Genralized Distance with Optimal Lag

Edmund Butler (edmundbutler3@comcast.net)

12/06/22

1 Introduction

The generalized van Rossum distance is defined as follows.

As before, let $S = \{s_i\}, i = 1, ..M$ and $T = \{t_j\}, j = 1, ..N$ be the increasing times of two spike trains and let τ be the characteristic time for the response to a spike.

Let $P = \{p_i\}, i = 1, ..M$ and $Q = \{q_i\}, i = 1, ..N$ be associated positive weights. The p_i and q_i values may depend on the times in a translation independent manner, for example $p_i = f_i(s_2 - s_1, ..., s_M - s_{N-1})$ and $q_i = g_i(t_2 - t_1, .., t_N - t_{N-1})$ for positive functions $f_i()$ and $g_i()$.

The transform of the spike train $S = \{s_i\}$ is a real function $\mathfrak{R}_{\tau,S,P}$ defined by Equation 1 below, where τ is a positive parameter.

$$\mathfrak{R}_{\tau,S,P}(u) = \sum_{i=1}^{M} p_i e^{-(u-s_i)/\tau} \mathbf{1}(u > s_i) \tag{1}$$

This expression decays exponentially in u, so is in L^p for any p. The generalized distance, D, is the L^2distance between $\mathfrak{R}_S$ and $\mathfrak{R}_T$.

$$D^2 = \int_{-\infty}^{\infty} (\mathfrak{R}_{\tau,S}(u) - \mathfrak{R}_{\tau,T}(u))^2 du \tag{2}$$

The lagged distance is

$$D^2(c) = \int (\mathfrak{R}_S(u) - \mathfrak{R}_T(u - c))^2 du \tag{3}$$

Then the optimal lag, c_{opt}, minimizes $D(c)$ and the optimal distance is $D(c_{opt})$. This document develops a new algorithm for finding the exact value of c_{opt} and for computing $D(c_{opt})$.

Some special cases of interest:

1. $p_i = 1$ - the standard Van Rossum distance
2. $p_i = s_{i+1} - s_i$ - an interval-based metric with useful properties
3. $p_i = \rho_i$ where ρ_i is a weight, perhaps a probability that the associated spike is valid.

2 The Optimal Lag

This section proves that there are only finitely many possible choices for c_{opt} and develops an efficient algorithm for finding it. Without loss of generality, $\tau = 1$ is assumed.

2.1 The Theorem

The main theorem asserts that there are only a finite number of possibilities for c_{opt}, obtained by matching a particular s_i with a particular t_j. Some formulas developed during the proof of the theorem form the basis for the algorithms.

2.1.1 Lemma 1

Let $\mathfrak{R}_S()$ and $\mathfrak{R}_T()$ be as above, then

$$\int \mathfrak{R}_S(u)\mathfrak{R}_T(u)du = {}^1\!/_2 \sum_{i,j} p_i q_j e^{-|s_i - t_j|} \tag{4}$$

Proof
Multiply the product of sums term by term:
$\mathfrak{R}_S(u)\mathfrak{R}_T(u) = \sum_{i,j} p_i q_j e^{-(2u-s_i-t_j)}\mathbf{1}(u > \max(s_i, t_j))$
Take the integral inside the sum and use:
$\int_{t=a}^{\infty} e^{-2(u-b)}du = {}^1\!/_2 e^{-2(a-b)}$
Substitute $b = (s_i+t_j)/2$ and $a = \max(s_i, t_j)$, and use $\max(s, t)-(s+t)/2 = |s-t|/2$:
$\int \mathfrak{R}_S(u)\mathfrak{R}_T(u)du = {}^1\!/_2 \sum_{i,j} p_i q_j e^{-|s_i-t_j|}\Box$

2.1.2 Corollary

$$\int \mathfrak{R}_S^2 = \sum_{1\le i\le j\le N} p_i p_j e^{s_i - s_j} - {}^1\!/_2 \sum_{i=1}^{N} p_i^2 \tag{5}$$

Proof
Substitute $\mathfrak{R}_S$ for $\mathfrak{R}_T$ in Equation 4 and rearrange terms.$\Box$

2.1.3 Lemma 2

Let x_i and w_i $i = 1, ..K$ be reals with $w_i > 0$, then $\sum_{i=1}^{K} w_i e^{-|x_i-c|}$ is maximized over all reals by $c = x_{i_0}$ for some $i_0 \le$K.
Proof
Consider $F(c) = \sum_{i=1}^{N} w_i e^{-|x_i-c|}$.
$F()$ is positive, continuous and tends to 0 at $\pm\infty$, so it must have a maximum.
For any c not an x_i, $F''(c) = F(c) > 0$. So no such c can be a maximum. The result follows.$\Box$

2.1.4 Theorem

The minimum for $D(c)$ is taken on by some c of the form $s_i - t_j$.

Proof:

We can write

$$D^2(c_{opt}) = \int \Re_S^2 + \int \Re_T^2 - 2\max(c) \int \Re_S(u)\Re_T(u-c)du \qquad (6)$$

so $c_{opt} = \text{argmax}(c) \int \Re_S(u)\Re_T(u-c)du$, which by Lemma 1 is $\text{argmax}(c) \sum p_i q_j e^{-|s_i - t_j - c|}$, so Lemma 2 shows $c_{opt} = s_i - t_j$ for some i and j.$\square$

2.2 The Algorithms

This section develops recursive algorithms to calculate the terms in Equation 6.

2.2.1 Algorithm 1 - WLnormSQ

The goal is to calculate $\int \Re_S^2$ using the representation in Equation 5. Rewrite the double summation:

$$\sum_{1 \le i \le j \le N} p_i p_j e^{s_i - s_j} = \sum_{k=1}^{N} p_k A_k ; A_k = \sum_{j=1}^{k} p_j e^{s_i - s_k}$$

The A_k values are calculated by a stable recursion.

Algorithm WLnormSQ
Input: S - vector of increasing times
Output: $\int \Re_S(u)^2 du$
Step 1. Compute $A_k = \sum_{i \mid i \le k} p_i e^{s_i - s_k}$ for $k = 1, ...N$ by the recursion:
$A_1 = p_1$,$A_{k+1} = p_{k+1} + A_k e^{s_k - s_{k+1}}$
Step 2. Return $\sum_{k=1}^{N} (p_k A_k - p_k^2/2)$

2.2.2 Algorithm 2 - WLcorr

The goal is to calculate $\max(c) \int \Re_S(u)\Re_T(u-c)du$, together with the maximizing c, using the representation in Equation 4. First we sort the set of all differences between s and t. This defines the nondecreasing sequence $X = x_1, ..., x_{MN} = sort(\{s_i - t_j\})$, .and the associated weights $W = w_1, .., w_{MN} = \{p_i q_j\}$ in the same order as X. Equation 4 becomes

$$2 \int \Re_S(u)\Re_T(u-c)dt = \sum_{i=1}^{MN} w_i e^{-|x_i - c|}$$

Since $c \in$X, any possible maximum has the form $c = x_{i_c}$, so

$$2 \int \Re_S(u)\Re_T(u-c)du = \sum_{i \le i_c} w_i e^{x_i - c} - w_{i_c} + \sum_{i \ge i_c} w_i e^{c - x_i} \qquad (7)$$

If we define

$$A_k = \sum_{i|i\le k} w_i e^{x_i - x_k}, B_k = \sum_{i|i\ge k} w_i e^{x_k - x_i}$$

it is sufficient to maximize $A_k + B_k$, then $c_{opt} = x_{k_{max}}$.

This is essentially a quadratic algorithm (really $\mathrm{O}(MN\log(M+N))$ because of the sort.) So it is more expensive than the fast algorithm without a lag (linear), but if you need the lag, it is less expensive than trying out all MN possible lags (cubic.)

Algorithm WLcorr
Inputs: Two increasing sequences, S and T.
Outputs: The optimal lag, C, and the correlation Corr=$\int \mathfrak{R}_S(u)\mathfrak{R}_T(u-C)$
Step 1: Compute $\{x_k\}$ =sort($\{s_i - t_j \| i \le M, j \le N\}$) and let $\{w_k\} = \{p_i q_j\}$ in the same order as $\{x_k\}$.
Step 2: Compute $A_k = \sum_{i=1}^{k} w_i e^{x_i - x_k}$ for k=1,...,MN by the recursion:
$A_1 = w_1$,$A_{k+1} = w_{k+1} + e^{x_k - x_{k+1}} A_k$
Step 3: Compute $B_k = \sum_{i=k}^{MN} w_i e^{x_k - x_i}$ for $k = MN, ..., 1$ by the recursion:
$B_{MN} = w_{MN-1}$,$B_{k-1} = w_{k-1} + e^{x_{k-1} - x_k} B_k$
Step 4: Compute $k_{max} = \mathrm{argmax}(k)(A_k + B_k)$
$C = x_{k_{max}}$
$Corr = (A_{k_{max}} + B_{k_{max}} - w_{k_{max}})/2$

2.2.3 Algorithm 3 - WLfastCorr

The algorithm WLcorr automatically applies the optimal lag, but it can be useful to compute a correlation without lag, especially if that computation is much faster. Algorithm WLfastCorr computes $\int \mathfrak{R}_S(u)\mathfrak{R}_T(u)du$ in time O(M+N).

The main idea is to precompute partial sums of exponentials in the t direction. With care to account for $s_i == t_j$ only once, write Equation 4 as

$$\int \mathfrak{R}_S \mathfrak{R}_{\mathfrak{T}} = {}^1\!/_2 \Big(\sum_{i,j|s_i<t_j} p_i q_j e^{s_i - t_j} + \sum_{i,j|s_i\ge t_j} p_i q_j e^{t_j - s_i} \Big) \triangleq {}^1\!/_2 (U + V) \qquad (8)$$

Calculate $U_i = \sum_{j|t_j>s_i} q_j e^{s_i - t_j}$ and $V_i = \sum_{j|t_j\le s_i} q_j e^{t_j - s_i}$ by the following recurrences.

Algorithm WLfastCorr
Input: Two increasing sequences S and T.
Output: The correlation $\int \mathfrak{R}_S \mathfrak{R}_T$
Step 1: Calculate U_k by the recurrence:

$$U_N = \sum_{j|t_j > s_N} q_j e^{s_N - t_j}; U_{k-1} = e^{s_{k-1} - s_k} U_k + \sum_{j|s_{k-1} < t_j \leq s_k} q_j e^{s_{k-1} - t_j}$$

Step 2: Similarly, calculate V_k by the recurrence:

$$V_1 = \sum_{j|t_j \leq s_1} q_j e^{t_j - s_1}; V_{k+1} = e^{s_k - s_{k+1}} V_k + \sum_{j|s_k \leq t < s_{k+1}} q_j e^{t_j - s_{k+1}}$$

Step 3: Return $1/2 \sum p_k (U_k + V_k)$

2.3 Implementation

A C++ implementation of these algorithms is available at https://github.com/edmundbutler/vanrossumlag.

The main entry point is weightedLag() with the arguments:

```
bool weightedLag(sIn, tIn, tau,
                 sNorm, tNorm,
                 corr, lag)
Inputs: sIn, tIn - the spike trains
        tau - the scale
Outputs: sNorm, tNorm - L2 norms of the transforms
         corr - the maximized correlation
         lag - the maximizing lag
```

The outputs may be used to calculate the lagged distance

$$D^2 = sNorm^2 + tNorm^2 - 2 * corr$$

or the correlation coefficient

$$CC = corr/(sNorm * tNorm)$$

.

Also, the distance can be normalized by factors of $\alpha = 1/\sum p_i$ and $\beta = 1/\sum q_i$ so the distance is given by $D^2_{norm} \triangleq \int (\alpha\, \mathfrak{R}_S(u) - \beta\, \mathfrak{R}_T(u)) du$.

$$D^2_{norm} = \alpha^2\, sNorm^2 + \beta^2\, tNorm^2 - 2\alpha\beta\, corr \tag{9}$$