An Optimal Linear Time Algorithm for Quasi-Monotonic Segmentation

Daniel Lemire
University of Quebec at Montreal (UQÀM)
100 Sherbrooke West
Montréal, Qc, Canada, H2X 3P2
lemire@ondelette.com

Martin Brooks, Yuhong Yan
National Research Council of Canada
1200 Montreal Road
Ottawa, ON, Canada, K1A 0R6
First.Last@nrc.gc.ca

Abstract

Monotonicity is a simple yet significant qualitative characteristic. We consider the problem of segmenting an array in up to K segments. We want segments to be as monotonic as possible and to alternate signs. We propose a quality metric for this problem, present an optimal linear time algorithm based on novel formalism, and compare experimentally its performance to a linear time top-down regression algorithm. We show that our algorithm is faster and more accurate. Applications include pattern recognition and qualitative modeling.

1 Introduction

Monotonicity is one of the most natural and important qualitative properties for sequences of data points. It is easy to determine where the values are strictly going up or down, but we only want to identify significant monotonicity. For example, the drop from 2 to 1.9 in the array $0,1,2,1.9,3,4$ might not be significant and might even be noise-related. The quasi-monotonic segmentation problem is to determine where the data is approximatively increasing or decreasing.

We present a metric for the quasi-monotonic segmentation problem called the Optimal Monotonic Approximation Function Error (OMAFE); this metric differs from previously introduced OPMAFE metric [2] since it applies to all segmentations and not just "extremal" segmentations. We formalize the novel concept of a maximal $*$-pair and shows that it can be used to define a unique labelling of the extrema leading to a novel optimal segmentation algorithm. We also present an optimal linear time algorithm to solve the quasi-monotonic segmentation problem given a segment budget together with an experimental comparison to quantify the benefits of our algorithm.

ICDM05, New Orleans, Louisiana, U.S.A., Nov. 27-30 2005, (NRC Code 48277).

2 Monotonicity Error Metric (OMAFE)

Suppose n samples noted $F : D = \{x, \ldots, x_n\} \rightarrow \mathbb{R}$ with $x_1 < x_2 < \ldots x_n$. We define, $F_{|[a,b]}$ as the restriction of F over $D \cap [a,b]$. We seek the best monotonic (increasing or decreasing) function $f : \mathbb{R} \rightarrow \mathbb{R}$ approximating F. Let $\Omega_\uparrow$ (resp. $\Omega_\downarrow$) be the set of all monotonic increasing (resp. decreasing) functions. The **Optimal Monotonic Approximation Function Error (OMAFE)** is $\min_{f \in \Omega} \max_{x \in D} |f - F|$ where Ω is either $\Omega_\uparrow$ or $\Omega_\downarrow$.

The segmentation of a set D is a sequence $S = X_1, \ldots, X_K$ of intervals in $\mathbb{R}$ with $[\min D, \max D] = \bigcup_i X_i$ such that $\max X_i = \min X_{i+1} \in D$ and $X_i \cap X_j = \emptyset$ for $j \neq i+1, i, i-1$. Alternatively, we can define a segmentation from the set of points $X_i \cap X_{i+1} = \{y_{i+1}\}$, $y_1 = \min X_1$, and $y_{K+1} = \max X_K$. Given $F : \{x_1, \ldots, x_n\} \rightarrow \mathbb{R}$ and a segmentation, the Optimal Monotonic Approximation Function Error (OMAFE) of the segmentation is $\max_i \mathrm{OMAFE}(F_{|X_i})$ where the monotonicity type (increasing or decreasing) of the segment X_i is determined by the sign of $F(\max X_i) - F(\min X_i)$. Whenever $F(\max X_i) = F(\min X_i)$, we say the segment has no direction and the best monotonic approximation is just the flat function having value $(\max F_{|X_i} - \min F_{|X_i})/2$. The error is computed over

each interval independently and so, optimal monotonic approximation functions are not required to agree at $\max X_i = \min X_{i+1}$. Segmentations should alternate between increasing and decreasing, otherwise sequences such as $0,2,1,0,2$ can be segmented as two increasing segments $0,2,1$ and $1,0,2$: we consider it is natural to aggregate segments with the same monotonicity.

We solve for the best monotonic function as follows. If we seek the best monotonic increasing function, we first define $\overline{f}_{\uparrow}(x) = \max\{F(y) : y \leq x\}$ (the maximum of all previous values) and $\underline{f}_{\uparrow}(x) = \min\{F(y) : y \geq x\}$ (the minimum of all values to come). If we seek the best monotonic decreasing function, we define $\overline{f}_{\downarrow}(x) = \max\{F(y) : y \geq x\}$ (the maximum of all values to come) and $\underline{f}_{\downarrow}(x) = \min\{F(y) : y \leq x\}$ (the minimum of all previous values). These functions, which can be computed in linear time, are all we need to solve for the best approximation function as shown by the next theorem which is a well-known result [5].

Theorem 1 *Given $F : D = \{x_1, \ldots, x_n\} \to \mathbb{R}$, a best monotonic increasing approximation function to F is $f_{\uparrow} = (\overline{f}_{\uparrow} + \underline{f}_{\uparrow})/2$ and a best monotonic decreasing approximation function is $f_{\downarrow} = (\overline{f}_{\downarrow} + \underline{f}_{\downarrow})/2$. The corresponding error (OMAFE) is $\max_{x \in D}(|\overline{f}_{\uparrow}(x) - \underline{f}_{\uparrow}(x)|)/2$ or $\max_{x \in D}(|\overline{f}_{\downarrow}(x) - \underline{f}_{\downarrow}(x)|)/2$ respectively.*

3 A Scale-Based Algorithm for Quasi-Monotonic Segmentation

We use the following proposition to prove that the segmentations we generate are optimal (see Theorem 2).

Proposition 1 *A segmentation $y_1, \ldots, y_{K+1}$ of $F : D = \{x_1, \ldots, x_n\} \to \mathbb{R}$ with alternating monotonicity has a minimal OMAFE ε for a number of alternating segments K if*

A. $F(y_i) = \max F([y_{i-1}, y_{i+1}])$ or $F(y_i) = \min F([y_{i-1}, y_{i+1}])$ for $i = 2, \ldots, K$;

B. in all intervals $[y_i, y_{i+1}]$ for $i = 1, \ldots, K$, there exists z_1, z_2 such that $|F(z_2) - F(z_1)| > 2\varepsilon$.

For simplicity, we assume F has no consecutive equal values, i.e. $F(x_i) \neq F(x_{i+1})$ for $i = 1, \ldots, n-1$; our algorithms assume all but one of consecutive equal values values have been removed. We say x_i is a maximum if $i \neq 1$ implies $F(x_i) > F(x_{i-1})$ and if $i \neq n$ implies $F(x_i) > F(x_{i+1})$. Minima are defined similarly.

Our mathematical approach is based on the concept of δ-pair:

Definition 1 *The tuple x, y ($x < y \in D$) is a δ-pair (or a pair of scale δ) for F if $|F(y) - F(x)| \geq \delta$ and for all $z \in D$, $x < z < y$ implies $|F(z) - F(x)| < \delta$ and $|F(y) - F(z)| < \delta$. A δ-pair's direction is increasing or decreasing according to whether $F(y) > F(x)$ or $F(y) < F(x)$.*

δ-Pairs having opposite directions cannot overlap but they may share an end point. δ-Pairs of the same direction may overlap, but may not be nested. We use the term "$*$-pair" to indicate a δ-pair having an unspecified δ. We say that a $*$-pair is significant at scale δ if it is of scale δ' for $\delta' \geq \delta$.

We define δ-monotonicity as follows:

Definition 2 *Let X be an interval, F is δ-monotonic on X if all δ-pairs in X have the same direction; F is strictly δ-monotonic when there exists at least one such δ-pair. In this case:*

- *F is δ-increasing on X if X contains an increasing δ-pair.*
- *F is δ-decreasing on X if X contains a decreasing δ-pair.*

A δ-monotonic interval X satisfies $\mathrm{OMAFE}(X) < \delta/2$. We say that a $*$-pair x, y is **maximal** if whenever z_1, z_2 is a $*$-pair of a larger scale in the same direction containing x, y, then there exists a $*$-pair w_1, w_2 of an opposite direction contained in z_1, z_2 and containing x, y. For example, the sequence $1,3,2,4$ has 2 maximal $*$-pairs: $1,4$ and $3,2$. Maximal $*$-pairs of opposite direction may share a common point, whereas maximal $*$-pairs of the same direction may not. Maximal $*$-pairs cannot overlap, meaning that it cannot be the case that exactly one end point of a maximal $*$-pair lies strictly between the end points of another maximal $*$-pair; either neither point lies strictly between or both do. In the case that both do, we say that the one maximal $*$-pair properly contains the other. All $*$-pairs must be contained in a maximal $*$-pair.

Lemma 1 *The smallest maximal $*$-pair containing a $*$-pair must be of the same direction.*

Our approach is to label each extremum in F with a scale parameter δ saying that this extremum is "significant" at scale δ and below. Our intuition is that by picking extrema at scale δ, we should have a segmentation having error less than $\delta/2$.

Definition 3 *The scale labelling of an extremum x is the maximum of the scales of the maximal $*$-pairs for which it is an end point.*

For example, given the sequence $1,3,2,4$ with 2 maximal $*$-pairs ($1,4$ and $3,2$), we would give the following labels in order $4,1,1,4$.

Definition 4 *Given $\delta > 0$, a maximal alternating sequence of δ-extrema $Y = y_1 \dots y_{K+1}$ is a sequence of extrema each having scale label at least δ, having alternating types (maximum/minimum), and such that there exists no sequence properly containing Y having these same properties. From Y we define a maximal alternating δ-segmentation of D by segmenting at the points $x_1, y_2 \dots y_K, x_n$.*

Theorem 2 *Given $\delta > 0$, let $P = S_1 \dots S_K$ be a maximal alternating δ-segmentation derived from maximal alternating sequence $y_1 \dots y_{K+1}$ of δ-extrema. Then any alternating segmentation Q having OMAFE(Q) $<$ OMAFE(P) has at least $K+1$ segments.*

Sequences of extrema labelled at least δ are generally not maximal alternating. For example the sequence $0,10,9,10,0$ is scale labelled $10,10,1,10,10$. However, a simple relabelling of certain extrema can make them maximal alternating. Consider two same-sense extrema $z_1 < z_2$ such that lying between them there exists no extremum having scale at least as large as the minimum of the two extrema's scales. We must have $F(z_1) = F(z_2)$, since otherwise the point upon which F has the lesser value could not be the endpoint of a maximal $*$-pair. This is the only situation which causes choice when constructing a maximal alternating sequence of δ-extrema. To eliminate this choice, replace the scale label on z_1 with the largest scale of the opposite-sense extrema lying between them.

3.1 Computing a Scale Labelling Efficiently

Algorithm 1 produces a scale labelling in linear time. Extrema from the original data are visited in order, and they alternate (maxima/minima) since we only pick one of the values when there are repeated values (such as $1,1,1$).

The algorithm has a main loop (lines 5 to 12) where it labels extrema as it identifies extremal $*$-pairs, and stack the extrema it cannot immediately label. At all times, the stack (line 3) contains minima and maxima in **strictly** increasing and decreasing order respectively. Also at all times, the last two extrema at the bottom of the stack are the absolute maximum and absolute minimum (found so far). Observe that we can only label an extrema as we find new extremal $*$-pairs (lines 7, 10, and 14).

- If the stack is empty or contains only one extremum, we simply add the new extremum (line 12).
- If there are only 2 extrema z_1, z_2 in the stack and we found either a new absolute maximum or new absolute minimum (z_3), we can pop and label the oldest one (z_1) (lines 9, 10, and 11) because the old pair (z_1, z_2) forms a maximal $*$-pair and thus must be bounded by extrema having at least the same scale while the oldest value (z_1) doesn't belong to a larger maximal $*$-pair. Otherwise, if there are only 2 extrema z_1, z_2 in the stack and the new extrema z_3 satisfies $z_3 \in (\min(z_1, z_2), \max(z_1, z_2))$, then we add it to the stack since no labelling is possible yet.
- While the stack contains more than 2 extrema (lines 6, 7 and 8), we consider the last three points on the stack (s_3, s_2, s_1) where s_1 is the last point added. Let z be the value of the new extrema. If $z \in (\min(s_1, s_2), \max(s_1, s_2))$, then it is simply added to the stack since we cannot yet label any of these points; we exit the while loop. Otherwise, we have a new maximum (resp. minimum) exceeding (resp. lower) or matching the previous one on stack, and hence s_1, s_2 is a maximal $*$-pair. If $z \neq s_2$, then s_3, z is a maximal $*$-pair and thus, s_2 cannot be the end of a maximal $*$-pair and s_1 cannot be the beginning of one, hence both s_2 and s_1 are labelled. If $z = s_2$ then we have successive maxima or minima and the same labelling as $z \neq s_2$ applies.

During the "unstacking" (lines 13 and following), we visit a sequence of minima and maxima forming increasingly larger maximal $*$-pairs.

Once the labelling is complete, we find $K+2$ extrema having largest scale in time $O(nK)$ using $O(K)$ memory, then we remove all extrema having the same scale as the smallest scale in these $K+2$ extrema (removing at least one), we replace the first and the last extrema by 0 and $n-1$ respectively. The result is an optimal segmentation having at most K segments.

Algorithm 1 Algorithm to compute the scale labelling in $O(n)$ time.

INPUT: an array d containing the y values indexed from 0 to $n-1$, repeated consecutive values have been removed
OUTPUT: a scale labelling for all extrema
$S \leftarrow$ empty stack, First(S) is the value on top, Second(S) is the second value
define $\delta(d,S) = |d_{\text{First}(S)} - d_{\text{Second}(S)}|$
for e index of an extremum in d, e's are visited in increasing order **do**
  while length(S) > 2 and (e is a minimum such that $d_e \leq \text{Second}(S)$ or e is a maximum such that $d_e \geq \text{Second}(S)$) **do**
    label First(S) and Second(S) with $\delta(d,S)$
    pop stack S twice
  if length(S) is 2 and (e is a minimum such that $d_e \leq \text{Second}(S)$ or e is a maximum such that $d_e \geq \text{Second}(S)$) **then**
    label Second(S) with $\delta(d,S)$
    remove Second(S) from stack S
  stack e to S
while length of $S > 2$ **do**
  label First(S) with $\delta(d,S)$
  pop stack S
label First(S) and Second(S) with $\delta(d,S)$

4 Experimental Results and Comparison to Top-Down Linear Spline

We compare our optimal $O(nK)$ algorithm with the top-down linear spline algorithm [4] which runs in $O(nK^2)$ time. It successively segments the data starting with only one segment, each time picking the segment with the worse linear regression error and finding the best segmentation point; the linear regression is not continuous from one segment to the other. The regression error can be computed in constant time if one has precomputed the range moments [3]. We run through the segments and aggregate consecutive segments having the same sign where the sign of a segment $[y_k, y_{k+1}]$ is defined by $F(y_{k+1}) - F(y_k)$.

4.1 Data Source

We used samples from the MIT-BIH Arrhythmia Database [1]. These ECG recordings used a sampling rate of 360 samples per second per channel with 11-bit resolution. We keep 4000 samples (11 seconds) and about 14 pulses, and we do no preprocessing such as baseline correction. We can estimate that a typical pulse has about 5 "easily" identifiable monotonic segments. Hence, out of 14 pulses, we can estimate that there are about 70 significant monotonic segments, some of which match the domain-specific markers (reference points P, Q, R, S, and T). A qualitative description of such data is useful for pattern matching applications, for example.

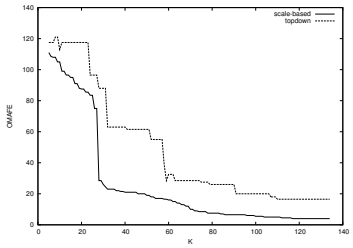

Figure 1. Results from experiments over ECG data: the optimal algorithm is considerably more accurate.

4.2 Results

We implemented both the scale-based segmentation algorithm and the L_2 norm top-down linear spline approximation algorithm in Python(version 2.3). Each run was repeated 3 times and we observed that the scale-based segmentation implementation is faster than the top-down linear spline approximation implementation by a factor of 10.

The OMAFE with respect to the maximal number of segments (K) is given in Fig. 1. By counting on about 5 monotonic segments per pulse with a total of 14 pulses, there should about 70 monotonic segments in the 4000 samples under consideration. We see that the decrease in OMAFE with the addition of new segments starts to level off between 50 and 70 segments as predicted.

References

[1] A. L. Goldberger *et al.* PhysioBank, PhysioToolkit, and PhysioNet. *Circulation*, 101(23):215–220, 2000.

[2] M. Brooks, Y. Yan, and D. Lemire. Scale-based monotonicity analysis in qualitative modelling with flat segments. In *IJCAI'05*, 2005.

[3] D. Lemire. Wavelet-based relative prefix sum methods for range sum queries in data cubes. In *CASCON*. IBM, October 2002.

[4] T. Palpanas, M. Vlachos, E. Keogh, D. Gunopulos, and W. Truppel. Online amnesic algorithm of streaming time series. In *ICDE*, 2004.

[5] V. A. Ubhaya. Isotone optimization I. *Approx. Theory*, 12:146–159, 1974.