ESANN'2005 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 27-29 April 2005, d-side publi., ISBN 2-930307-05-6.

Linear Algebra for Time Series of Spikes

Andrew Carnell[1] and Daniel Richardson[2]

University of Bath - Department of Computer Science
Bath - UK

Abstract. The set of time series of spikes is expanded into a vector space, V, by taking all linear combinations. A definition is then given for an inner product on this vector space. This gives a definition of norm, of distance between time series, and of orthogonality. This also allows us to compute the best approximation to a given time series which can be formed by a linear combination of some given collection of time series. It is shown how this can be applied to a very simple learning or approximation problem.

1 Introduction

We define a spike at time t_1 to be a function $s(t_1)$ of time t so that $s(t_1)(t) = 1$ if $t = t_1$, and is zero otherwise. We define a time series to be a finite sum of spikes,

$$\sum_{i=1}^{N} s(t_i)$$

with $t_1, \ldots, t_N$ distinct times.

We define a weighted time series to be a finite sum of the form

$$\sum_{i=1}^{N} c_i s(t_i)$$

The coefficients c_i and the times t_i can be any real numbers, and the number of terms, N, can be any natural number. We let V be the vector space, over the real numbers, of weighted time series, with the obvious definitions of addition and scalar multiplication. V is infinite dimensional with uncountable basis.

We consider the following basic problems. Suppose $w_1, \ldots, w_k$ are time series and suppose also that we are given a goal time series g, and an output neuron G, which behaves as one of the spiking models discussed, for example in [1]. Let $inp(G) = c_1 w_1 + \cdots + c_k w_k$ be input to G. Let $out(G)$ be the output time series produced by G when given this input.

Problem 1). Find values of weights $c_1, \ldots, c_k$ so that $inp(G)$ is close to g.

Problem 2). Find values of weights $c_1, \ldots, c_k$ so that $out(G)$ is close to g. In order to say more precisely what "close" means, we define an inner product on V.

2 Inner Product

An inner product on a vector space, V, over the reals is a function $< u, w >: V \times V \to R$ so that:

1. $< u, w > = < w, u >$.
2. $< u + v, w > = < u, w > + < v, w >$.
3. $< cu, w > = c < u, w >$, for any real c.
4. $< u, u > \geq 0$.
5. $< u, u > = 0$ only if $u = 0$.

Because of the linearity of the inner product, and since V is formed by linear combinations of spikes, we only need to define the inner product between two spikes. We define

$$< s(t_1), s(t_2) >= e^{-\|t_1 - t_2\|/\delta}$$

where δ is some scaling factor.

In general

$$< \sum c_i s(t_i), \sum d_j s(r_j) >= \sum c_i d_j e^{-\|t_i - r_j\|/\delta}$$

We should check that this is an inner product. Suppose $u = \sum c_i s(t_i)$. In order to show that $< u, u > \geq 0$, define $F(u)$ to be $\sum c_i e^{t_i - t} h(t_i)$, where $h(t_i)$ is the function of t which is equal to zero for $t < t_i$, and is equal to 1 for $t \geq t_i$. We may think of $F(u)$ as a hypothetical post synaptic response to weighted time series u. For simplicity, set time scale $\delta = 1$.

$$\int_{-\infty}^{\infty} F(s(t_1))F(s(t_2))dt = \int_{max(t_1,t_2)}^{\infty} e^{t_1 + t_2 - 2t} dt = 2 < s(t_1), s(t_2) >$$

In general $\int_{-\infty}^{\infty} F(u)F(v)dt = 2 < u, v >$. Since $\int_{-\infty}^{\infty} F(u)^2 dt = 0$ if and only if $u = 0$, we get conditions 4) and 5) above.

From an intuitive point of view $< u, v >$ measures correlation of u and v.

From this we get $norm(w) = \sqrt{(} < w, w >)$, $d(u, w) = norm(u - w)$, which gives us a metric on time series. Following the discussion above, we may think of $d(u, v)$ as a measure of the difference between hypothetical post synaptic responses to u and v. To give some idea of how this works, suppose $\delta = 1/33$, and time is measured in seconds. Then $d(s(t_1), s(t_1 + 0.01)) = 0.75$ approximately.

We also get u is orthogonal to w if and only if $< u, w > = 0$.

Additionally we get $Proj_w(u) = (< u, w > / < w, w >)w$. This is the projection of u onto direction w. This may be understood as the best approximation to u which can be expressed as a multiple of w.

Example 1 *Take time scale* $\delta = 1$. *Suppose* $w_1 = s(1) + s(2), w_2 = s(2) + s(3), w_3 = s(1) + s(3), u = s(2)$. *Then* $Proj_{w_1}(u) =< u, w_1 > / < w_1, w_1 > w_1 = s(1)/2 + s(2)/2$. *We note that, as expected,* $u - Proj_{w_1}(u)$ *is orthogonal to* $Proj_{w_1}(u)$. *We can use the Gram Schmidt process as usual to find an orthogonal basis for the subspace spanned by* (w_1, w_2, w_3). *Once we have this orthogonal basis, we can, as usual, find the best approximation in the subspace to any given element of* V.

3 Approximation

We now get some solutions to problems 1 and 2.

3.1 Gram Schmidt Solution to Problem 1

Use Gram Schmidt process on $w_1, \ldots, w_k$ to get an orthogonal basis for the subspace $Span(w_1, \ldots, w_k)$. Suppose this orthogonal basis is $w_1*, ...w_m*$. We can find the best approximation to g in this subspace by

$$\sum c_i w_i *$$

where $c_i = proj_{w_i*}(g)$

This is guaranteed to give the optimal solution to problem 1), i.e. the unique linear combination in the subspace which is closest to the goal.

3.2 Iterative Approximate Solution to Problem 1

Define $E = g - inp(G)$. Until $norm(E)$ is small, loop:

Pick i at random. Define $ch(c_i) := Proj_{w_i} E$. Let $c_i := c_i + ch(c_i)$. Then $inp(G) := inp(G) + ch(c_i)w_i$.

3.3 Iterative Approximate Solution to Problem 2

Define $E = g - out(G)$. Until $norm(E)$ is small, loop:

Pick i at random. Define $ch(c_i) := norm(Proj_{w_i} E)/norm(w_i)$. Let $c_i := c_i + ch(c_i)$. Then $inp(G) := inp(G) + ch(c_i)w_i$.

4 Testing

The following tests were performed using the iterative algorithm, outlined in sections 3.2 and 3.3. The purpose of the first set of tests is to demonstrate the ability of the algorithm to alter weight values $c_1, ..., c_k$ such that *inp(G)* becomes close to a required goal time series, *g*. We are attempting to bring the distance - as defined by *norm(g-inp(G))* - between the goal and the input vector to a minimum.

For each experiment, the goal and each time series that make up *inp(G)* consist of 10 spikes that have been randomly drawn from a uniform distribution in the interval (0,1). The initial values of the weights $c_1, ..., c_k$ are set to zero. All experiments are performed in CSIM, more details of which can be found at *www.lsm.tugraz.at/csim/index.html*.

Figure 1A shows a plot of the distance between the goal and *inp(G)*, with respect to the number of iterations of the algorithm, where *inp(G)* consists of just 10 input channels. In this experiment we used a time scale of 1/33; so, $< s(t_1), s(t_2) >= e^{-33|t_1 - t_2|}$. It can be clearly seen that initially, the distance falls sharply by a small amount before leveling off. The reason for this is simply

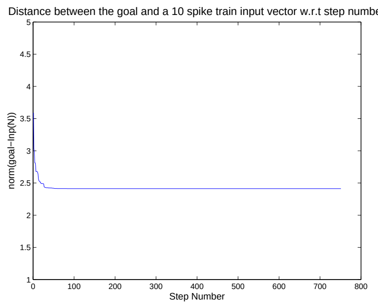

Figure: 1A

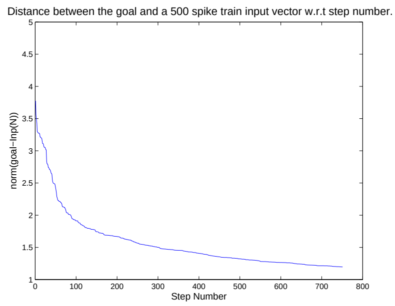

Figure: 1B

Fig. 1: Distance between the goal and the input vector w.r.t step number, with *inp(G)* consisting of 10 (Fig: 1A) and 500 (Fig1B) spike trains

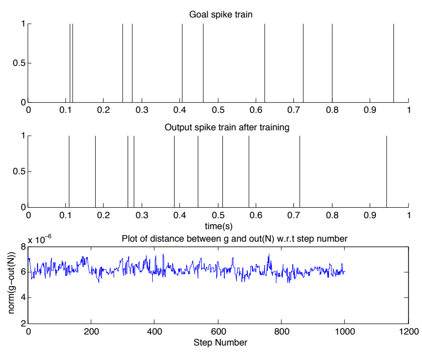

Figure: 2A

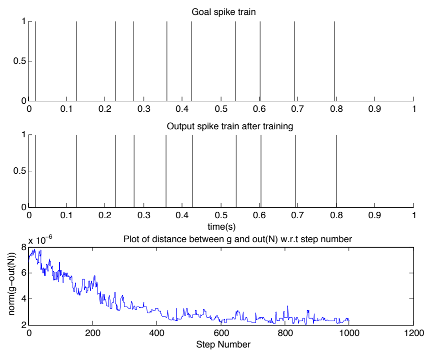

Figure: 2B

Fig. 2: Randomly generated goal spike trains and the trained output produced by a LIF neuron and the associated distance between them during training with *inp(G)* consisting of 10 (Fig: 2A) and 500 (Fig: 2B) spike trains.

that any initial adjustment to the weight of an input channel is likely to have the effect of decreasing the distance by a relatively large amount.

Figure 1B is a plot of the same variables, but with *inp(G)* consisting of 500 input channels. These figures clearly demonstrate that the iterative algorithm steadily reduces the distance between our goal and our input. Additionally it is clear that more input channels i.e. more spikes, produce a much better approximation to our goal.

The second set of experiments is designed to apply the iterative training algorithm to alter the input weights $c_1, ..., c_k$ of a spiking neuron which, receives *inp(G)* as an input, to produce our goal time series as an output.

The neuron used, G, is a basic Leaky Integrate and Fire (LIF) neuron, with capacitance, resistance, threshold and resting potential equal to 3.03×10^{-8} F, 10^6 Ohm, -0.045 V, -0.06 V respectively. The time scale of $1/33$ was used to match the time constant on which G operates. Similar results were obtained with different time scales and parameters.

In the top segment of figure 2A we can see our randomly generated goal time series and in the second segment is the spiking output of the LIF neuron - see [1] - after training. The neuron output is now somewhat similar to our goal, but not very. This is linked to the fact that only 10 input spike trains were used.

Figure 2B shows the result of increasing the number of input channels to 500. The much increased diversity of the spikes that populate *inp(G)* means that there is a much greater likelihood that we are able to construct our goal spike train with increased accuracy. The trained output is now extremely close to our goal.

The third segments of figures 2A and 2B illustrate the course of *norm(g-out(G))* with each iteration of the algorithm. This plot is noticeably different from the distance plots of figures 1A and 1B. The peaks are due to the adjustment of weights which then cause the neuron to fire when it is not desired, or to lose a spike where it is desired. This over adjustment is then corrected by a single sharp change or series of sharp changes. For the 500 channel case, it can be seen that the general trend is that the distance decreases as the number of iterations increases. The distance plot for the 10 channel case shows very little decrease for the same number of iterations.

It is clear that to construct the input to a spiking neuron in order to produce an accurate representation of a specific goal time series it is necessary that the input vectors be highly diverse.

5 Discussion of the Metric $d(u, v)$

One of the good properties of this metric is that it is continuous with respect to variation in times of spikes, as well variation of coefficients. We have $lim_{\epsilon \to 0}$ $d(s(t_1), s(t_2 + \epsilon)) = d(s(t_1), s(t_2))$. Also any two weighted time series u and v are connected by the straight line $(1 - x)u + xv$, as x goes from 0 to 1.

This should be contrasted with the more usual approach, which is to divide a time interval into small subintervals, and to represent a time series of spikes by a vector of zeroes and ones, the length of the vector being the number of subintervals, and the ith component of the vector being 1 if and only if a spike occurs in the ith sub-interval. We can then define some metric on these Boolean vectors, but however this is done it will not vary continuously with small perturbations in the times of the spikes. Also it is not always clear what would be a good path from one Boolean vector to another, especially if they have different numbers of spikes.

Another good feature of our metric is that it gives a unique optimal solution to Problem 1), whereas if Boolean vectors are used, the result depends on the size of the subintervals.

We note that the metric we propose is not new, but is similar, for example, to the metric used in [3].

Let $V(I)$ be vector space V with spikes confined to closed bounded interval I. Let $V(I, k, \Delta)$ be $V(I)$ with all coefficients having absolute value no more than k, and with spikes no closer to each other than Δ.

As usual we define compactness of a space to mean that any infinite sequence has a subsequence which tends to a limit.

Theorem 1 *$V(I, k, \Delta)$ is compact.*

Proof. Let (v_i) be an infinite sequence of weighted time series from $V(I, k, \Delta)$. We need to show that this has a limit point in $V(I, k, \Delta)$. Each v_i has the form $\sum_{i=1}^{N_i} c_{ij} s(t_{ij})$. The number of terms N_i is bounded by $length(I)/\Delta$. Since there are only finitely many possibilities for this N_i, there is an infinite subsequence of (v_i) in which N_i is constant, say N. Refine (v_i) so that this is the case. We can further refine (v_i) so that for each j, the sequences (c_{ij}) and (t_{ij}) tend to limits c_j* and t_j* respectively as i tends to infinity. After refining (v_i) in this way v_i tends to

$$\sum_{j=1}^{N} c_j* \; s(t_j*).$$

This implies that the firing activity of any finite collection of neurons can be represented in a finite product of compact metric spaces, which is therefore also compact.

The significance of this for learning is that any continuous real objective function achieves its optimal value on a compact set.

6 Discussion

The two problems considered above are subproblems of the more difficult problem of constructing given stimulus-response patterns. This can be modelled in the following way. Let S be a time series of spikes on an interval $[a, b]$, and let R be a time series on a subsequent interval $[b, c]$. Suppose that we give S to every neuron in a network. The problem would be to pick the weights to the output in such a way that we get response R at the output of our output neuron. This should be true for a typical internal state of the network. That is to say, in accordance with the ideas of anytime computing, we are not allowed to prepare the network by setting the internal state at the moment when the stimulus S is given. See [2], [3].

References

[1] W. Gerstner, W. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, Cambridge University Press, Cambridge, 2002.

[2] T. Natschlager, W. Maass and H. Markram, The "liquid computer", a novel strategy for real-time computing on time series, *Special issue on Foundations of Information Processing of TELEMATIK*, 8 (1): 32-36, 2002.

[3] T. Natschlager, W. Maass 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.