_n|}{\tau}\right) = \kappa(t_m, t_n), \tag{11}$$

and using the fact that a spike train can be written as a sum of spikes (equation 2), one obtains the spike train inner product [Carnell and Richardson 2005],

$$\begin{aligned} K(s_i, s_j) &= \langle s_i, s_j \rangle \\ &= \left\langle \sum_{m=1}^{N_i} \delta(t - t_m^i), \sum_{n=1}^{N_j} \delta(t - t_n^j) \right\rangle \\ &= \sum_{m=1}^{N_i} \sum_{n=1}^{N_j} \langle \delta(t - t_m^i), \delta(t - t_n^j) \rangle \\ &= \sum_{m=1}^{N_i} \sum_{n=1}^{N_j} \kappa(t_m^i, t_n^j). \end{aligned} \tag{12}$$

Another example is the kernel,

$$K^*(s_i, s_j) = \begin{cases} \displaystyle\max_{l=0,1,\ldots,(N_i-N_j)} \sum_{n=1}^{N_j} \kappa(t_{n+l}^i - t_n^j), & N_i \geq N_j \\ \displaystyle\max_{l=0,1,\ldots,(N_j-N_i)} \sum_{n=1}^{N_i} \kappa(t_n^i - t_{n+l}^j), & N_i < N_j. \end{cases} \tag{13}$$

which quantifies the presence of related spike patterns between spike trains s_i, s_j [Chi and Margoliash 2001, Chi et al. 2007]. Since this kernel measures if spike trains have a one-to-one correspondence of a given sequence of spike times, it could be utilized, for example, to study temporal precision and reliability in neural spike trains in response to stimulus, or detect/classify these stimuli.

The prime advantage of this approach is that it allows for spike train inner products to be easily designed for a specific application by utilizing available a priori knowledge about the problem, as the second example illustrates. On the hand, this is also a disadvantage. If the inner product is designed without this information, as in the first example, then it is not easy to determine what the inner product is quantifying. More fundamentally, it is unclear what is the implicitly assumed point process model or if the inner product has any biological interpretation. From this point of view, the methodology presented next is clearly more advantageous.

3.1.2 Functional representations of spike trains

Rather than defining the spike train inner products directly, in the following we define inner products on functional representations of spike trains. This bottom-up construction from fundamental data descriptors is unlike the previously mentioned approach and is rarely taken in machine learning, but it explicitly shows the implied synaptic model and/or the assumed point process model. Hence, this explicit construction is very informative about the inner product and derived constructs, since it clearly exposes the limitations of a given definition. This knowledge can be of paramount importance in both theoretical and neurophysiological studies, because the outcome of these studies depends on the implied model and the expressive power of the measures utilized to quantify it.

Consider two spike trains, $s_i, s_j \in \mathcal{S}(\mathcal{T})$, and denote their corresponding evoked membrane potentials by $v_{s_i}(t)$ and $v_{s_j}(t)$, with $t \in \mathcal{T}$. For spike trains with finite duration (i.e., $t \in \mathcal{T}$, with $\mathcal{T}$ any finite interval), the membrane potentials are bounded, and therefore

$$\int_{\mathcal{T}} v_{s_i}^2(t)dt < \infty. \tag{14}$$

That is, membrane potentials are square integral functions in $\mathcal{T}$. Hence, $v_{s_i}(t), v_{s_j}(t) \in L_2(\mathcal{T})$. Consequently, an inner product for spike trains can be taken as the usual inner product of membrane potentials in $L_2(\mathcal{T})$ space,

$$V(s_i, s_j) = \left\langle v_{s_i}(t), v_{s_j}(t) \right\rangle_{L_2(\mathcal{T})} = \int_{\mathcal{T}} v_{s_i}(t) v_{s_j}(t)dt. \tag{15}$$

Clearly, the inner product can only quantify spike train characteristics expressed in the membrane potential. That is, the limitations of the inner product are set by the chosen synapse model.

Similarly, inner products for spike trains can be easily defined from their statistical descriptors. Considering spike trains s_i and s_j and the conditional intensity functions of the underlying point processes, denoted by $\lambda_{s_i}(t|H_t^i)$ and $\lambda_{s_j}(t|H_t^j)$, the inner product can be defined as

$$I(s_i, s_j) = \left\langle \lambda_{s_i}(t|H_t^i), \lambda_{s_j}(t|H_t^j) \right\rangle_{L_2(\mathcal{T})} = \int_{\mathcal{T}} \lambda_{s_i}(t|H_t^i)\lambda_{s_j}(t|H_t^j)dt, \tag{16}$$

where, as before, the usual inner product in $L_2(\mathcal{T})$ was utilized. Note that, just like membrane potentials, conditional intensity functions are square integrable in a finite interval. This is the *cross-intensity* (CI) kernel defined in Paiva et al. [2009]. The advantage in defining the inner product from the conditional intensity functions is that the resulting *inner product incorporates the statistics of the point processes directly.* Moreover, in the general form given, this inner product can be utilized for *any* point process model since the conditional intensity function is a complete characterization of the point process [Cox and Isham 1980, Daley and Vere-Jones 1988].

Both of these definitions are very general but, in practice, some particular form must be considered. Some of these particular cases are presented and discussed next.

For the simplest synaptic model presented, with membrane potential given by equation (3), it is possible to evaluate the inner product without explicit computation of the integral. Substituting equation (3) in the inner product definition (equation (15)), and using the linearity of the integral yields,

$$V(s_i, s_j) = \sum_{m=1}^{N_i} \sum_{n=1}^{N_j} \int_{\mathcal{T}} h(t - t_m^i) h(t - t_n^j) dt = \sum_{m=1}^{N_i} \sum_{n=1}^{N_j} k(t_m^i, t_n^j) dt, \tag{17}$$

where k is the autocorrelation of h. As remarked in the end of section 2.2, it is possible to reason about the membrane potential in equation (3) as estimation of the intensity function through kernel smoothing (equation (10)). Indeed, if one considers the cross-intensity inner product (equation (16)) constrained to inhomogeneous Poisson processes and substitutes the kernel smoothing intensity estimator in equation (10), we obtain the result in equation (17). This shows that the inner product in equation (17) is limited to Poisson point processes, and will fail to distinguish between spike trains with the same intensity function even if they are were generated by two different point processes. For this reason, this inner product was called the *memoryless cross-intensity* (mCI) kernel in Paiva et al. [2009]. It is noteworthy that in this case the inner product also matches the definition obtained directly from spike times, shown in equation (12). However, the meaning of the inner product, what it quantifies, and its limitations, are now easily understood in light of the latter derivation.

The limitations of the previous inner product are easily verified, both from its implied synapse model and its assumed point process. A more expressive inner product can be obtained if the synapse model with saturation is utilized instead. The inner product takes the same general form of equation (15) but with the membrane potential now given by equation (7), and can be written as,

$$V^{\dagger}(s_i, s_j) = \int_{\mathcal{T}} v_{s_i}^{\dagger}(t) v_{s_j}^{\dagger}(t) dt, = \int_{\mathcal{T}} f\left(v_{s_i}(t)\right) f\left(v_{s_j}(t)\right) dt, \tag{18}$$

where $v_{s_i}(t), v_{s_j}(t)$ denotes the potentials as one obtains from the linear synapse model (equation (3)). Equivalently, the dual interpretation of the potentials $v_{s_i}(t), v_{s_j}(t)$ as intensity functions of inhomogeneous Poisson processes can be utilized to write the above inner product as,

$$V^{\dagger}(s_i, s_j) = \int_{\mathcal{T}} f\left(\lambda_{s_i}(t)\right) f\left(\lambda_{s_j}(t)\right) dt. \tag{19}$$

This inner product is conceptually similar to the *nonlinear cross-intensity* (nCI) kernel defined by Paiva et al. [2009],

$$I_{\sigma}^{\dagger}(s_i, s_j) = \int_{\mathcal{T}} \mathcal{K}_{\sigma}\left(\lambda_{s_i}(t), \lambda_{s_j}(t)\right) dt, \tag{20}$$

where $\mathcal{K}_{\sigma}$ is a symmetric positive definite kernel with kernel size parameter σ. The difference is that in equation (19) the nonlinear coupling of the intensity functions is shown explicitly, whereas in the nCI kernel it merely implied by $\mathcal{K}$. Hence, an advantage of the derivation given here is that it explicitly shows the implied synapse model, which supports the plausibility and significance of the inner product shown from spike train signal processing. Moreover, from RKHS theory it is known that $\mathcal{K}$ denotes an inner product of the transformed linear membrane potentials, but the transformation is usually unknown. Taking the definition in equation (19), we could define $\mathcal{K}(x, y) = f(x)f(y)$ which connects the two definitions, and gives explicitly the nonlinear transformation into the inner product space.

As remarked in sections 2.1 and 2.2, an important characteristic of the synapse model with binding saturation (equation (7)) is the dependence on higher-order interactions between multiple spikes in time. That is, the nonlinearity introduces memory. Statistically, this means that, if kernel smoothing is used as the intensity function estimator in equation (19), this inner product is sensitive to point processes with memory. This statement has been verified empirically for the nCI [Paiva et al. 2009]. In theory, depending on the nonlinearity utilized the memory depth of the inner product can be infinity. However, in practice, the decaying dynamics of the conductance after a spike and the saturation in the nonlinearity forces the influence of past spikes to die out. Moreover, this means that, because of the decreasing weighting of the higher-order spike interactions, the inner product will loose specificity as the memory requirements to distinguish spike trains from different point processes increases.

As mentioned earlier, in its general form, $I(s_i, s_j)$ (equation (16)) is an optimal inner product since the conditional intensity functions completely characterize the underlying point process. On the hand, as discussed in section 2.2, estimation of general conditional intensity functions is not feasible in practice and even though their estimation under constrained scenarios is troublesome. Although both $V^\dagger$ and $I^\dagger$ also have constrained memory dependencies, their estimation is very simple and offers a shortcut around the current difficulties in the estimation of conditional intensity functions, providing a trade-off between memory depth for specificity.

3.2 Properties of the defined inner products

In this section some properties of the inner products defined in the previous section are presented. The main purpose of the properties shown is to prove the defined inner products are well defined and have an associated reproducing kernel Hilbert space (RKHS) which provides the necessary mathematical structure for machine learning.

Property 1. *The inner products in equations* (15) *and* (16) *are:*

i. Symmetric;

ii. Linear operators in the space of the evoked membrane potentials or intensity functions, respectively;

iii. The inner product of a spike train with itself is non-negative and zero if and only if the spike train is empty (i.e., has no spikes).

This property asserts that the proposed definitions verify the axioms of an inner product. Actually, because the defined inner products operate on elements of $L_2(\mathcal{T})$ and correspond to the usual dot product in that space, this property is a direct consequence of the properties inherited from L_2.

Property 2. *The inner products in equations* (15) *and* (16) *correspond to symmetric positive definite kernels in spike train space.*

The proof is given in appendix B. Through the work of Moore [1916] and due to the Moore-Aronszajn theorem [Aronszajn 1950], the following two properties result as corollaries.

Property 3. *For any set of n spike trains ($n \geq 1$), the inner product matrix, with the ijth element given by inner product of spike trains s_i and s_j, is symmetric and non-negative definite.*

Property 4. *There exists an Hilbert space for which the defined spike trains inner products is a reproducing kernel.*

Properties 2 through 4 are equivalent in the sense that any of these properties implies the other two. To verify this note that, by definition, property 3 is merely a restatement of property 2,[2] and a kernel being symmetric and positive definite is a necessary and sufficient condition to property 4 [Aronszajn 1950].

An important consequence of these properties, explicitly stated in property 4, is the existence of a unique RKHS associated with each of the inner products defined earlier. This is of importance because it ensures the existence of a complete metric space,[3] and it abstracts the design of machine learning methods from the exact inner product definition.

Property 5. *The inner products in equations* (15) *and* (16) *verify the Cauchy-Schwarz inequality,*

$$\begin{aligned} V^2(s_i, s_j) &\leq V(s_i, s_i)V(s_j, s_j), \textit{ and} \\ I^2(s_i, s_j) &\leq I(s_i, s_i)I(s_j, s_j), \end{aligned} \tag{21}$$

for any two spike trains $s_i, s_j \in \mathcal{S}(\mathcal{T})$.

As before, the proof is given in appendix B. The Cauchy-Schwarz inequality will be shown to lead to a spike train metric in the next section.

Property 6. *The inner products defined on functions in $L_2(\mathcal{T})$, using either membrane potentials or conditional intensity functions, are congruent*[4] *to the corresponding inner product in the RKHS.*

This property is a direct consequence of the approach we used to define the inner product. In other words, in our case we defined the inner product between spike trains as the inner product of their functional representations in $L_2(\mathcal{T})$. Hence, by definition, that inner product establishes the existence of the RKHS and its inner product, making them equal. See Paiva et al. [2009, section 5.1] for further details.

[2] Note that we are considering a "positive definite" kernel in the sense of Aronszajn [1950] and Parzen [1959]. That is, a kernel (i.e., function of two arguments), $\kappa : X \times X \to \mathbb{R}$, is positive definite if and only if,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \kappa(x_i, x_j) \geq 0,$$

for any $n > 0$, $x_1, \ldots, x_n \in X$ and $a_1, \ldots, a_n \in \mathbb{R}$.

[3] A complete metric space is a metric space in which every Cauchy sequence is convergent. The metric space is naturally induced with the inner product definition. Basically, this ensures that any sequence of points and their limit is contained in the space.

[4] See appendix C for the definition of congruent inner products.

3.3 Distances induced by inner products

Metrics, and distances in particular, can be very useful in classification and analysis of data. Spike trains are no exception. The importance of spike train metrics can be observed from the attention it has received in the literature [Victor and Purpura 1997, van Rossum 2001, Schreiber et al. 2003]. However, when used without further mathematical structure, their usefulness is limited. In this section we demonstrate that by first defining inner products we naturally induce two distances in the inner product space. More than presenting spike train distances, we aim to further support the argument that defining inner products is a principled way to build bottom-up the required structure for general machine learning.

Given an inner product space, (at least) two distances can be immediately defined. The first utilizes the fact that an inner product naturally induces a norm,[5] which in turn provides a distance between differences of elements. The second distance derives from the Cauchy-Schwarz inequality.

Consider the inner product between spike trains defined in equation 15. (The same idea applies directly for *any* spike train inner product.) The natural norm induced by this inner product is given by,

$$\left\| v_{s_i}(t) \right\| = \sqrt{\left\langle v_{s_i}(t), v_{s_i}(t) \right\rangle_{L_2(\mathcal{T})}} = \sqrt{V(s_i, s_j)}. \tag{22}$$

Hence, the *norm distance* between spike trains $s_i, s_j \in \mathcal{S}(\mathcal{T})$ is

$$d_{ND}(s_i, s_j) = \left\| v_{s_i} - v_{s_j} \right\| = \sqrt{\left\langle v_{s_i} - v_{s_j}, v_{s_i} - v_{s_j} \right\rangle} \tag{23a}$$

$$= \sqrt{\left\langle v_{s_i}, v_{s_i} \right\rangle - 2\left\langle v_{s_i}, v_{s_j} \right\rangle + \left\langle v_{s_j}, v_{s_j} \right\rangle} \tag{23b}$$

$$= \sqrt{V(s_i, s_i) - 2V(s_i, s_j) + V(s_j, s_j)}, \tag{23c}$$

where the linearity of the inner product was utilized from equation (23a) to (23b). Note that, even if we do not have an explicit functional decomposition of the inner product, as occurs for the inner products (or kernels) defined in section 3.1.1, as long as symmetry and positive definiteness holds the same idea applies. In this case, the inner products in equation (23) are simply the inner products in the induced RKHS.

An alternative distance can be obtained based on the Cauchy-Schwarz inequality (property 5). We can define the *Cauchy-Schwarz (CS) distance* between two spike trains as

$$d_{CS}(s_i, s_j) = \arccos \frac{V(s_i, s_j)^2}{V(s_i, s_i)V(s_j, s_j)}. \tag{24}$$

From the properties of norm and the Cauchy-Schwarz inequality it is easy to verify that either distance satisfies the three distance axioms: symmetry, positiveness, and the triangle inequality. Indeed, the norm distance is basically the generalization of the idea behind the Euclidean distance to a space of functions. On the other hand, the CS distance measures the "angular distance" between spike trains, as suggested by the usual dot product between two vectors $\vec{x}, \vec{y}$ written as $\vec{x} \cdot \vec{y} = \|\vec{x}\| \, \|\vec{y}\| \cos(\theta)$, with θ the angle between the two vectors. Hence, a major difference between the norm distance and the CS distance is that the latter is not an Euclidean measure but a Riemannian metric, which measures the geodesic distance between points in a curved space.

[5] One can also reason in terms of the norm in the RKHS, because an RKHS is an Hilbert space and, therefore, has a norm. In fact, the two norms are equivalent because of the congruence between the spaces [Paiva et al. 2009].

As observed by Houghton [2009], it is possible to define a van Rossum-like distance [van Rossum 2001] for each synaptic model. Clearly, taking the inner product in the definition of the norm distance to be the inner product between membrane potentials given by each synapse model, equation (15), one obtains the distances suggested by Houghton. In other words, van Rossum's distance as originaly defined [van Rossum 2001] is a particular case of the norm distance. More specifically, van Rossum's distance considers the inner product in equation (15), with the potential given by equation (3), using a causal decaying exponential function as the smoothing function h. Also, it had been noted before that other smoothing functions could be utilized [Schrauwen and Campenhout 2007]. However, the approach taken here requires only an inner product, thus being far more general. Moreover, the dual perspective in terms of conditional intensity functions clearly identifies that, regardless of the choice of smoothing function h, van Rossum's distance is sensitive at most to Poisson process statistics. On the other hand, using nonlinear synapse models one introduces memory into the inner product, explaining Houghton's results both neurophysiologically and statistically.

A spike train metric closely related to the CS distance had also been previously suggested in the literature. Indeed, the Cauchy-Schwarz distance can be compared with the "correlation measure" between spike trains proposed by Schreiber et al. [2003]. We can observe that the latter corresponds to the argument of the arc cosine and thus denotes the cosine of an angle between spike train. In the case of the metrics proposed by Schreiber et al., the norm and inner product where given by equation (15), with the potentials given by equation (3) using the Gaussian smoothing function. However, the metric suffers from the same limitation as van Rossum's distance since it implicitly assumes Poisson processes. Using our methodology the metric can be easily extended. Moreover, notice that Schreiber's et al. "correlation measure" is only a pre-metric since it does not verify the triangle inequality. In d_{CS} this is ensured by the arc cosine function.

4 Applications

To exemplify the importance of the developments shown here, in the following we derive two fundamental machine learning algorithms for spike trains: principal component analysis (PCA) and Fisher linear discriminant (FLD). These two algorithms serve also to demonstrate that the framework presented here allows the development of both supervised and unsupervised learning methods. The primary aim here is to demonstrate the ease with which machine learning algorithms for spike trains can be developed once the mathematical structure created by an inner product and the induced RKHS is established.

4.1 Unsupervised learning: principal component analysis

PCA is widely used for data analysis and dimensionality reduction [Diamantaras and Kung 1996]. PCA will be derived for spike trains directly in the RKHS induced by any of the inner products defined. This approach highlights that optimization with spike trains is possible by the definition of an inner product, and more specifically through the mathematical structure provided by the induced RKHS. This is also the traditional approach in the functional analysis literature [Ramsay and Silverman 1997] and has the advantage of being completely general, regardless of the spike train inner product.

4.1.1 Derivation

In the following, let P denote a spike train inner product inducing an RKHS $\mathcal{H}$, such as any of the inner products defined earlier. Consider a set of spike trains, $\{s_i \in \mathcal{S}(\mathcal{T}), i = 1, \ldots, N\}$, for which we wish to determine the principal components. Computing the principal components of the spike trains directly is not feasible because we would not know how to define a principal component (PC), however, this is a trivial task in the RKHS induced by a spike train inner product.

Let $\{\Lambda_{s_i} \in \mathcal{H}, i = 1, \ldots, N\}$ denote the set of elements in the RKHS $\mathcal{H}$ (associated with P) corresponding to the given spike trains. That is, the transformation or mapping of the spike trains into $\mathcal{H}$. The mean of the transformed spike trains is,

$$\bar{\Lambda} = \frac{1}{N} \sum_{i=1}^{N} \Lambda_{s_i}. \tag{25}$$

Thus, the centered transformed spike trains (i.e., with the mean removed) can be obtained as,

$$\tilde{\Lambda}_{s_i} = \Lambda_{s_i} - \bar{\Lambda}. \tag{26}$$

PCA finds an orthonormal transformation providing a compact description of the data. Determining the principal components of spike trains in the RKHS can be formulated as the problem of finding the set of orthonormal vectors in the RKHS such that the projection of the centered transformed spike trains $\{\tilde{\Lambda}_{s_i}\}$ has *maximum variance*. This means that the principal components can be obtained by solving an optimization problem in the RKHS. A function $\xi \in \mathcal{H}$ (i.e., $\xi : \mathcal{S}(\mathcal{T}) \longrightarrow \mathbb{R}$) is a principal component if it maximizes the cost function,

$$J(\xi) = \sum_{i=1}^{N} \left[\mathrm{Proj}_{\xi}(\tilde{\Lambda}_{s_i})\right]^2 - \rho \left(\|\xi\|^2 - 1\right) \tag{27}$$

where $\mathrm{Proj}_{\xi}(\tilde{\Lambda}_{s_i})$ denotes the projection of the ith centered transformed spike train onto ξ, and ρ is the Lagrange multiplier for the constraint $\left(\|\xi\|^2 - 1\right)$ imposing that the principal components have unit norm. To evaluate this cost function one needs to be able to compute the projection and the norm of the principal components. In an RKHS, an inner product is the projection operator and the norm is naturally defined (see equation (22)). Thus, the above cost function can be expressed as,

$$J(\xi) = \sum_{i=1}^{N} \left\langle \tilde{\Lambda}_{s_i}, \xi \right\rangle^2 - \rho \left(\langle \xi, \xi \rangle - 1\right), \tag{28}$$

In practice, we always have a finite number of spike trains. Hence, by the representer theorem [Kimeldorf and Wahba 1971, Schölkopf et al. 2001], ξ is restricted to the subspace spanned by the centered transformed spike trains $\{\tilde{\Lambda}_{s_i}\}$. Consequently, there exist coefficients $b_1, \ldots, b_N \in \mathbb{R}$ such that

$$\xi = \sum_{j=1}^{N} b_j \tilde{\Lambda}_{s_j} = \mathbf{b}^T \tilde{\mathbf{\Lambda}} \tag{29}$$

where $\mathbf{b}^T = [b_1, \ldots, b_N]$ and $\tilde{\mathbf{\Lambda}} = \left[\tilde{\Lambda}_{s_1}, \ldots, \tilde{\Lambda}_{s_N}\right]^T$. Substituting in equation (28) yields

$$\begin{aligned} J(\xi) &= \sum_{i=1}^{N} \left(\sum_{j=1}^{N} b_j \left\langle \tilde{\Lambda}_{s_i}, \tilde{\Lambda}_{s_j} \right\rangle \right) \left(\sum_{k=1}^{N} b_k \left\langle \tilde{\Lambda}_{s_i}, \tilde{\Lambda}_{s_k} \right\rangle \right) \\ &\quad + \rho \left(1 - \sum_{j=1}^{N} \sum_{k=1}^{N} b_j b_k \left\langle \tilde{\Lambda}_{s_i}, \tilde{\Lambda}_{s_k} \right\rangle \right) \\ &= \mathbf{b}^T \tilde{\mathbf{P}}^2 \mathbf{b} + \rho \left(1 - \mathbf{b}^T \tilde{\mathbf{P}} \mathbf{b} \right). \end{aligned} \tag{30}$$

where $\tilde{\mathbf{P}}$ is the inner product matrix of the centered spike trains; that is, the $N \times N$ matrix with elements

$$\begin{aligned} \tilde{\mathbf{P}}_{ij} &= \left\langle \tilde{\Lambda}_{s_i}, \tilde{\Lambda}_{s_j} \right\rangle = \left\langle \Lambda_{s_i} - \bar{\Lambda}, \Lambda_{s_j} - \bar{\Lambda} \right\rangle \\ &= \left\langle \Lambda_{s_i}, \Lambda_{s_j} \right\rangle - \frac{1}{N} \sum_{l=1}^{N} \left\langle \Lambda_{s_i}, \Lambda_{s_l} \right\rangle - \frac{1}{N} \sum_{l=1}^{N} \left\langle \Lambda_{s_l}, \Lambda_{s_j} \right\rangle + \frac{1}{N^2} \sum_{l=1}^{N} \sum_{n=1}^{N} \left\langle \Lambda_{s_l}, \Lambda_{s_n} \right\rangle \\ &= P(s_i, s_j) - \frac{1}{N} \sum_{l=1}^{N} P(s_i, s_l) - \frac{1}{N} \sum_{l=1}^{N} P(s_l, s_j) + \frac{1}{N^2} \sum_{l=1}^{N} \sum_{n=1}^{N} P(s_l, s_n). \end{aligned} \tag{31}$$

In matrix notation,

$$\tilde{\mathbf{P}} = \mathbf{P} - \frac{1}{N}(\mathbf{1}_N \mathbf{1}_N^T \mathbf{P} + \mathbf{P} \mathbf{1}_N \mathbf{1}_N^T) + \frac{1}{N^2} \mathbf{1}_N \mathbf{1}_N^T \mathbf{P} \mathbf{1}_N \mathbf{1}_N^T, \tag{32}$$

where $\mathbf{P}$ is the spike train inner product matrix, $\mathbf{P}_{ij} = \left\langle \Lambda_{s_i}, \Lambda_{s_j} \right\rangle = P(s_i, s_j)$, and $\mathbf{1}_N$ is the $N \times 1$ vector with all ones (such that $\mathbf{1}_N \mathbf{1}_N^T$ is a $N \times N$ matrix of ones). This means that $\tilde{\mathbf{P}}$ can be computed directly from $\mathbf{P}$ without the need to explicitly remove the mean of the transformed spike trains.

From equation (30), finding the principal components simplifies to the problem of estimating the coefficients $\{b_i\}$ that maximize $J(\xi)$. Since $J(\xi)$ is a quadratic function its extrema can be found by equating the gradient to zero. Taking the derivative with regards to $\mathbf{b}$ (which characterizes ξ) and setting it to zero results in

$$\frac{\partial J(\xi)}{\partial \mathbf{b}} = 2\tilde{\mathbf{P}}^2 \mathbf{b} - 2\rho \tilde{\mathbf{P}} \mathbf{b} = 0, \tag{33}$$

and thus corresponds to the usual eigendecomposition problem[6]

$$\tilde{\mathbf{P}} \mathbf{b} = \rho \mathbf{b}. \tag{34}$$

This means that any eigenvector of the centered inner product matrix is a solution of equation (33). Thus, the eigenvectors determine the coefficients of equation (29) and characterize the principal components. It is easy to verify that, as expected, the variance of the projections onto each

[6]Note that the simplification in the eigendecomposition problem is valid regardless if the inner product matrix is invertible or not, since $\tilde{\mathbf{P}}^2$ and $\tilde{\mathbf{P}}$ have the same eigenvectors and the eigenvalues of $\tilde{\mathbf{P}}^2$ are the eigenvalues of $\tilde{\mathbf{P}}$ squared.

principal component equals the corresponding eigenvalue squared. So, the ordering of ρ specifies the relevance of the principal components.

To compute the projection of a given input spike train s onto the kth principal component (corresponding to the eigenvector with the kth largest eigenvalue) we need only to compute the inner product of Λ_s with ξ_k in the RKHS. That is,

$$\begin{aligned}\mathrm{Proj}_{\xi_k}(\Lambda_s) &= \langle \Lambda_s, \xi_k \rangle_{\mathcal{H}} \\ &= \sum_{i=1}^{N} b_{ki} \left\langle \Lambda_s, \tilde{\Lambda}_{s_i} \right\rangle \\ &= \sum_{i=1}^{N} b_{ki} \left(P(s, s_i) - \frac{1}{N} \sum_{j=1}^{N} P(s, s_j) \right).\end{aligned} \tag{35}$$

In summary, the algorithm proceeds as follows:

1. Compute $\tilde{\mathbf{P}}$, the centered inner product matrix, according to equation (32);
2. Compute the k leading eigenvectors of $\tilde{\mathbf{P}}$ (i.e., the eigenvectors corresponding to the largest eigenvalues);
3. Compute the projections onto the k principal components, according to equation (35).

One can easily notice that the above result is very similar to kernel PCA [Schölkopf et al. 1998]. Indeed, kernel PCA which is a well known example of functional PCA applied in an RKHS of discrete data [Ramsay and Silverman 1997], which is in essence what we do here.

4.1.2 Results

To demonstrate the algorithm and empirically verify the differences between the various inner products, we show some results on synthetic spike trains generated from two homogeneous renewal point processes. The goal of the simulation is, first, to show that spike train PCA can be utilized to statistically analyze a set of spike trains in an unsupervised manner, and, second, to verify which inner products yield a projection consistent with our expectation. That is, that an inner product should be able to differentiate the different statistical characteristics in the spike trains, due to the different renewal processes, from intrinsic spike train variability, and therefore yield two separate clusters.[7]

The spike trains used in the simulation are one second long and have mean spike rate 20 spk/s. The inter-spike interval was gamma distributed with shape parameters $\theta = 0.5$ and $\theta = 3$, for the two renewal processes. The algorithm is applied on two datasets, one for computation of the principal components (i.e., a "training set"), and the other for testing. The evaluation and testing datasets comprised a total of 50 and 200 spike trains, respectively, half from each renewal process. The simulated spike trains are shown in figure 3.

[7]Although in general PCA does not preserve separability, this can be expected in this case because the clusters due to each renewal process should be compact. That is, the inner product should yield higher values between spike trains from the same renewal process (low in cluster variability) and lower values between spike trains from different renewal processes (high between cluster variability).

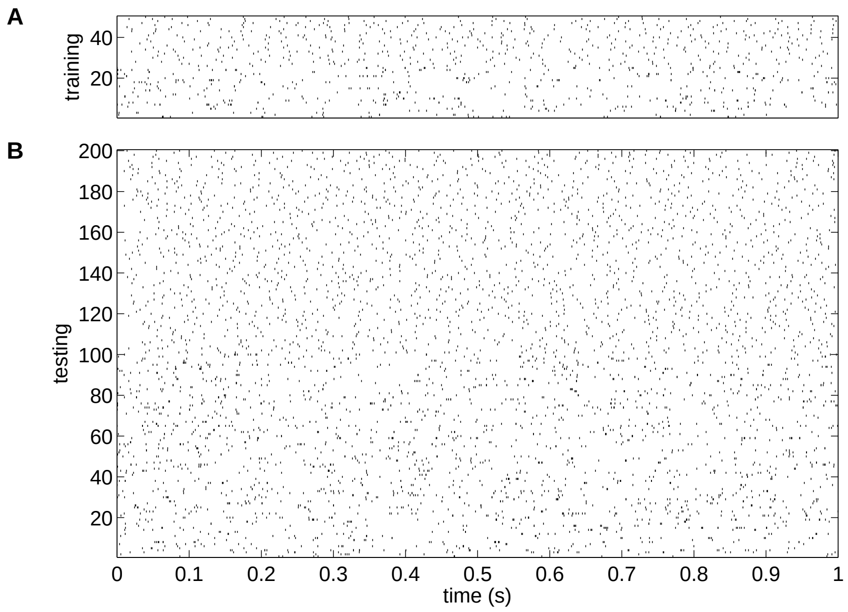
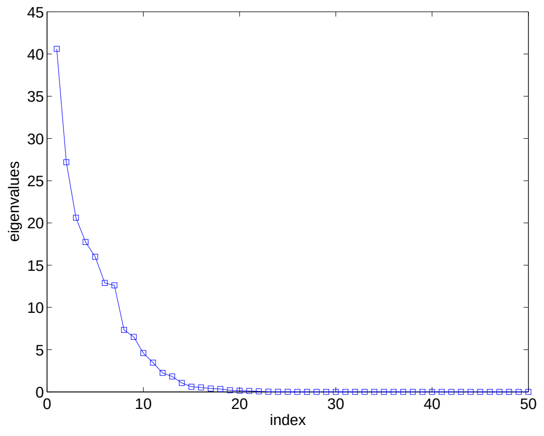


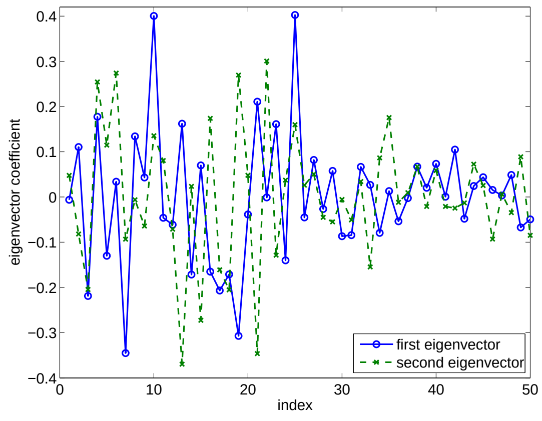
Figure 3: Spike trains used for computing the principal components (A), and for testing (B). In either case, the bottom half corresponds to spike trains drawn from a renewal process with shape parameter $\theta = 0.5$, and the top half from a renewal process with $\theta = 3$. Notice the difference in variability of the inter-spike intervals, with spike trains in the top half ($\theta = 3$) being much more regular due to the more compact distribution of the inter-spike intervals.

The PCA algorithm was applied using three different spike train inner products, defined in equations (17), (18) and (20). In the first case, k was chosen to be the laplacian function, $k(x, y) = \exp(-\left|x - y\right| / \tau)/2\tau$ with $\tau = 50$ms, which corresponds to a causal exponential smoothing function h with the same width [Paiva et al. 2009]. In the second case, the same smoothing function, also with $\tau = 2$ms, and the tanh nonlinearity function were used. For the nCI kernel (equation (20)), the fast implementation described in Paiva [2008] was used, with smoothing width $\tau = 50$ms and Gaussian kernel $\mathcal{K}_\sigma$ with $\sigma = 1$. These parameters, and choice of smoothing function and k, allows the algorithms to be directly compared, with differences only due to the actual form of the inner product. From an implementation perspective, the only difference between using different inner products is in the computation of the centered inner product matrix (equation (31). The remainder of the algorithm remains unaltered.

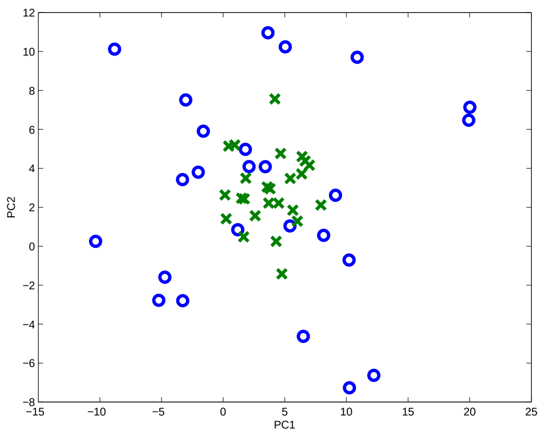
For the inner product defined in equation (17), the result of the eigendecomposition and projection of the spike trains onto the first two principal components is shown in figure 4. The first observation is that there is no clear distinction in the spike train variability in the first two principal components, but rather the variability decreases gradually. Moreover, both the eigenvectors and the projections shown in figure 4(b)–(d) reveal an overlap of the spike trains from the two renewal processes, being noticeable only the higher dispersion of spike trains from the first renewal model ($\theta = 0.5$) due to the more irregular firing. This means that the inter-process variability is smaller than the within-process variability. In other words, the inner product



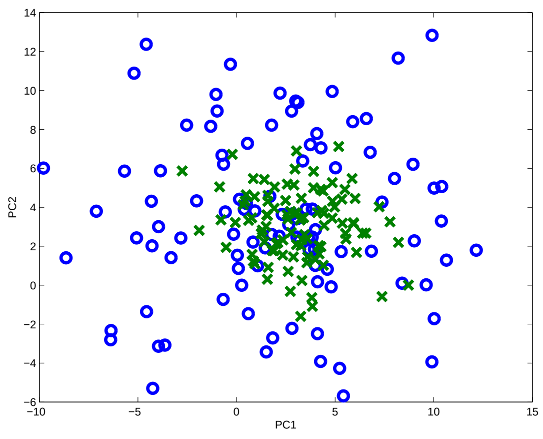
(a) Eigenvalues in decreasing order.



(b) First two eigenvectors.



(c) Projection of the spike trains in the training set.

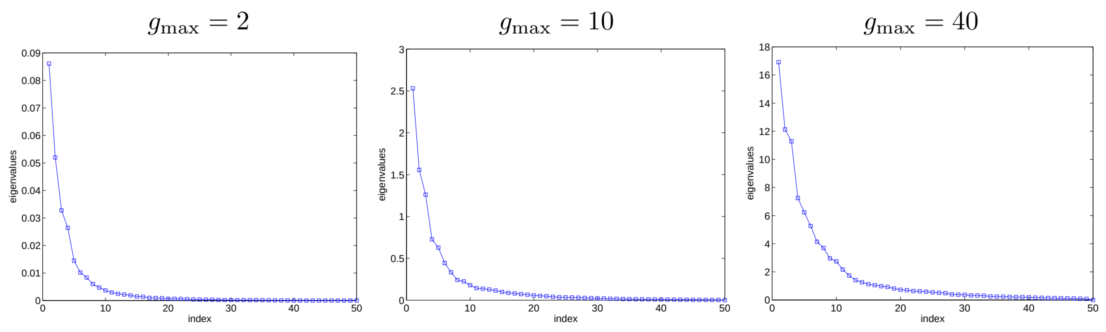


(d) Projection of the spike trains in the testing set.

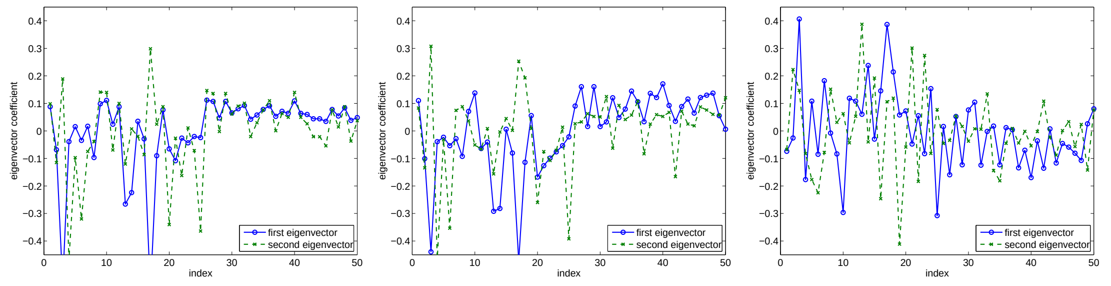
Figure 4: Eigendecomposition (top row) and projected spike trains onto the first two principal components (bottom) for PCA computed using the inner product in equation (17). (Different marks in the projection plots denote spike trains from different renewal processes.)

does not distinguish between spike trains with different inter-spike interval statistics, sensing only the difference in variability. This result is not surprising however. As pointed out in section 3.1.2, this inner product is memoryless and can discriminate only differences in intensity, which is the same in this case.

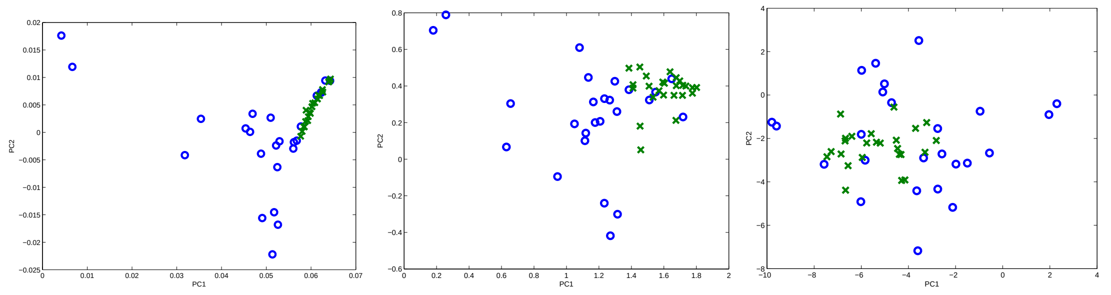
Similarly, we computed the principal components of the same set of spike trains using the inner product defined in equation (18). However, in this case, the choice of the $g_{\max}$ scaling parameter is very important. This parameter controls the transition point between the linear and saturation regions of the nonlinearity. More fundamentally, $g_{\max}$ controls the weighting of the higher-order spike interaction in time (cf. equation (46)) and, therefore, the ability to sense different inter-spike interval statistics. The results are shown in figure 5. It is easy to verify that lower values of $g_{\max}$ yield more meaningful (i.e., better separated) projections of the spike trains from the two renewal



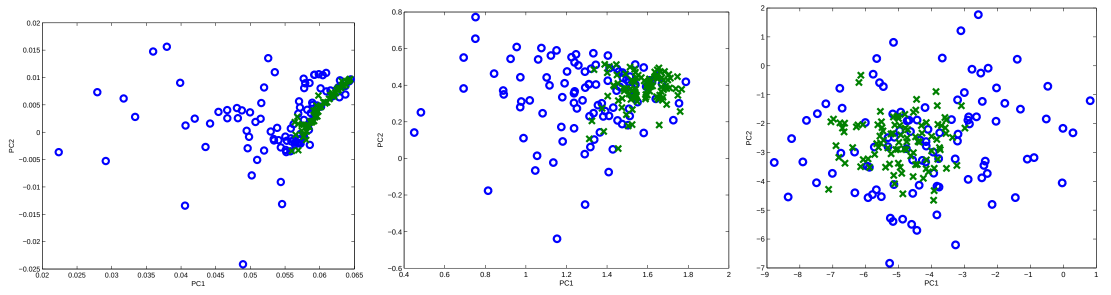
(a) Eigenvalues in decreasing order.



(b) First two eigenvectors.

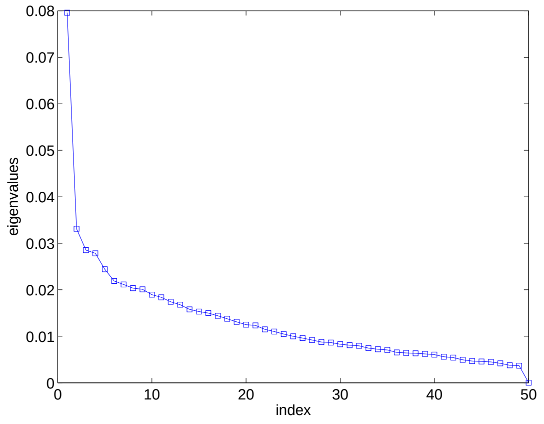


(c) Projection of the spike trains in the training set.

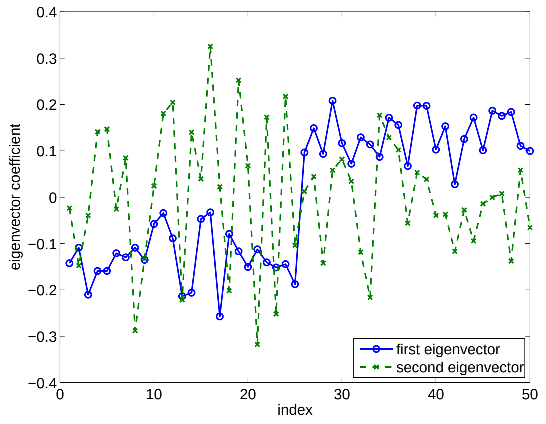


(d) Projection of the spike trains in the testing set.

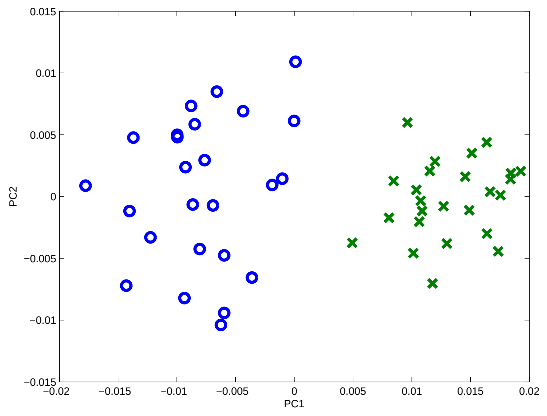
Figure 5: Eigendecomposition (top) and projected spike trains onto the first two principal components (bottom) for PCA computed using the inner product in equation (18). Each column correspond to the results for a given value of $g_{\max}$ shown on top. (Different marks in the projection plots denote spike trains from different renewal processes.)



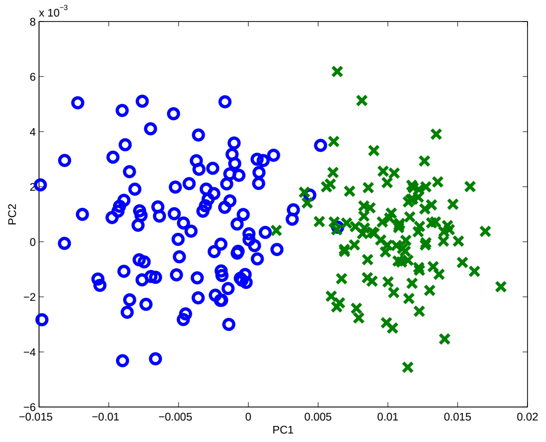
(a) Eigenvalues in decreasing order.



(b) First two eigenvectors.



(c) Projection of the spike trains in the training set.



(d) Projection of the spike trains in the testing set.

Figure 6: Eigendecomposition (top row) and projected spike trains onto the first two principal components (bottom) for PCA computed using the inner product in equation (20). (Different marks in the projection plots denote spike trains from different renewal processes.)

processes. This is noticeable, for example, in the top left plot, which show a better separation between the first principal components. However, if $g_{\max}$ is decreased further the results start to worsen (not shown). This is easy to understand since the nonlinearity will almost always be in the saturated region and thus the functional representations will be almost always equal. Conversely, as $g_{\max}$ is increased, the results converge toward those shown in figure 4. This occurs because as $g_{\max}$ is increased the nonlinearity will operate in its linear region, having almost no effect.

Finally, we computed the principal components using the inner product in equation (20); that is, the nCI kernel proposed in Paiva et al. [2009]. The results are shown in figure 6. Although this inner product has a kernel size parameter (of $\mathcal{K}_\sigma$), the results where very similar with any value within the range 0.1 to 10. It is notorious in the distribution of the eigenvalues the significantly higher variability captured in the first principal component versus the remaining. In comparison with the previous inner products only the results in figure 5 for $g_{\max} = 2$ approximate this behavior. The importance of the first principal component can be verified from the first eigenvector of the eigendecomposition, figure 6(b), which shows a clear distinction between spike trains generated from different renewal processes. This observation is clearly noticeable in the projections of spike trains from the two processes (figure 6(c)–(d)). This clearly shows that in this case the within-process variability is smaller than between-process. This seems a more faithful representation of the data since the within-process variability is only "noise" (i.e., due to stochasticity in the realizations). The separation between projections of the spike trains from different renewal processes was also noticeable in the results in figure 5 (for $g_{\max} = 2$), although not as strickingly. Moreover, differences in variability between processes where clearly represented. This is because, in this latter case, the nonlinear kernel $\mathcal{K}$ measures differences rather than the products in the smoothed spike train, which factors out the variability between realizations as long as they are similar.

4.2 Supervised learning: Fisher's linear discriminant

Classification of spike trains is very important. Classification can be used to decode the applied stimulus from a spike train. In neurophysiological studies, this has been utilized to study spike coding properties [Victor and Purpura 1996, Machens et al. 2003, 2001]. Indeed, it was the need to classify spike trains that first motivated several spike train distances [Victor and Purpura 1996].

In the following, we derive Fisher's linear discriminant for spike trains. Note that in spite of the linear decision boundary, the discrimination occurs in the RKHS. Hence, depending on the inner product, the decision boundary can easily be nonlinear in the spike train (or their functional representation) space. From a different perspective, one can think that, given an appropriate inner product, the classes can be made linearly separable, as occurs in kernel methods. The approach shown turns out to be very similar to that of kernel Fisher discriminant proposed by Mika et al. [1999].

4.2.1 Derivation

As before, we will use P to denote a generic spike train inner product inducing an RKHS $\mathcal{H}$ with mapping Λ to the RKHS. Consider two sets of spike trains, $C_1 = \{s_i^1 \in \mathcal{S}(\mathcal{T}), i = 1, \ldots, N_1\}$ and $C_2 = \{s_j^2 \in \mathcal{S}(\mathcal{T}), j = 1, \ldots, N_2\}$, one for each class. With some abuse of notation, the set of all spike trains will be denoted $C = C_1 \cup C_2 = \{s_n \in \mathcal{S}(\mathcal{T}), n = 1, \ldots, N\}$, with $N = N_1 + N_2$. Without loss of generality, only the case for two classes is considered here. For the general case, one can repeatedly apply the procedure discussed next between each class versus all others, or extend the derivation to find a subspace preserving discrimination.

The Fisher linear discriminant in the RKHS associated with the inner product is given by the element $\psi \in \mathcal{H}$, such that the means μ_k and variances σ_k^2 of the classes data projected onto ψ maximize the criterion,

$$J_F(\psi) = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}, \tag{36}$$

with the mean and variance of the projected classes given by,

$$\mu_k = \langle \psi, \bar{\Lambda}_k \rangle = \frac{1}{N_k} \sum_{i=1}^{N_k} \left\langle \psi, \Lambda_{s_i^k} \right\rangle \tag{37a}$$

$$\sigma_k^2 = \frac{1}{N_k} \sum_{i=1}^{N_k} \left(\left\langle \psi, \Lambda_{s_i^k} \right\rangle - \mu_k \right)^2 = \frac{1}{N_k} \sum_{i=1}^{N_k} \left\langle \psi, \Lambda_{s_i^k} \right\rangle^2 - \mu_k^2, \tag{37b}$$

where $\bar{\Lambda}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \Lambda_{s_i^k}$ is the mean, in the RKHS, of the kth class.

By the representer theorem [Kimeldorf and Wahba 1971, Schölkopf et al. 2001], ψ can be written as a linear combination of the spike trains mapped onto the RKHS $\{\Lambda_{s_i} \in \mathcal{H} : i = 1, \ldots, N, s_i \in C\}$. That is, there exist coefficients $c_1, \ldots, c_N \in \mathbb{R}$ such that,

$$\psi = \sum_{j=1}^{N} c_j \Lambda_{s_j} = \mathbf{c}^T \mathbf{\Lambda} \tag{38}$$

where $\mathbf{c}^T = [b_1, \ldots, b_N]$ and $\mathbf{\Lambda} = \left[\Lambda_{s_1}, \ldots, \Lambda_{s_N}\right]^T$. Substituting in equation (37a) yields,

$$\mu_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \sum_{j=1}^{N} c_j \left\langle \Lambda_{s_j}, \Lambda_{s_i^k} \right\rangle = \frac{1}{N_k} \mathbf{c}^T \mathbf{P}_k \mathbf{1}_{N_k} = \mathbf{c}^T \mathbf{M}_k, \tag{39}$$

with $\mathbf{M}_k = \frac{1}{N_k} \mathbf{P}_k \mathbf{1}_{N_k}$, and $\mathbf{1}_{N_k}$ is the $N_k \times 1$ vector of all ones. Likewise, substituting in equation (37b) results in,

$$\begin{aligned}
\sigma_k^2 &= \frac{1}{N_k} \sum_{i=1}^{N_k} \sum_{j=1}^{N} \sum_{n=1}^{N} c_j c_n \left\langle \Lambda_{s_j}, \Lambda_{s_i^k} \right\rangle \left\langle \Lambda_{s_i^k}, \Lambda_{s_n} \right\rangle - \mu_k^2 \\
&= \frac{1}{N_k} \mathbf{c}^T \mathbf{P}_k \mathbf{P}_k^T \mathbf{c} - \frac{1}{N_k^2} \mathbf{c}^T \mathbf{P}_k \mathbf{1}_{N_k} \mathbf{1}_{N_k}^T \mathbf{P}_k^T \mathbf{c} \\
&= \frac{1}{N_k} \mathbf{c}^T \mathbf{P}_k \left(\mathbf{I}_{N_k} - \frac{1}{N_k} \mathbf{1}_{N_k} \mathbf{1}_{N_k}^T \right) \mathbf{P}_k^T \mathbf{c},
\end{aligned} \tag{40}$$

where $\mathbf{P}_k$ is the $N \times N_k$ matrix with the ijth element given by the inner product between the ith spike train of C and the jth spike train of C_k, and $\mathbf{I}_{N_k}$ is the $N_k \times N_k$ identity matrix. Consequently, the Fisher criterion in equation (36) can be written as,

$$J_F(\mathbf{c}) = \frac{\mathbf{c}^T \mathbf{S}_b \mathbf{c}}{\mathbf{c}^T \mathbf{S}_w \mathbf{c}}, \tag{41}$$

with between-class scatter matrix, $\mathbf{S}_b$, and within-class scatter matrix, $\mathbf{S}_w$, given by,

$$\mathbf{S}_b = (\mathbf{M}_1 - \mathbf{M}_2)(\mathbf{M}_1 - \mathbf{M}_2)^T, \tag{42a}$$

$$\mathbf{S}_w = \sum_{k=1,2} \mathbf{P}_k (\mathbf{I}_{N_k} - \frac{1}{N_k} \mathbf{1}_{N_k} \mathbf{1}_{N_k}^T) \mathbf{P}_k^T \tag{42b}$$

As in the usual Fisher linear discriminant result [Duda et al. 2000], the vector $\mathbf{c}$ that maximizes J_F can be obtained as

$$\mathbf{c} = \mathbf{S}_w^{-1} (\mathbf{M}_1 - \mathbf{M}_2). \tag{43}$$

Clearly, this solution is ill-posed in general since $\mathbf{S}_w$ is not guaranteed to be invertible. Hence, in practice, $\mathbf{S}_w + \epsilon\mathbf{I}_N$ is used instead. Basically, the term $\epsilon\mathbf{I}_N$ implements Tikhonov regularization, with multiplier ϵ.

To compute the projection of a given spike train onto the Fisher discriminant, in the training set (i.e., the set C) or in testing, we need to compute the inner product, in the RKHS, between the transformed spike train and ψ, given as,

$$\begin{aligned}\mathrm{Proj}_{\psi}(\Lambda_s) &= \left\langle \Lambda_s, \psi \right\rangle \\ &= \sum_{j=1}^{N} c_j \left\langle \Lambda_s, \Lambda_{s_j} \right\rangle \\ &= \sum_{j=1}^{N} c_j P(s, s_j).\end{aligned} \tag{44}$$

After projection onto the Fisher linear discriminant, one still needs to find the optimum threshold for classification. However, this is a relatively simple task and many methods exist in the literature [Duda et al. 2000].

To summarize, the algorithm can be implemented in the following steps:

1. Compute the $\mathbf{P}_k$ matrices, $k = 1, 2$;
2. Compute the matrices $\mathbf{M}_k$ and $\mathbf{S}_w$ according to equations (39) and (42b);
3. Compute the vector $\mathbf{c}$ which defines the Fisher linear discriminant in the RKHS, according to equation (43);
4. Find the optimum classification threshold in the training set;
5. Classify the testing spike trains by computing their projection onto the Fisher linear discriminant (equation (44)) and thresholding.

4.2.2 Results

In this section, we demonstrate the algorithm for synthetic spike trains generated from two homogeneous renewal point processes. Specifically, we utilized the same datasets used to evaluate the PCA algorithm in section 4.1.2, shown in figure 3. In this example classification is primarily a task of discriminating between spike trains with different inter-spike interval distributions.

As before, the Fisher linear discriminant algorithm was compared using three different spike train inner products, defined in equations (17), (18) and (20). The same parameters were used. As with the PCA algorithm, the Fisher linear discriminant algorithm is independent of the inner product used; the difference being only in the computation of the inner product matrices, $\mathbf{P}_k$. The classification threshold was the value that minimized the probability of error in the training set.

For the inner product of equation (18) one must carefully chose the appropriate scaling parameter $g_{\max}$ of the f nonlinearity. For simplicity, in this case, we tested the Fisher discriminant classifier using a range of values between 0.5 and 50, and report the results on the testing set. Of course, in a practical scenario the optimum $g_{\max}$ value would have to be found by cross-validation.

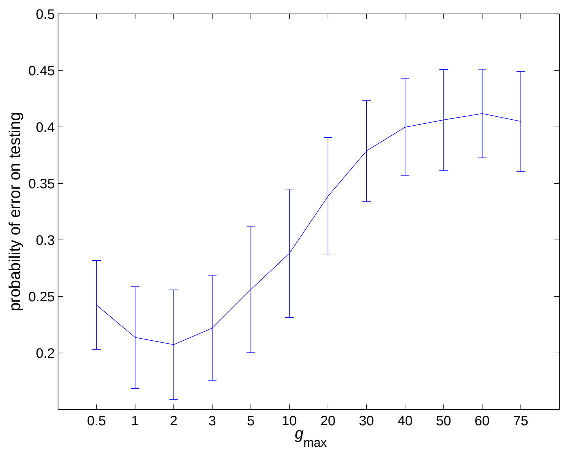


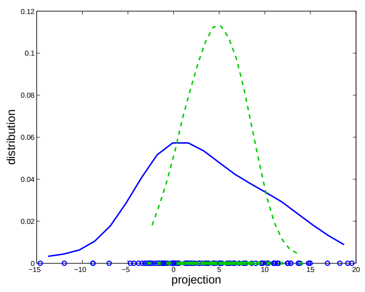
Figure 7: Probability of error in the testing set of the Fisher discriminant classifier for different values of $g_{\max}$. The statistics, mean and standard deviation, where estimated over 100 Monte Carlo runs.

Inner product	Probability of error
equation (17)	0.401 ± 0.040
equation (18)	0.207 ± 0.048
equation (20)	0.025 ± 0.013

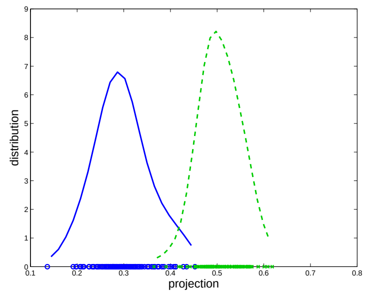
Table 1: Probability of error in the testing set of the Fisher discriminant classifier using three different inner products. For the inner product in equation (18), the result with lowest classification error is shown (obtained with $g_{\max} = 2$, cf. figure 7). The results are the statistics (mean $\pm$ standard deviation) over 100 Monte Carlo runs.

The probability of error in the testing set is shown in figure 7. The figure can be intuitively understood in light of the role of the nonlinearity in the inner product. For large values of $g_{\max}$, the nonlinearity operates in its linear range and therefore the result is similar to using the inner product in equation (17). As $g_{\max}$ is decreased, the classification improves because the nonlinearity allows the inner product to utilize more information from the higher-order moments (cf. equation (46)). However, for very small values, the performance decreases because the nonlinearity obfuscates the dynamics of the linear component of the synapse model.

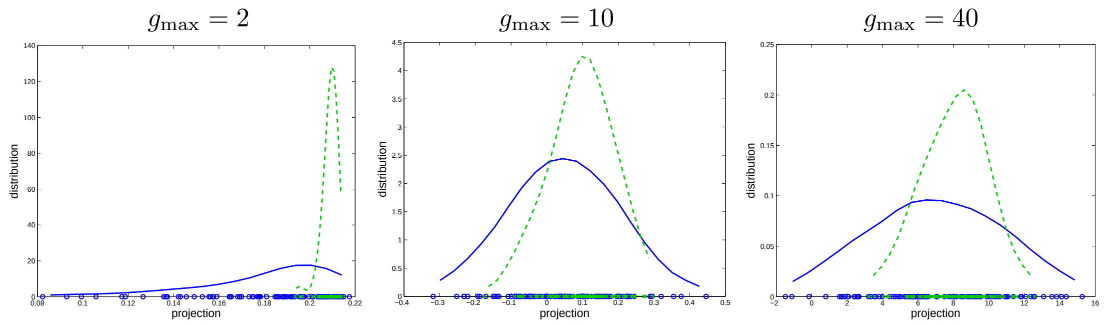
The classification results using each of the inner products are summarized in table 1. To help understand the results, the distribution of the projections onto the Fisher discriminant for one of the Monte Carlo runs is shown in figure 8. Overall, these results clearly show the importance of using inner products that are able to capture memory information. For these datasets, the inner product defined in equation (17) performs clearly worse (notice in figure 8(a) the overlap in the projections onto the Fisher discriminant). Because the discriminating characteristic is the different inter-spike interval distributions, but because this inner product is only sensitive to differences in intensity, the results are only slightly above chance level. Using the inner product in



(a) Using the inner product in equation (17).



(b) Using the inner product in equation (20).



(c) Using the inner product in equation (18), for different values of $g_{\max}$.

Figure 8: Distribution of the projections onto the Fisher discriminant of spike trains in the testing set. For the inner product in equation (18), distributions are shown for three values of $g_{\max}$.

equation (18) greatly improves the probability of error because of the use of memory information from the higher-order interactions, but only if an appropriate value of $g_{\max}$ is chosen, as shown in figure 7 and noticed in figure 8(c). Finally, using the inner product in equation (20) yields the best results. As discussed for the PCA results, these results are considerably better than the previous ones because the latter inner product nonlinearly measures the similarity between functional representations. This avoids the ambiguity in the inner product in equation (18),[8] adding to the expressiveness of the inner product (figure 8(b)). Also, in the latter inner product we have to choose the bandwidth parameter σ of the nonlinear kernel $\mathcal{K}_\sigma$. However, unlike the former, the sensitivity on this parameter is very low. We tried values between 0.1 to 10, but the results changed only $\pm 0.1\%$.

5 Conclusion

A general framework to develop spike train machine learning methods was presented. The most important contribution of this work is on how to define spike train inner products for building this framework. Unlike the top-down approach often taken in kernel methods, the approach

[8] Specifically, we are referring to the ambiguity arising from the fact that pairs of spike trains functional representations with the same geometric mean in time and/or the same arithmetic mean over time have the same inner product value.

emphasized here builds the inner products bottom-up from basic spike train descriptors. As argued in section 3, inner products are the fundamental operators needed for machine learning. Then, utilizing the mathematical structure of the reproducing kernel Hilbert space induced by a spike train inner product, it is easy to derive from first principles machine learning algorithms, as demonstrated in section 4.

Two different approaches were presented on how to define spike train inner products. The first, introduced in section 3.1.1, builds on spike times as basic elements. Although it allows for a priori knowledge in spike domain to be easily incorporated in the inner product definition, these definitions per se lack a more in-depth understanding of both the biological and/or statistical implications. Hence, the approach emphasized in this chapter is based on functional representations of spike trains, which were motivated either as a biological-modeling problem or from statistical descriptors. One of the key advantages of this approach is the explicit construction of the inner product. In this way, the properties of the inner product, induced RKHS, and derived constructs are brought forth. The biological-modeling perspective highlights the role of synaptic models, and thus aims to approximate the computational mechanisms in a neuron. On the other hand, by defining the inner products from conditional intensity functions one focus on the characteristics of the underlying point processes. Perhaps most importantly, verifying the duality between the two representations can provide valuable insight about the statistical limitations of a synaptic model. Therefore, these inner products (or their derived constructs) can be used to analyze neurophysiological data, and hint at the underlying principles of information encoding from these perspectives.

The inner product with the form given in equation (16) is statistically completely general but, in practice, its usefulness is limited by the current methods for estimation of the conditional intensity function. Although methods for point processes with memory have been recently proposed in the literature, currently there is only effective intensity function estimators for Poisson process.[9] In this regards, however, the inner products defined in equations (18) and (20) show considerable promise. Through the use of nonlinearity, those inner products circumvent these limitations and show sensitivity to spike time statistics, albeit with limited expressiveness. In our experiments using renewal point processes both inner products clearly outperform the inner product in equation (17). It must be remarked that the inner product in equation (17) is in essence a continuous version of the widely used cross-correlation of binned spike trains [Park et al. 2008, Paiva et al. 2008], which shows the fundamental limitations of many spike train methods currently in use.

A Higher-order spike interactions through nonlinearity

In this section we prove the statement that the nonlinearity in equation (7) reflects higher-order interactions in the spike train.

As an example, consider the nonlinear function,

$$f(x) = g_{\max}\left[1 - \exp\left(-\frac{x^2}{2g_{\max}^2}\right)\right]. \tag{45}$$

[9]By "effective" it is meant estimation with a reasonably small amount of data, such as 1 second long spike trains.

That is, f is an inverted Gaussian, appropriately scaled by $g_{\max}$. The MacLaurin series expansion of f (i.e., Taylor series around the origin) is

$$f(x) = \frac{1}{2}\frac{x^2}{g_{\max}} - \frac{1}{8}\frac{x^4}{g_{\max}^3} + \frac{1}{48}\frac{x^6}{g_{\max}^5} - \frac{1}{384}\frac{x^8}{g_{\max}^7} + \cdots \tag{46}$$

In the nonlinear model described by equation (7), the argument of the nonlinearity is the evoked potential according to the linear model, given by equation (3). Hence, the powers on x in the MacLaurin series extend to,

$$\begin{aligned} x^2 &= \left(\sum_{i=1}^{N} h(t-t_i)\right)^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} h(t-t_i)h(t-t_j) \\ x^4 &= \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} h(t-t_i)h(t-t_j)h(t-t_k)h(t-t_l) \\ &\cdots \end{aligned} \tag{47}$$

This shows that the membrane potential of the nonlinear model, equation (7), is a weighted sum of the moments of the linear potential. In turn, these moments depend on the interactions, smoothed by h, of a number of spikes of the same order as the moment.

The moments accounted for and their weighting in the nonlinear membrane potential depend on the choice of f. In the case of the just shown, it depends only on even order moments. However, if the tanh nonlinearity was utilized in f instead,

$$f(x) = g_{\max}\tanh\left(\frac{x}{g_{\max}}\right), \qquad x \in \mathbb{R}^+, \tag{48}$$

then, the nonlinear membrane potential would depend only on odd order moments.

B Proofs

This section presents the proofs for properties 2, 3 and 5, given in section 3.2.

Proof of Property 2. The symmetry follows immediately from property 1. In the following we will use the general form of the inner products in equations (15) and (16),

$$P(s_i, s_j) = \left\langle \phi_{s_i}, \phi_{s_j} \right\rangle = \int_{\mathcal{T}} \phi_{s_i}(t)\phi_{s_j}(t)dt, \tag{49}$$

where ϕ denotes either the membrane potential or conditional intentity function.

By definition, the kernel on spike trains, $P : \mathcal{S}(\mathcal{T}) \times \mathcal{S}(\mathcal{T}) \to \mathbb{R}$, is positive definite if and only if [Aronszajn 1950, Parzen 1959],

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j P(s_i, s_j) \geq 0, \tag{50}$$

for any $n > 0$, $s_1, s_2, \ldots, s_n \in \mathcal{S}(\mathcal{T})$ and $a_1, a_2, \ldots, a_n \in \mathbb{R}$. Using the general form yields,

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j P(s_i, s_j) &= \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \int_{\mathcal{T}} \phi_{s_i}(t)\phi_{s_j}(t)dt \\ &= \int_{\mathcal{T}} \left(\sum_{i=1}^{n} a_i \phi_{s_i}(t)\right)\left(\sum_{j=1}^{n} a_j \phi_{s_j}(t)\right) dt \\ &= \left\langle \sum_{i=1}^{n} a_i \phi_{s_i}, \sum_{j=1}^{n} a_j \phi_{s_j} \right\rangle \\ &= \left\| \sum_{i=1}^{n} a_i \phi_{s_i} \right\| \geq 0,\end{aligned} \tag{51}$$

since the norm is non-negative by definition. □

Proof of Property 3. Again, symmetry follows immediately from property 1. A matrix $\mathbf{P}$ is non-negative definite if and only if $\mathbf{a}^T\mathbf{P}\mathbf{a} \geq 0$, for any $\mathbf{a}^T = [a_1, \ldots, a_n]$ with $a_i \in \mathbb{R}$. For an inner product matrix, we have that (using the general inner product form),

$$\mathbf{a}^T\mathbf{P}\mathbf{a} = \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j P(s_i, s_j) \geq 0, \tag{52}$$

as shown in the previous proof (equation (51)). Hence, the inner product matrices are symmetric and non-negative definite. □

Proof of Property 5. Consider the 2×2 inner product matrix associated with any of the defined inner products,

$$\mathbf{P} = \begin{bmatrix} P(s_i, s_i) & P(s_i, s_j) \\ P(s_i, s_j) & P(s_j, s_j) \end{bmatrix}. \tag{53}$$

From property 3, this matrix is symmetric and non-negative definite. Hence, its determinant is non-negative [Harville 1997, pg. 245]. That is,

$$\det(\mathbf{P}) = P(s_i, s_i)P(s_j, s_j) - P(s_i, s_j)^2 \geq 0, \tag{54}$$

which proves the result in equation (21). □

C Brief introduction to RKHS theory

In this appendix, we briefly introduce some basic concepts of reproducing kernel Hilbert space (RKHS) theory necessary for the understanding of this chapter. The presentation here is meant to be as general and introductory as possible.

The fundamental result in RKHS theory is the well-known *Moore-Aronszajn theorem* [Aronszajn 1950, Moore 1916]. Let K denote a symmetric and positive definite function of two variables defined on some space E. That is, a function $K(\cdot, \cdot) : E \times E \to \mathbb{R}$ which verifies:

(i) Symmetry: $K(x, y) = K(y, x), \quad \forall x, y \in E$.

(ii) Positive definiteness: for any finite number of $n \in \mathbb{N}$ points $x_1, x_2, \ldots, x_n \in E$, and any corresponding n coefficients $c_1, c_2, \ldots, c_n \in \mathbb{R}$,

$$\sum_{i=1}^{l} \sum_{j=1}^{l} c_i c_j K(x_i, x_j) \geq 0. \tag{55}$$

These are sometimes called Mercer conditions [Mercer 1909]. Then the Moore-Aronszajn theorem [Aronszajn 1950, Moore 1916] guaranties that there exists a unique Hilbert space $\mathcal{H}$ of real valued functions on E such that, for every $x \in E$,

(i) $K(x, \cdot) \in \mathcal{H}$, and

(ii) for any $f \in \mathcal{H}$

$$f(x) = \langle f(\cdot), K(x, \cdot) \rangle_{\mathcal{H}} . \tag{56}$$

The identity on equation (56) is called the *reproducing property* of K, and, for this reason, $\mathcal{H}$ is said to be a reproducing kernel Hilbert space with reproducing kernel K. Note that equation (55) is equivalent to say that, for a set of points $\{x_i \in E : i = 1, \ldots, n\}$, the matrix,

$$\mathbf{K} = \begin{bmatrix} K(x_1, x_1) & K(x_1, x_2) & \ldots & K(x_1, x_n) \\ K(x_2, x_1) & K(x_2, x_2) & \ldots & K(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ K(x_n, x_1) & K(x_n, x_2) & \ldots & K(x_n, x_n) \end{bmatrix}, \tag{57}$$

is positive *semi*-definite. $\mathbf{K}$ often referred to as the *Gram matrix*.

A very important result of this theorem is that K evaluates the inner product in the RKHS. This can readily verified using the reproducing property,

$$K(x, y) = \langle K(x, \cdot), K(y, \cdot) \rangle_{\mathcal{H}} , \tag{58}$$

for any $x, y \in E$. This identity is the *kernel trick*, well known in kernel methods, and the main tool for computation in the RKHS, since most algorithms can be formulated using only inner products, as discussed in the introduction of section 3.

Another important concept is that of congruent spaces. Two spaces, say E and F, are said to be congruent if there exists an isometric isomorphism between them. This means that there exist a one-to-one mapping (i.e., an isomorphism) $\mathscr{G} : E \to F$, such that, for any $x, y \in E$,

$$\langle x, y \rangle_E = \langle \mathscr{G}(x), \mathscr{G}(y) \rangle_F . \tag{59}$$

That is, if there exists a one-to-one inner product-preserving mapping between the two spaces. The mapping $\mathscr{G}$ is called a congruence. By two congruent inner products it is meant that there exists a mapping between the spaces where the inner products are defined, such that equation (59) holds.

Acknowledgements

A. R. C. Paiva was supported by Fundação para a Ciência e a Tecnologia (FCT), Portugal, under grant SRFH/BD/18217/2004. This work was partially supported by NSF grants ECS-0422718 and CISE-0541241.

References

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404, May 1950.

L. A. Baccalá and K. Sameshima. Directed coherence: a tool for exploring functional interactions among brain structures. In M. A. L. Nicolelis, editor, *Methods for Neural Ensemble Recordings*, pages 179–192. CRC Press, 1999.

R. Barbieri, M. C. Quirk, L. M. Frank, M. A. Wilson, and E. N. Brown. Construction and analysis of non-Poisson stimulus-response models of neural spiking activity. *Journal of Neuroscience Methods*, 105(1):25–37, Jan. 2001. doi: 10.1016/S0165-0270(00)00344-7.

S. M. Bohte, J. N. Kok, and H. L. Poutré. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1–4):17–37, Oct. 2002. doi: 10.1016/S0925-2312(01)00658-0.

E. N. Brown, D. P. Nguyen, L. M. Frank, M. A. Wilson, and V. Solo. An analysis of neural receptive field plasticity by point process adaptive filtering. *Proceedings of the National Academy of Science, USA*, 98:12261–12266, 2001.

A. Carnell and D. Richardson. Linear algebra for time series of spikes. In *Proceedings of the European Symposium on Artificial Neural Networks*, pages 363–368, Bruges, Belgium, Apr. 2005.

Z. Chi and D. Margoliash. Temporal precision and temporal drift in brain and behavior of zebra finch song. *Neuron*, 32(1–20):899–910, Dec. 2001.

Z. Chi, W. Wu, Z. Haga, N. G. Hatsopoulos, and D. Margoliash. Template-based spike pattern identification with linear convolution and dynamic time warping. *Journal of Neurophysiology*, 97(2):1221–1235, Feb. 2007. doi: 10.1152/jn.00448.2006.

M. Christen, A. Kohn, T. Ott, and R. Stoop. Measuring spike pattern reliability with the Lempel-Ziv-distance. *Journal of Neuroscience Methods*, 156(1-2):342–350, Sept. 2006. doi: 10.1016/j.jneumeth.2006.02.023.

D. R. Cox and V. Isham. *Point Processes*. Chapman and Hall, 1980.

D. J. Daley and D. Vere-Jones. *An Introduction to the Theory of Point Processes*. Springer-Verlag, New York, NY, 1988.

P. Dayan and L. F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. MIT Press, Cambridge, MA, USA, 2001.

K. I. Diamantaras and S. Y. Kung. *Principal Component Neural Networks: Theory and Applications*. John Wiley & Sons, 1996.

J. E. Dowling. *Neurons and Networks: An Introduction to Behavioral Neuroscience*. Harvard University Press, Cambridge, MA, 1992.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification.* Wiley Interscience, 2nd edition, 2000.

U. R. Eden, L. M. Frank, R. Barbieri, V. Solo, and E. N. Brown. Dynamic analysis of neural encoding by point process adaptive filtering. *Neurocomputing*, 16(5):971–998, May 2004.

W. J. Freeman. *Mass Action in the Nervous System.* Academic Press, New York, 1975. URL `http://sulcus.berkeley.edu/MANSWWW/MANSWWW.html`.

W. Gerstner and W. Kistler. *Spiking Neuron Models.* MIT Press, 2002.

D. A. Harville. *Matrix algebra from a statistician's perspective.* Springer, 1997.

C. Houghton. Studying spike trains using a van Rossum metric with a synapse-like filter. *Journal of Computational Neuroscience*, 26:149–155, 2009. doi: 10.1007/s10827-008-0106-6.

C. Houghton and K. Sen. A new multineuron spike train metric. *Neural Computation*, 20(6): 1495–1511, 2008. doi: 10.1162/neco.2007.10-06-350.

J. M. Hurtado, L. L. Rubchinsky, and K. A. Sigvardt. Statistical method for detection of phase-locking episodes in neural oscillations. *Journal of Neurophysiology*, 91(4):1883–1898, Apr. 2004.

M. A. Jabri, R. J. Coggins, and B. G. Flower, editors. *Learning in Graphical Models.* Springer, 1996. ISBN 0-412-61630-0, 978-0-412-61630-3.

F. V. Jensen. *Bayesian Networks and Decision Graphs.* Springer, New York, NY, 2001. ISBN 0-387-95259-4.

M. I. Jordan, editor. *Learning in Graphical Models.* MIT Press, 1998. ISBN 0-262-60032-3, 978-0-262-60032-3.

T. Kailath. RKHS approach to detection and estimation problems–part I: Deterministic signals in gaussian noise. *IEEE Transactions on Information Theory*, 17(5):530–549, Sept. 1971.

T. Kailath and D. L. Duttweiler. An RKHS approach to detection and estimation problems–part III: Generalized innovations representations and a likelihood-ratio formula. *IEEE Transactions on Information Theory*, 18(6):730–745, Nov. 1972.

T. Kailath and H. L. Weinert. An RKHS approach to detection and estimation problems–part II: Gaussian signal detection. *IEEE Transactions on Information Theory*, 21(1):15–23, Jan. 1975.

A. F. Karr. *Point Processes and their Statistical Inference.* Marcel Dekker, New York, NY, 1986.

R. E. Kass and V. Ventura. A spike-train probability model. *Neural Computation*, 13(8):1713–1720, Aug. 2001.

R. E. Kass, V. Ventura, and C. Cai. Statistical smoothing of neuronal data. *Network: Computation in Neural Systems*, 14:5–15, 2003.

R. E. Kass, V. Ventura, and E. N. Brown. Statistical issues in the analysis of neuronal data. *Journal of Neurophysiology*, 94:8–25, 2005. doi: 10.1152/jn.00648.2004.

G. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82–95, Jan. 1971.

T. Kreuz, J. S. Haas, A. Morelli, H. D. I. Abarbanel, and A. Politi. Measuring spike train synchrony. *Journal of Neuroscience Methods*, 165(1):151–161, Sept. 2007. doi: 10.1016/j.jneumeth.2007.05.031.

W. Maass and C. M. Bishop, editors. *Pulsed Neural Networks*. MIT Press, 1998.

W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11): 2531–2560, 2002. doi: 10.1162/089976602760407955.

C. K. Machens, P. Prinz, M. B. Stemmler, B. Ronacher, and A. V. M. Herz. Discrimination of behaviorally relevant signals by auditory receptor neurons. *Neurocomputing*, 38–40:263–268, 2001.

C. K. Machens, H. Schütze, A. Franz, O. Kolesnikova, M. B. Stemmler, B. Ronacher, and A. V. M. Herz. Single auditory neurons rapidly discriminate conspecific communication signals. *Nature Neuroscience*, 6(4):341–342, Apr. 2003.

V. Z. Marmarelis. *Nonlinear Dynamic Modelling of Physiological Systems*. IEEE Press Series in Biomedical Engineering. John Wiley & Sons, 2004.

J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London – Series A*, 209:415–446, 1909.

S. Mika, G. Ratsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In *Proceedings of the International Workshop on Neural Networks for Signal Processing*, pages 41–48, Aug. 1999. doi: 10.1109/NNSP.1999.788121.

E. H. Moore. On properly positive Hermitian matrices. *Bulletin of the American Mathematical Society*, 23:59, 1916.

N. S. Narayanan, E. Y. Kimchi, and M. Laubach. Redundancy and synergy of neuronal ensembles in motor cortex. *Journal of Neuroscience*, 25(17):4207–4216, 2005. doi: 10.1523/JNEUROSCI.4697-04.2005.

A. R. C. Paiva. *Reproducing Kernel Hilbert Spaces for Point Processes, with applications to neural activity analysis*. PhD thesis, University of Florida, 2008.

A. R. C. Paiva, I. Park, and J. C. Príncipe. Reproducing kernel Hilbert spaces for spike train analysis. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-2008*, Las Vegas, NV, USA, Apr. 2008.

A. R. C. Paiva, I. Park, and J. C. Príncipe. A reproducing kernel Hilbert space framework for spike train signal processing. *Neural Computation*, 21(2):424–449, Feb. 2009. doi: 10.1162/neco.2008.09-07-614.

I. Park, A. R. C. Paiva, T. B. DeMarse, and J. C. Príncipe. An efficient algorithm for continuous-time cross correlation of spike trains. *Journal of Neuroscience Methods*, 128(2):514–523, Mar. 2008. doi: 10.1016/j.jneumeth.2007.10.005.

E. Parzen. Statistical inference on time series by Hilbert space methods. Technical Report 23, Applied Mathematics and Statistics Laboratory, Stanford University, Stanford, California, Jan. 1959.

B. Pesaran, J. S. Pezaris, M. Sahani, P. P. Mitra, and R. A. Andersen. Temporal structure in neuronal activity during working memory in macaque parietal cortex. *Nature Neuroscience*, 5 (8):805–811, Aug. 2002. doi: 10.1038/nn890.

J. O. Ramsay and B. W. Silverman. *Functional Data Analysis*. Springer-Verlag, 1997. ISBN 0-387-94956-9.

R.-D. Reiss. *A Course on Point Processes*. Springer-Verlag, New York, NY, 1993.

B. J. Richmond, L. M. Optican, and H. Spitzer. Temporal encoding of two-dimensional patterns by single units in primate primary visual cortex. I. Stimulus-response relations. *Journal of Neurophysiology*, 64(2):351–369, Aug. 1990.

E. Schneidman, W. Bialek, and I. Berry, Michael J. Synergy, redundancy, and independence in population codes. *Journal of Neuroscience*, 23(37):11539–11553, 2003.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors. *Advances in Kernel Methods: Support Vector Learning*. MIT Press, 1999.

B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Proceedings of the European Conference on Computational Learning Theory*, pages 416–426, July 2001.

B. Schrauwen and J. V. Campenhout. Linking non-binned spike train kernels to several existing spike train distances. *Neurocomputing*, 70(7–8):1247–1253, Mar. 2007. doi: 10.1016/j.neucom.2006.11.017.

S. Schreiber, J. M. Fellous, D. Whitmer, P. Tiesinga, and T. J. Sejnowski. A new correlation-based measure of spike timing reliability. *Neurocomputing*, 52–54:925–931, June 2003. doi: 10.1016/S0925-2312(02)00838-X.

D. L. Snyder. *Random Point Processes*. John Viley & Sons, New York, 1975.

D. Song, R. H. M. Chan, V. Z. Marmarelis, R. E. Hampson, S. A. Deadwyler, and T. W. Berger. Nonlinear dynamic modeling of spike train transformations for hippocampal-cortical prostheses. *IEEE Transactions on Biomedical Engineering*, 54(6):1053–1066, June 2007. doi: 10.1109/TBME.2007.891948.

W. Truccolo, U. T. Eden, M. R. Fellows, J. P. Donoghue, and E. N. Brown. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology*, 93:1074–1089, Feb. 2005. doi: 10.1152/jn.00697.2004.

M. Tsodyks and H. Markram. The neural code between neocortical pyramidal neurons depends on neurotransmitter releaseprobability. *Proceedings of the National Academy of Science, USA*, 94(2):719–723, Jan. 1997.

M. C. W. van Rossum. A novel spike distance. *Neural Computation*, 13(4):751–764, 2001.

V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.

J. D. Victor. Binless strategies for estimation of information from neural data. *Physical Reviews E*, 66(5):051903, 2002. doi: 10.1103/PhysRevE.66.051903.

J. D. Victor and K. P. Purpura. Nature and precision of temporal coding in visual cortex: A metric-space analysis. *Journal of Neurophysiology*, 76(2):1310–1326, Aug. 1996.

J. D. Victor and K. P. Purpura. Metric-space analysis of spike trains: theory, algorithms, and application. *Network: Computation in Neural Systems*, 8:127–164, Oct. 1997.

G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics.* SIAM, 1990.

Y. Wang, A. R. C. Paiva, J. C. Príncipe, and J. C. Sanchez. Sequential Monte Carlo point process estimation of kinematics from neural spiking activity for brain machine interfaces. *Neural Computation*, 21(10):2894–2930, Oct. 2009. doi: 10.1162/neco.2009.01-08-699.