Fast and Incremental Erosion Score Computation

Gilles Moyse, Marie-Jeanne Lesot

▶ **To cite this version:**

Gilles Moyse, Marie-Jeanne Lesot. Fast and Incremental Erosion Score Computation. IPMU 2014 - International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Jul 2014, Montpellier, France. pp.376-385, 10.1007/978-3-319-08795-5_39 . hal-00990983

HAL Id: hal-00990983
https://hal.science/hal-00990983

Submitted on 9 Jan 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Fast and Incremental Computation for the Erosion Score

Gilles Moyse and Marie-Jeanne Lesot

Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005, Paris, France
CNRS, UMR 7606, LIP6, F-75005, Paris, France
firstname.lastname@lip6.fr

Abstract The erosion score is a Mathematical Morphology tool used primarily to detect periodicity in data. In this paper, three new computation methods are proposed, to decrease its computational cost and to allow to process data streams, in an incremental variant. Experimental results show the significant computation time decrease, especially for the efficient levelwise incremental approach which is able to process a one million point data stream in 1.5s.

Keywords: Mathematical Morphology, Erosion Score, Incremental

1 Introduction

Mathematical Morphology (MM) defines a set of techniques for the analysis of spatial structures, and is widely used in image processing, understanding, segmentation or compression [10,13]. Functional MM applies its principles to function values and has been used for several types of data processing tasks, such as such as signal sieving [1,14], signal pre-processing [17], text categorisation [4], fuzzy classes identification [5] or gradual rule extraction [11].

This paper focuses on the erosion score operator, that has been applied to efficiently detect periodicity in time series, interpreted as functions associating x_t at each time t [9]. The erosion score is based on the fundamental MM erosion operator and is more precisely defined as the sum of successive erosions until total erosion.

This paper considers the implementation issue for this operation and proposes three computation methods both to decrease the computational cost and to allow to process data incrementally: the proposed levelwise approach is based on the identification of levels in the data, to reduce the number of steps required to compute the erosion score for all data points. On the other hand, the proposed incremental extensions of both the basic and the levelwise computation methods make it possible to progressively update erosion scores when new data points become available, so as to process data streams.

The paper is organised as follows: Section 2 recalls the general MM principles and the erosion score as well as existing works related to optimised methods to compute MM operations. The following sections then respectively introduce the Levelwise, Incremental and Levelwise Incremental approaches. Lastly, Section 6 presents the experiments carried out to compare the proposed approaches.

2 Context and Motivations

Erosion Score Operation Mathematical Morphology relies on two basic operators, erosion and dilation, combined in various ways to define more complex composed operators (see [13] for a detailed presentation). In functional MM, given a function $f : E \rightarrow F$ and a *structuring element* B defined as a subset of E of a known shape, e.g. an interval centred at the origin, *erosion* is the function $\epsilon_B(f) : E \rightarrow F$ defined as $[\epsilon_B(f)](x) = \inf_{b \in B} f(x + b)$. Dilation is defined in a similar way, using the sup operator. These two basic operations can be used repeatedly or alternatively, leading to different types of composed operators, such as opening, closing or alternated filters [13].

The operator proposed in [9] to detect periodicity relies on the computation and aggregation of successive erosions. Given a time series X containing n values $\{x_1, \dots x_n\}$ in $[0, 1]$ obtained at regular time intervals and the structuring element $B = (-1, 0, 1)$, the previous erosion definition induces the following recursive form for the j^{th} erosion of the i^{th} value: $x_i^j = \min(x_{i-1}^{j-1}, x_i^{j-1}, x_{i+1}^{j-1})$, denoting by convention $x_i^0 = x_i$, $x_0 = x_{n+1} = +\infty$. The iterativity property of this erosion yields $x_i^j = \min(x_{i-j}, \dots, x_{i+j})$.

The erosion score is defined for a minimum zero X satisfying the property $\exists i \in \{1...n\}$ such that $x_i = 0$ as the sum of the successive erosions until total erosion: denoting z_i the number of erosion steps needed for x_i to be totally eroded, i.e. the smallest erosion step j such that $x_i^j = 0$

$$es_i = \sum_{j=0}^{z_i} x_i^j \tag{1}$$

Other methods based on successive erosions mostly aim at being used in 2D contexts, as the erosion curve [6] or the ultimate erosion [3].

Implementation Optimisation Efficient implementation methods have been proposed for many MM operations, to make it possible to compute them faster [16] or to allow them to process data incrementally [2].

In particular, various optimisations to quickly compute successive erosions have been proposed (see [7] for a recent state of the art): for instance they reduce the number of redundant operations when using various structuring elements [12] or propose a two pass optimisation to ensure a constant time computation of an erosion for any structuring element. However, such methods are not relevant for the erosion score computation where a single and simple structuring element is considered.

Another category of methods rely on the identification of specific values in the dataset, called anchors [15] or obtained after filtering out useless values to reduce the computation time [2]. The methods proposed in the paper belong to this category but they use another definition for the values of interest, as the latter are used to compute a different score.

3 Levelwise Method

The "naive" implementation of the erosion score consists in computing the successive erosions of the dataset and summing the obtained values until all eroded values equal 0. Since x_i is eroded in z_i iterations by definition of z_i and the whole dataset is processed at each iteration, its complexity is $O((\max_i z_i) \times n)$.

In this section, a levelwise approach is proposed: it does not process the whole dataset at each iteration but reduces the number of iterations for each data point individually. It is based on the identification of key erosion steps, i.e. a subset of the initial dataset sufficient to compute the erosion scores.

3.1 Notations

When computing the successive eroded values x_i^j, it can be observed that some of them are equal to the previous one. Identifying only the key ones, defined as the x_i^j different from the previous one, allows to compute the erosion score by adding the distinct values multiplied by their number of repetitions.

Formally, the key erosion steps are such that $x_i^j \neq x_i^{j-1}$, i.e. $x_i^j < x_i^{j-1}$ due to their definition. Let us denote $J_i^< = \{j \in \{1, \dots, n\} | x_i^j < x_i^{j-1}\}$, ω_i its size and D_i the ordered set of its values, sorted in ascending order, to which 0 is added as d_{i0}: $D_i = \{d_{i0}, \dots, d_{i\omega_i}\}$ is an ordered subset of $\{0, \dots, n\}$ where only the key steps are kept, in that the erosion score can be computed knowing them only. It can be noted that the maximal value of D_i is $d_{i,\omega_i} = z_i$. It holds that $\forall l \in \{0, \dots, \omega_i - 1\}$, $d_{il} < d_{il+1}$ and $x_i = x_i^{d_{i0}} = x_i^{d_{i0}+1} = \dots = x_i^{d_{i1}-1} > x_i^{d_{i1}} = x_i^{d_{i1}+1} = \dots = x_i^{d_{i2}-1} > x_i^{d_{i2}}$ and so on until $x_i^{d_{i,\omega_i}-1} > x_i^{d_{i,\omega_i}} = 0$.

We also introduce the notations $\chi_{il} = x_i^{d_{il}}$ and λ_{il} its index such that $\chi_{il} = x_{\lambda_{il}}$. d_{il} can be seen as the number of points between x_i and its l^{th} key value, χ_{il} is its value and λ_{il} its index. An illustrative example is given in Fig. 1.

3.2 Levelwise Computation of the Erosion Score

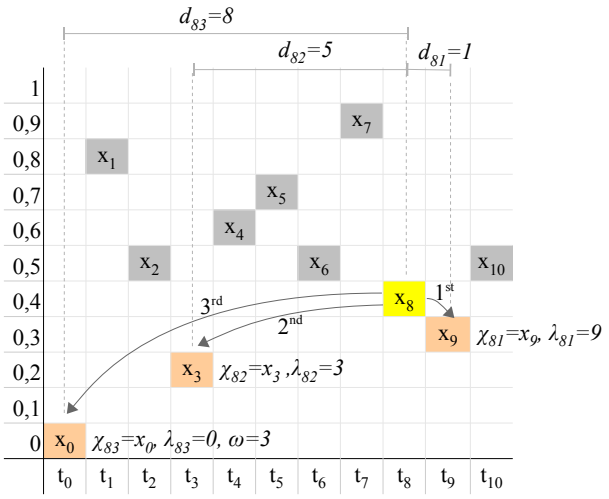
For any data point x_i, D_i then contains the key erosions, so the erosion score can be computed knowing these values only, as stated in the following theorem:

Theorem 1. *Levelwise computation of* es_i

$$es_i = \sum_{l=0}^{\omega_i - 1} (d_{i,l+1} - d_{il})\, \chi_{il} = \sum_{l=0}^{\omega_i - 1} (|\lambda_{i,l+1} - i| - |\lambda_{il} - i|) x_{\lambda_{il}}$$

Proof. The demonstration directly follows from the definitions of χ, λ and d developing the definition given in Eq. (1)

$$\begin{aligned} es_i &= \underbrace{x_i^0 + \dots + x_i^{d_{i1}-1}}_{(d_{i1}-d_{i0})\chi_{i0}} + \underbrace{x_i^{d_{i1}} + \dots + x_i^{d_{i2}-1}}_{(d_{i2}-d_{i1})\chi_{i1}} + \dots + \underbrace{x_i^{d_{i,\omega_i-1}} + \dots + x_i^{d_{i,\omega_i}-1}}_{\left(d_{i\omega_i}-d_{i,\omega_i-1}\right)\chi_{i,\omega_i-1}} + \underbrace{x_i^{d_{i,\omega_i}}}_{0} \\ &= \sum_{l=0}^{\omega_i - 1} (d_{i,l+1} - d_{il})\, \chi_{il} \end{aligned}$$



$x_8^0 = x_8$
$x_8^1 = x_9 < x_8$
$x_8^2 = x_8^3 = x_8^4 = x_9$
$x_8^5 = x_3 < x_8^1$
$x_8^6 = x_8^7 = x_3$
$x_8^8 = x_0 < x_8^5$

$i = 8,\ \omega_8 = 3$

$d_{80} = 0,\ \chi_{80} = x_8,\ \lambda_{80} = 8$
$d_{81} = 1,\ \chi_{81} = x_9,\ \lambda_{81} = 9$
$d_{82} = 5,\ \chi_{82} = x_3,\ \lambda_{82} = 3$
$d_{83} = 8,\ \chi_{83} = x_0,\ \lambda_{82} = 0$

Figure 1. Example and computation of $D_8 = \{d_{80}, d_{81}, d_{82}, d_{83}\}$, χ_{8l} and λ_{8l}

The expression based on λ_{il} is obtained from the fact that $d_{il} = |\lambda_{il} - i|$.

Based on this theorem, the levelwise implementation is computed from the λ_{il} values only. They are computed by searching x_i key erosions until a zero is reached, i.e. for j from 1 to z_i, so in z_i iterations. Since the erosion score is computed at the same time, the complexity of this method is $O\left(\sum z_i\right)$ which is lower than the naive one presented previously.

4 Incremental Method

Incremental approaches aim at processing the data successively, updating intermediate results instead of considering each data point independently. They can decrease the computational time; moreover they make it possible to process data streams, where the whole dataset is not available at once but data points are obtained progressively.

In this section and the following, we propose incremental methods to compute the erosion score, respectively based on the basic and the levelwise approaches.

4.1 Notations

In the context of data streams, x_{n+1} denotes the latest data point received at time $t+1$. We denote $x_i(t) = x_i$ if $i \in \{1, \ldots, n\}$ and $x_i(t) = +\infty$ otherwise. For simplicity sake, the notation x_i is preferred to $x_i\,(t)$ when no ambiguity arises. The value $x_0 = 0$ is added in order to ensure the minimum zero property and thus that the algorithm terminates. At time t, the j^{th} erosion of x_i is denoted $x_i^j\,(t)$ and its erosion score $es_i\,(t)$.

The objective of incremental methods is to compute the new erosion scores $es_i\,(t+1)$ from the existing erosion scores $es_i\,(t)$ and the new value x_{n+1}.

4.2 Update Equations for Eroded Values and Erosion Score

The theorem below establishes the update equation that gives the new eroded values for any data point when a new data point x_{n+1} is collected.

Theorem 2. *Update equations for the successive eroded values*
Denoting $q = \max\{k \in \{1,\dots,n\} \,|\, x_k \leq x_{n+1}\}$ *and* $m = (n+1+q)/2$,

$$x_i^j(t+1) = \begin{cases} x_i^j(t) & \text{if } i \leq m \\ x_i^j(t) & \text{if } i > m \text{ and } j < n+1-i \\ x_{n+1} & \text{if } i > m \text{ and } n+1-i \leq j < i-q \\ x_q^{j-(i-q)} & \text{if } i > m \text{ and } j \geq i-q \end{cases}$$

Proof. q is the index of the latest data point less than or equal to the new point x_{n+1} and m the middle of the index interval between q and $n+1$.

The proof consists in studying, for any i and j whether x_{n+1} and/or x_q are involved in the computation of $x_i^j(t+1)$. Since $x_i^j = \min(x_{i-j},\dots,x_{i+j})$, this is equivalent to checking whether $n+1$ and/or q belongs to $\{i-j,\dots,i+j\}$.

If x_{n+1} is not involved, then $x_i^j(t+1) = x_i^j(t)$. This is the case if the data point is closer to x_q than to x_{n+1}, so when $i \leq m$, since $x_q \leq x_{n+1}$ by definition. If $i > m$ and $j < n+1-i$, x_{n+1} is not involved too, since $n+1 \notin \{i-j,...i+j\}$.

If $i > m$ and $n+1-i \leq j < i-q$, then x_{n+1} is involved but not x_q, so $x_i^j(t+1) = x_{n+1}$. Indeed, for all $l \in \{q+1,\dots,n+1\}$, $x_l \geq x_{n+1}$ and the minimal data value on all index intervals included in $\{q+1,\dots,n+1\}$ is x_{n+1}.

Finally, if $i > m$ and $j \geq i-q$, then both x_{n+1} and x_q are involved, so $x_i^j(t+1) \leq x_q \leq x_{n+1}$, by definition of x_q. Therefore:

$$\begin{aligned} x_i^j(t+1) &= \min(x_{i-j},\dots,x_q,\dots,x_{n+1},\dots,x_{i+j}) = \min(x_{i-j},\dots,x_q) \\ &= \min(x_{q-(j-i+q)},\dots,x_q,\dots,x_{q+(j-i+q)}) = x_q^{j-i+q} \end{aligned}$$

These update equations lead to the update equations for the erosion score:

Theorem 3. *Computation of* $es\,(t+1)$
Denoting $q = \max\{k \in \{1,\dots,n\} \,|\, x_k \leq x_{n+1}\}$ *and* $m = (n+1+q)/2$,

$$es_i\,(t+1) = \begin{cases} es_i\,(t) & \text{if } i \leq m \\ es_q\,(t) + 2\,(i-m)\,x_{n+1} + \sum_{j=0}^{n-i} x_i^j\,(t) & \text{otherwise} \end{cases}$$

Proof. The theorem is a direct consequence of Theorem 2: if $i \leq m$, the successive erosions are not modified so neither is their sum. If $i > m$, the following decomposition of the erosion score proves the theorem:

$$es_i\,(t+1) = \underbrace{\sum_{j=0}^{n-i} x_i^j\,(t+1)}_{=\sum_{j=0}^{n-i} x_i^j(t)} + \underbrace{\sum_{j=n+1-i}^{i-q-1} x_i^j\,(t+1)}_{=2(i-m)x_{n+1}} + \underbrace{\sum_{j=i-q}^{+\infty} x_i^j\,(t+1)}_{=es_q(t)}$$

It is easily proven that the complexity of the incremental method is $O\left(\phi^2\right)$, where $\phi = n - q$, i.e. the distance between the new point and the latest lower data point.

5 Incremental Levelwise Method

This section proposes an alternative incremental method, based on the levelwise expression of the erosion score stated in Theorem 1 and on incremental update equations for the λ indices.

Theorem 4. *Incremental computation of* λ_{il}

Denoting $q = \max\{k \in \{1,\dots,n\} \text{ st } x_k < x_{n+1}\}$, $m = (n+1+q)/2$ *and* k_i *defined for* $i > m$ *such that* $d_{i,k_i-1}(t) < n+1-i \leq d_{ik_i}(t)$ *and* $k_{n+1} = 0$

$$\forall i,\, \forall l,\, \lambda_{il}(t+1) = \begin{cases} \lambda_{il}(t) & \text{if } i \leq m \\ \lambda_{il}(t) & \text{if } i > m \text{ and } l < k_i \\ n+1 & \text{if } i > m \text{ and } l = k_i \\ \lambda_{q,l-k_i-1}(t) & \text{if } i > m \text{ and } l > k_i \end{cases}$$

$$\forall i,\, \omega_i(t+1) = \begin{cases} \omega_i(t) & \text{if } i \leq m \\ k_i + \omega_q(t) & \text{if } i > m \end{cases}$$

Proof. The incremental expression stated in Theorem 2 allows the update of x_i^j for $j = 0...z_i$. Since D_i is a subset of $0...z_i$ containing only the key erosions, this proof is based on the one presented for the incremental method. k_i is introduced to represent the index in D_i denoting the first erosion involving x_{n+1}.

As in the incremental case, if $i \leq m$, or if $i > m$ and x_{n+1} is not involved, i.e. $l < k_i$, then the successive erosions are unchanged, so $\lambda_{il}(t) = \lambda_{il}(t+1)$.

If $l = k_i$ then the eroded value is x_{n+1}, so its index λ_{i,k_i} is $n+1$.

Finally, as proved for the incremental method, the next erosions, following the one equal to x_{n+1}, are the erosions of x_q. In the levelwise context, it implies that the key erosions following the one equal to x_{n+1} are also the key erosions of x_q for $l = 0...\omega_q$, so for $i > m$ and $l > k_i$, $\lambda_{il}(t+1) = \lambda_{q,l-k_i-1}(t)$.

Regarding ω_i the number of elements in D_i, when the eroded values are unchanged, i.e. $i \leq m$, the number of key erosions is unchanged too, and $\omega_i(t+1) = \omega_i(t)$. If $i > m$, then the key erosions are those from 0 to the first implying x_{n+1} whose index is k_i, plus the ω_q key erosions of x_q, yielding $\omega_i(t+1) = k_i + \omega_q(t)$.

This theorem together with the levelwise expression of the erosion score stated in Theorem 1 lead to the following levelwise incremental expression of the erosion score:

Theorem 5. *Incremental levelwise computation of es_i*

Denoting m, q and k_i as defined in Theorem 4 and p_i defined for $i \geq m$ such that $\lambda_{ip_i}(t) = q$:

$$\forall i,\ es_i(t+1) = \begin{cases} es_i(t) & \text{if } i \leq m \\ \chi_{i,k_i-1}(t)(n+1-i-d_{ik_i}(t)) & \text{if } m < i < n+1 \\ \quad -\sum_{j=k_i}^{p_i-1} \chi_{ij}(t)(d_{i,j+1}(t) - d_{ij}(t)) & \\ \quad +2x_{n+1}(i-m) + es_i(t) & \\ 2x_{n+1}(n+1-m) + es_q(t) & \text{if } i = n+1 \end{cases}$$

The variables d and χ are used to improve readability, but $es_i(t+1)$ can be computed with λ only since $d_{il} = |\lambda_{il} - i|$ and $\chi_{il} = x_{\lambda_{il}}$.

The proof is omitted because of space constraints. It follows from the decomposition of the sum given in Theorem 1 into 4 terms, corresponding to l lower than $k_i - 2$, equal to $k_i - 1$, between k_i and p_i, and greater than p_i. In each term, the $\lambda_{il}(t+1)$ values are replaced by their expression given by Theorem 4.

The implementation then consists in using a Λ matrix storing all λ_{il} values. When a new data point x_{n+1} is processed, its predecessor x_q is first identified. Then for each row whose index is greater than m, Λ is updated by computing k_i, inserting $n+1$ as the k_i^{th} value in the list, and copying the values from $(\lambda_{ql})_l$ at index k_i+1.

6 Experiments

6.1 Experimental Protocol

The 4 methods, namely "naive", "incremental", "levelwise", and "incremental levelwise", are compared over artificial datasets generated as noisy repetitions of identical blocks of different shapes (triangle, rectangle, sine, wave). Noise applies either on the size of the individual blocks, randomly enlarging or shrinking them, or on the data points, randomly adding or subtracting small values (see [8] for a more detailed presentation of the data generation process).

Each method is then applied to compute the erosion score of each data point in each dataset. For a given dataset, all methods return the same result. The data points are read one by one for the incremental approaches so as to emulate a stream.

For each method, the computational time as well as the memory consumption are measured; their average and standard deviation are computed over all the datasets randomly generated according to the protocol sketched above. Furthermore, the average value of ω_i is recorded for the levelwise methods.

The implementation is done in VB.NET and the experiments are run on a Windows® virtual machine started with 4 CPUs and 4 Go of memory on a physical machine with an Intel i7® CPU and 16 Go of memory.

6.2 Computational Time Comparison

Figure 2 shows the computational time for three levels of dataset sizes: the top graph compares all 4 methods for datasets containing less than 10,000 data points. It shows that the incremental methods outperform the non incremental ones. The naive implementation is significantly slower and less robust as the high standard deviation shows. Furthermore, the incremental methods run much faster in a real situation since the arrival of a new data implies only one computation whereas the non incremental ones have to run the computation anew over the whole dataset.

In order to differentiate more precisely the incremental methods, larger datasets are used, ranging from 10,000 to 100,000 data points as showed on the middle graph of Fig. 2. In this second round of experiments, the incremental levelwise method appears significantly faster than the incremental one. Moreover, the large values of standard deviation for the latter indicate a lack of robustness. This is due to the sensitivity of the position of zero values within the dataset for the methods not based on the levelwise approach. Indeed, as underlined in the complexity analysis of the 2 methods (see Section 2), if the largest z_i in a dataset is increased by only 1, another full scan of the dataset is needed with the naive method. With the levelwise approach on the contrary, one more iteration is needed only for the concerned data point

Finally, the largest datasets (bottom graph on Fig. 2) show that the incremental levelwise approach for erosion score computation is robust and can handle efficiently and in a very reasonable amount of time a large dataset on the order of one million data points. Moreover, since it is incremental, it can handle a million new data from one stream in 1.5 seconds, or equivalently handle 1 new data over a million streams in the same time.

6.3 Memory Consumption

In terms of memory, the non levelwise methods, whether incremental or not, are not based on specific data structures and thus do not need more memory than the space needed by the dataset and the resulting erosion scores: denoting n the number of data points, the memory consumption in this case is $2n$.

In the levelwise methods, the Λ matrix is stored, entailing an additional memory usage: it is implemented as a list of n lists each of them containing ω_i values. Its memory requirements is then $\sum \omega_i$ or equivalently $n \times avg(\omega_i)$. Over all carried out experiments, the average ω_i is 30, the minimum 2 and maximum 129, thus the required storing capacities remain reasonable.

Hence, the levelwise methods are more greedy in terms of memory than the non levelwise ones. Nonetheless, this can be mitigated simply for the incremental levelwise since when a zero value is reached, all previous values become useless in the erosion score, so the λ_i before the 0 value can be removed from Λ.

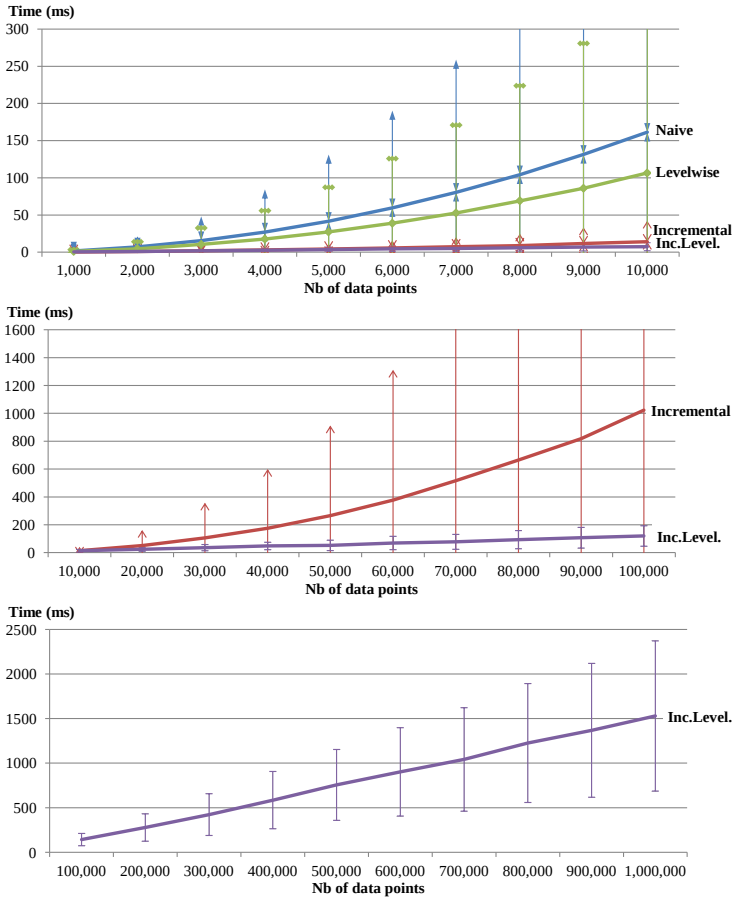


Figure 2. Computational time for (top) small datasets and all 4 methods, (middle) medium datasets and incremental methods, (bottom) large datasets and the incremental levelwise method.

7 Conclusion and Future Works

This paper proposed 3 variants to compute the erosion score based on one hand on a levelwise computation principle and on the other hand on update equations to progressively adapt to new incoming data points. Such incremental approaches make it possible to process data streams where data points are not available simultaneously. Experimental studies show the relevance of these variants and in particular the performance of the levelwise incremental approach, in terms of time consumption at the expense of a reasonable increase of memory storage.

Future works aim at integrating the efficient levelwise incremental method to the periodicity detection task, to identify periodicity in large time series. Other

perspective include the use of this approach to other time series pre-processing tasks where the series structure must be identified.

References

1. Bangham, A., Marshall, S.: Image and signal processing with mathematical morphology. Electronics & communication engineering journal 10(3), 117–128 (1998)
2. Dokládal, P., Dokládalová, E.: Computationally efficient, one-pass algorithm for morphological filters. Journal of Visual Communication and Image Representation 22(5), 411–420 (2011)
3. Lantuejoul, C., Maisonneuve, F.: Geodesic methods in quantitative image analysis. Pattern Recognition 17(2), 177–187 (1984)
4. Lefèvre, S., Claveau, V.: Topic segmentation: application of mathematical morphology to textual data. In: Proc. of ISMM 2011. pp. 472–481 (2011)
5. Marsala, C., Bouchon-Meunier, B.: Choice of a method for the construction of fuzzy decision trees. In: Proc. of FUZZ-IEEE 2003. vol. 1, pp. 584–589 (2003)
6. Mattioli, J., Schmitt, M.: Inverse problems for granulometries by erosion. Journal of Mathematical Imaging and Vision 2(2-3), 217–232 (1992)
7. Morard, V., Dokládal, P., Decencière, E.: One-Dimensional Openings, Granulometries and Component Trees in Per Pixel. IEEE Journal of Selected Topics in Signal Processing 6(7), 840–848 (2012)
8. Moyse, G., Lesot, M.-J., Bouchon-Meunier, B.: Mathematical morphology tools to evaluate periodic linguistic summaries. In: Proc. of FQAS 2013. pp. 257–268 (2013)
9. Moyse, G., Lesot, M.-J., Bouchon-Meunier, B.: Linguistic summaries for periodicity detection based on mathematical morphology. In: Proc. of IEEE SSCI FOCI 2013. pp. 106–113 (2013)
10. Najman, L., Talbot, H.: Mathematical Morphology. John Wiley & Sons (2013)
11. Oudni, A., Lesot, M.-J., Rifqi, M.: Characterisation of gradual itemsets based on mathematical morphology tools. In: Proc. of EUSFLAT 2013. pp. 826–833 (2013)
12. Pecht, J.: Speeding-up successive Minkowski operations with bit-plane computers. Pattern Recognition Letters 3(2), 113–117 (1985)
13. Serra, J.: Introduction to mathematical morphology. Computer Vision, Graphics, and Image Processing 35(3), 283–305 (1986)
14. Sun, Y., Chan, K.L., Krishnan, S.M.: Characteristic wave detection in ECG signal using morphological transform. BMC cardiovascular disorders 5(1), 28 (2005)
15. Van Droogenbroeck, M., Buckley, M.: Morphological erosions and openings: fast algorithms based on anchors. Journal of Mathematical Imaging and Vision 22(2-3), 121–142 (2005)
16. Vincent, L.: Fast grayscale granulometry algorithms. In: ISMM. vol. 2, pp. 265–272 (1994)
17. Wang, X., Tang, H., Zhao, X.: Noisy Speech Pitch Detection Based on Mathematical Morphology and Weighted MACF. In: SINOBIOMETRICS. vol. 3338, pp. 594–601 (2005)