2-D BINARY LOCALLY MONOTONIC REGRESSION

Alfredo Restrepo (Palacios) and Scott T. Acton†

Dpt. Ing. Eléctrica y Electrónica
Universidad de los Andes
A.A. 4976, Santafé de Bogotá, Colombia
arestrep@uniandes.edu.co

†School of Electrical end Computer Engineering
202 Engineering South
Stillwater, Oklahoma 74078 USA
sacton@okstate.edu

ABSTRACT

We introduce binary locally monotonic regression as a first step in the study of the application of local monotonicity for image estimation. Given an algorithm that generates a similar locally monotonic image from a given image, we can specify both the scale of the image features retained and the image smoothness. In contrast to the median filter and to morphological filters, a locally monotonic regression produces the optimally similar locally monotonic image. Locally monotonic regression is a computationally expensive technique, and the restriction to binary-range signals allows the use of Viterbi-type algorithms. Binary locally monotonic regression is a powerful tool that can be used in the solution of the image estimation, image enhancement, and image segmentation problems.

1. INTRODUCTION

Local monotonicity is a natural criterion for a performance measure for signal enhancement: impulses are not locally monotonic while edges are locally monotonic. The quality of local monotonicity first emerged in the study of root signals of the median filter. Given that the median filter is a suboptimal smoother under this criterion and that it is widely used in image processing, the study of locally monotonic (lomo) regression for images is of critical importance. A lomo regression [1] image is the closest image (under a semimetric) to a given input image that is lomo.

The main drawback of lomo regression is the high computational cost. Several efforts have been extended recently to decrease the complexity of the task [2], [3]. As Siridopoulos [3] has pointed out, the consideration of digital signals allows the use of Viterbi-type algorithms [4]. For 1-D signals, the complexity of these algorithms is linear with respect to the length of the signal.

This paper explores 2-D binary lomo regression. For an N by M binary image, we use the set of 1-D lomo signals of length N to compute a regression with a Viterbi algorithm, using a queue of lomo rows for the states in the Viterbi trellis. The resulting image is binary, lomo in the 2-D sense, and optimal in distance from the original image.

In Section 2 we compute the number of lomo signals of a given length, and indicate how to generate them. In Section 3 we give short definitions of 2-D local monotonicity and describe the algorithm used to compute the regression. Section 4 provides representative examples. The paper is concluded in Section 5.

2. 1-D BINARY LOMO-3 SIGNALS

Let $\mathbf{1}^{\mathbf{N}} = \{0, 1\}^{\{0, 1, \dots N-1\}}$ be the set of {0, 1}-valued signals of length N. The set of locally monotonic signals of degree α (lomo-α) contains signals in which each segment (contiguous subsequence) of length α is either non-increasing or non-decreasing. Let Λ denote the subset of lomo-3 binary signals; that is, signals with no segment equal to [0, 1, 0] or [1, 0, 1].

The tree in Fig. 1 indicates how the signals in Λ may be generated. Each node in the tree corresponds to a lomo-3 signal; the components of the signal are given by the labels of the nodes, read from left to right, of a path connecting the root Δ of the tree and the node in question.

The cardinality of Λ depends on the length N of the signals. After the first two columns at the root of the tree that contain the "border effects" at the left of a signal, a node branches either into one or two branches. Depending on the last two bits (at the right) of the corresponding signal, we classify the nodes in the tree into four sets: m_1, m_2, n_1 and n_2. The signals corresponding to nodes in m_1 end with two zeros; signals in m_2 end with the segment [1, 1]; signals in n_1 end with [0, 1], and signals in n_2 end with [1, 0]. Nodes in sets m_1 and m_2 branch into two branches, since either a zero or a one may be appended at the right of the signal to insure local monotonicity. Signals (binary strings) corresponding to nodes in sets n_1 and n_2 can only be grown by appending a 1 or a 0, respectively, since [0, 1, 0] and [1, 0, 1] violate monotonicity. A signal in set m_1 to which a 1 is appended becomes a signal in set n_1. If a 0 is appended, it remains in set m_1. A signal in set

m_2 to which a 0 is appended becomes a signal in set n_2. If a 1 is appended, it remains in set m_2, *etc*. This classification process is illustrated in Fig. 2.

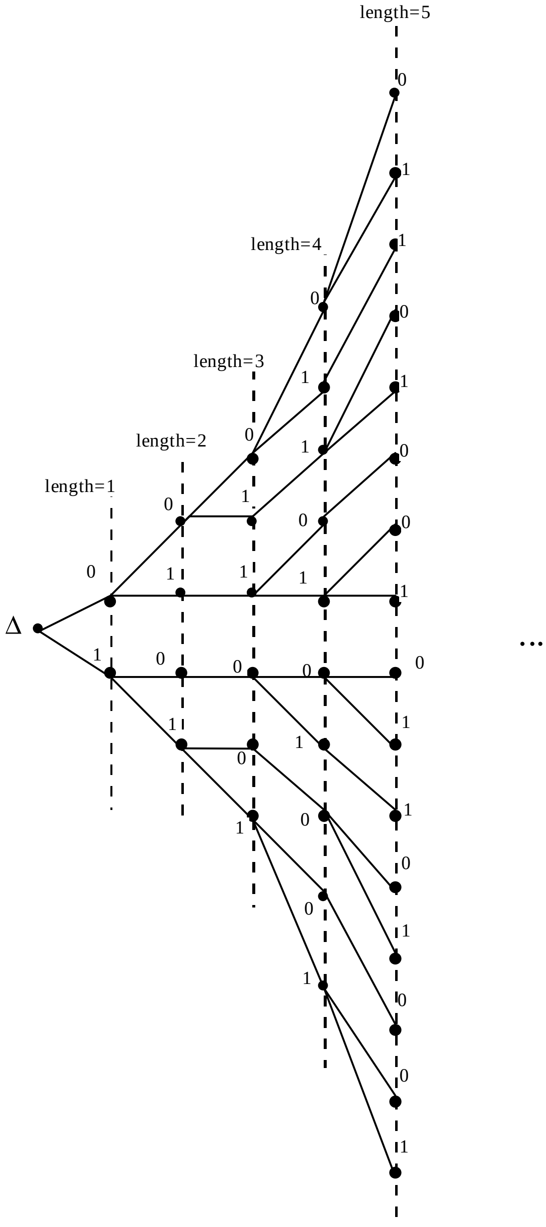


Fig. 1. A tree in which each node represents a binary lomo-3 signal.

Based on these considerations, the binary strings in each set, as the length is increased by one element, change as illustrated in Fig. 3. Let m = $m_1 + m_2$ and n = $n_1 + n_2$. After successively increasing the length of the signals under consideration the numbers evolve as illustrated in Fig. 3. A new value for m is given by the previous m + n and a new n by the previous m. As shown below, we may think of the vector $[n\ m]^T$ as evolving by the repeated application of a linear transformation.

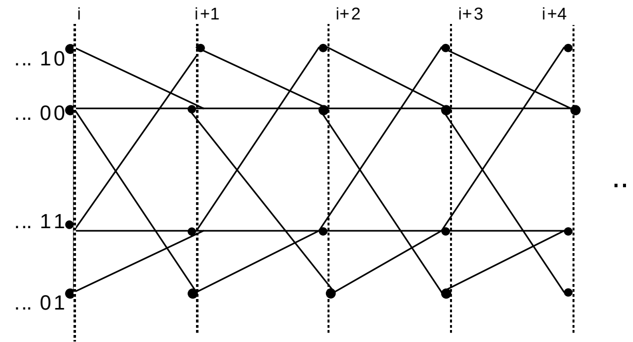


Fig. 2. Valid transitions that increase binary signal length (as elements are appended on the right).

At each stage, the vector $\begin{bmatrix} m_i \\ n_i \end{bmatrix}$ becomes the vector $\begin{bmatrix} m_i + n_i \\ m_i \end{bmatrix}$. Therefore, the linear transformation **m** $= \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ is being applied in the following manner:

$$\begin{bmatrix} m_{i+1} \\ n_{i+1} \end{bmatrix} = \mathbf{m} \begin{bmatrix} m_i \\ n_i \end{bmatrix}. \qquad (1)$$

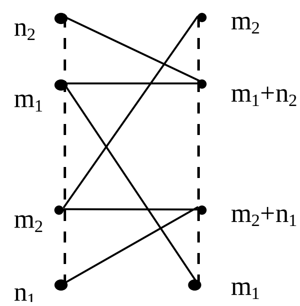


Fig. 3. A stage of an iteration shown in Fig. 4.

Since lomo regression is the process of finding the closest signal to a given signal that resides in the set of lomo signals, it is of interest to compute the cardinality of this set. Here, the number of lomo signals of length N is given by

$$\eta(N) = m_N + n_N \qquad (2)$$

starting with $\begin{bmatrix} m_2 \\ n_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$.

The matrix **m** can be decomposed as:

$$\mathbf{m} = \mathbf{p}^T \mathbf{d}\, \mathbf{p}, \qquad (3)$$

where **d** is a diagonal matrix, with the eigenvalues of **m** on the diagonal and **p** is unitary. Since,

$$\mathbf{m}^r = \mathbf{p}^T \mathbf{d}^r\, \mathbf{p} \qquad (4)$$

where

$$\mathbf{p} = \frac{1}{2}\begin{bmatrix} \sqrt{1+\frac{1}{\sqrt{5}}} & \sqrt{1-\frac{1}{\sqrt{5}}} \\ -\sqrt{1-\frac{1}{\sqrt{5}}} & \sqrt{1+\frac{1}{\sqrt{5}}} \end{bmatrix}, \qquad (5)$$

and the eigenvalues of **m** are given by $\lambda_{1,2} = 0.5 \pm 0.5\sqrt{5}$. Then, the number of binary lomo-3 signals of length N is

$$\eta(N) = 2\,[\lambda_1^{(N-2)}\,(1-\frac{2}{\sqrt{5}}) + \lambda_2^{(N-2)}\,(1+\frac{2}{\sqrt{5}})]. \qquad (6)$$

A similar line of reasoning may be used to obtain a formula for the number of lomo-α signals given α larger than 3.

TABLE I

N	# lomo-3 signals	# signals
1	2	2
2	4	4
3	6	8
4	10	16
5	16	32
6	26	64
7	42	128
8	68	256
9	110	512
10	178	1024

Table I indicates both the cardinality of the set of lomo signals and the total number of binary signals. One may observe that the percentage of lomo signals tends to zero as the image size is increased.

3. A VITERBI-TYPE ALGORITHM FOR 2-D BINARY LOMO REGRESSION

Consider an NxM binary image, that is, an element of $\mathbf{1}^{NxM} = \{0, 1\}^{\{0, 1, \ldots N-1\}x\{0, 1, \ldots M-1\}}$ with $N \le M$. Note that we do not require images to be square.

We now clarify the definition used for *2-D local monotonicity*. In defining 2-D local monotonicity, we use the variants *strong* and *weak*. Strong local monotonicity of degree α (LOMO-α) means that the image is lomo-α in the 1-D sense along each defined direction. Weak local monotonicity of degree α (lomo-α) infers that the image is lomo-α along at least one direction at each point in the image. The number of directions used also influences the definition of local monotonicity. When the horizontal, vertical and diagonal directions are considered, we have $LOMO^*$-α images in the strong case and $lomo^*$-α images in the weak case. When only horizontal and vertical orientations are considered, we have $LOMO^+$-α and $lomo^+$-α images. For each type of local monotonicity there is a corresponding regression. A lomo regression is a lomo image that is optimally close to a given image, under a given semimetric.

For lomo-α regression (in general), each state in the trellis [4] of the Viterbi algorithm consists of an image with α–1 lomo-α rows: α–2 *previous rows* including an *oldest row*, and one *current row*, each of length N. Thus, since there are η possible 1-D lomo signals of length N, the height of the trellis is η^2. The width of the trellis is given by M–α+2.

A transition from a previous state to a destination state is valid (and has a null transition cost) if the previous rows of the destination state correspond to all but the oldest row of the previous rows in the destination state, and the addition of a new 1-D lomo row (the current row of the destination state) gives a lomo image segment, of size Nxα. Otherwise, the cost is set to be infinite, and the transition is said to be invalid.

As the trellis is traversed with the consideration of each row of the image being regressed, the paths are pruned by computing the survivor *paths* with minimal cost that arrive at each state. The state cost is given by the number of bits in which the current row of the state differ from the corresponding row of the image being regressed (the Hamming distance, for example). Once a trellis is generated, the paths of minimal cost give the regressions of the image. A "depth first" technique may be used for the fast computation of *one* regression and a "breadth first" technique for finding all regressions.

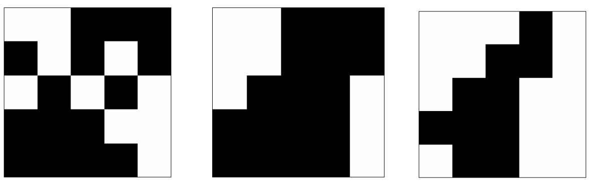
As compared to an exhaustive search technique, the Viterbi method is quite efficient. For example, on a 233MHz Pentium II computer, a 4x5 binary $LOMO^*$-3 regression required 6,465 seconds (over 1.5 hours) using exhaustive search and just 0.8 seconds using the Viterbi approach!

4. EXAMPLES

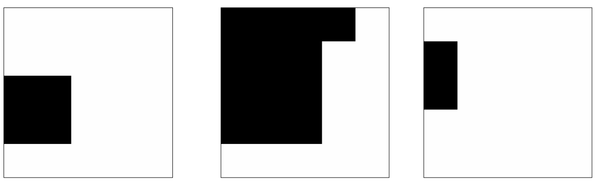
Two examples of $LOMO^*$-3 regression are presented, using the images in Fig. 4(a) and Fig. 5(a) as input images. The resultant images are $LOMO^*$-3 and resemble the respective input images (see Fig. 4(b) and Fig. 5(b)). The Hamming distances from the input images are given in Table II.

Typically, a median filter is used to produce lomo signals in 1-D. In this case, the 2-D median filter does not give a $LOMO^*$-3 image, and the error (Hamming distance) is almost twice that of the regression results in both cases. Another filter used to produce lomo signals in 1-D is the morphological open-close filter. For the 2-D examples given here, the open filter results (Fig. 4(d) and Fig. 5(d)) and the close filter results (Fig. 4(e) and Fig. 5(e)) are clearly unrepresentative of the input images. Furthermore, the open filter is biased toward the removal of isolated 1's and the close filter is biased toward the removal of isolated

0's. The open-close filter results are less biased, but are over-smoothed and bear no semblance to the original data. Thus, the lomo regression results are superior in terms of removing small features and producing a result that resembles the input imagery.



(a) (b) (c)



(d) (e) (f)

Fig. 4. A 5x5 binary image example (black = 1, white = 0). (a) Original image; (b) a LOMO*-3 regression; (c) 3x3 median filter result; (d) open result (2x2 structuring element (SE)); (e) close result (2x2 SE); (f) open-close result (2x2 SE).

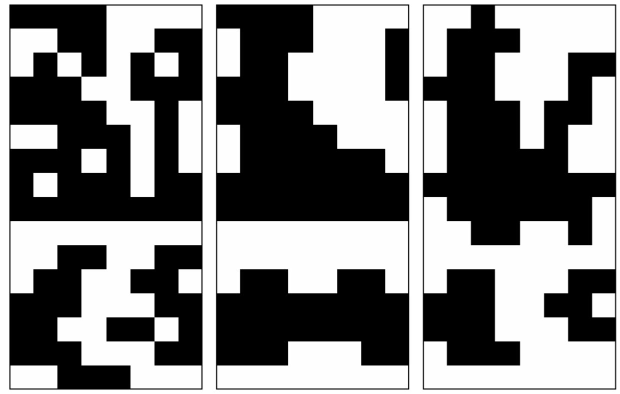
TABLE II

Processing Technique	Error 5x5 Ex.	Error 8x16 Ex.
LOMO*-3 Regression	4	28
Median Filter	9	48
Open Filter	13	63
Close Filter	12	62
Open-close Filter	15	68

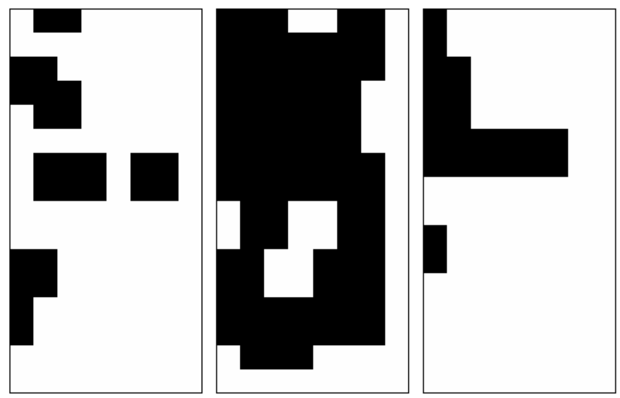
5. CONCLUSION

We have taken a first step in the solution of the problem of locally monotonic regression for images. The consideration of binary signals allows the use of a Viterbi algorithm. We are currently exploring the relationship of binary lomo regression with binary morphology with the goal of developing lomo regression as a shape-preserving, optimal smoother for images. Also, we are investigating practical lomo regression algorithms for gray level images.

Both lomo regression and the median filter can be used to denoise imagery and produce an image representation of a prescribed feature scale. Lomo regression is superior in terms of giving the closest lomo signal and removing alternating patterns due to image texture. The optimal 2-D regression approach can be used as a precursor to higher level image processing problems such as image segmentation and multiscale image coding.



(a) (b) (c)



(d) (e) (f)

Fig. 5. An 8x16 binary image example. (a) Original image; (b) a LOMO*-3 regression; (c) 3x3 median filter result; (d) open result (2x2 SE); (e) close result (2x2 SE); (f) open-close result (2x2 SE).

6. REFERENCES

[1] A. Restrepo and A.C Bovik. Locally monotonic regression. *IEEE trans. on Signal Processing,* vol. 41, pp. 2796-2810, 1993.

[2] A. Restrepo and R. de la Vega. Efficient computation of locally monotonic regression. *IEEE Signal Processing Letters,* vol. 3, pp. 263-265, 1996.

[3] N. Sidiropoulos. Fast digital locally monotonic regression. *IEEE trans. on Signal Processing,* vol. 45, pp. 389-395, 1997.

[4] H-L Lou. Implementing the Viterbi algorithm. *IEEE Signal Processing Magazine,* vol. 12, pp. 45-52, 1995.