FULL-LENGTH REPRINT – differs from the original in layout but not in contents
Chmielewski L.J., Orłowski A.,
Detecting changes with the robust competitive detector.
In Alexandre L., Salvador Sánchez J., Rodrigues J., Eds., Pattern Recognition and Image Analysis. IbPRIA 2017,
volume 10255 of Lecture Notes in Computer Science, pages 353-362,
Faro, Portugal, 20-23 Jun 2017. doi:10.1007/978-3-319-58838-4_39
The copyright owner of the publication is Springer.
The final publication is available at https://doi.org/10.1007/978-3-319-58838-4_39.
Contains brief erratum – see Sect. 4.

Detecting Changes with the Robust Competitive Detector

Leszek J Chmielewski and Arkadiusz Orłowski

Faculty of Applied Informatics and Mathematics – WZIM,
Warsaw University of Life Sciences – SGGW
ul. Nowoursynowska 159, 02-775 Warsaw, Poland
{leszek_chmielewski,arkadiusz_orlowski}@sggw.pl
http://www.wzim.sggw.pl

Abstract. The concept of the competitive filter is reminded and its ability to find changes in 1D data is extended by adding the robustness feature. The use of two affine approximators, one at the left and one at the right side of the considered data point, makes it possible to detect the points in which the function and its derivative changes, by subtracting the outputs from the approximators and analyzing their errors. The features of the detector are demonstrated on artificial as well as real-life data, with promising results.

Keywords: competitive detector, function change, derivative change, robust

1 Introduction

The changes in the observable environment are for us the main source of information. Our senses quickly get used to stable signals and tend to ignore them after some time, while the change in light intensity, a new sound or smell attracts our attention. A good indication that the changing signals are important is that even when we observe static objects our eyes are constantly making saccadic movements, which support the perception of interesting details of the scene.

The detector presented in this paper originated within the domain of image processing, where the problem of change detection is primarily related to edge detection [1], motion detection [11] and the detection of changes [10].

In 1991 Niedźwiecki, Sethares and Suchomski proposed a filter denoted as the *competitive filter* [8,9]. As a method to overcome the phenomenon of blurring the edges during image filtering, present in the methods based on averaging, they proposed to use two competing filters working simultaneously from the two sides of an edge in an image. As the output, the result from that filter was used in which the approximation error was smaller.

In 1996 one of the authors of this paper proposed to use the competitive filtering to detect edges [2]. The concept was successfully developed for one-dimensional data, but the extension to two dimensions failed [3] and was abandoned. The reason for the failure was the variety and complexity of shapes of

L.A. Alexandre et al. (Eds.): IbPRIA 2017, LNCS 10255, pp. 353-362, 2017.
DOI: 10.1007/978-3-319-58838-4_39

the image intensity function near the junctions of various kinds of edges. These shapes made it difficult to define the notion of the *two sides* of an edge which was basic in competitive filtering.

It can seem unexpected but to our best knowledge there were no other publications on the idea of the competitive filter or edge detector, similar to ours, than the ones just cited. In 2003 Liang and Looney [7] proposed to use the competition in edge detection. Pixels competed towards becoming edge pixels to reduce the edge thickness. This approach was different from the ours.

At present we have decided to revive the concept of competitive edge detection, solely in the application to one-dimensional data. The detector behaved well for such data; nevertheless, to improve its stability, it has been extended by including a robust analysis mechanism to the fitting of the two filters. This paper can be treated as the proof of concept of the robust version of the detector. The concept of the detector comes from the domain of image processing, so some image processing terminology will appear in this paper, although the relation of the detector to images is only historical. In particular, we shall use the terms *change*, *edge* and *jump* interchangeably.

The method proposed is well suited to be augmented by incorporating statistical methods of verifying the significance of the detected changes (see eg. [4,5]). The work in this direction is in progress. Moreover, the presented method does not need any model of the phenomenon described by the analyzed function. No assumptions are made on the form of the function. The function given as a series of points is locally approximated, with polynomials of a chosen degree, without the assumption of continuity in the point analyzed, which is a minimalistic approach as far as assumptions are considered. Therefore, we do believe that this method could provide a valuable alternative tool for detecting changes in various kinds of time series and other one-dimensional data.

This paper is organized as follows. In the next Section 2.1 the detector in its introductory form is described and its operation is illustrated with the result for a synthetic test image. The new feature of robustness is explained in Sect. 2.2. Then, in Sect. 3 the results of the detection of changes in some real-life data are presented. The question of efficiency is answered in Sect. 4. The discussion is in Sect. 5 and the proposed directions of development close the paper in Sect. 6.

2 The method

2.1 General concept

According to [9] let us take a sequence of measurements $z(x) = y(x) + n(x)$, where $n(x)$ is noise. The independent variable x is discrete. If x is understood as time, the measurements are known starting from the past, up to the point x_0 and further, to $x_0 + D$. The point x_0 is the point in which the filtering is performed and can be called the *central point.* Two approximators, referred to as the *$\underline{L}$eft* and the *$\underline{R}$ight* one, are used to find $y(x_0)$, the first one working at the left side of x_0, using $z(x), x \in [x_0 - s - \Delta, x_0 - \Delta]$ to find $\hat{y}_L(x_0)$, and the second one working at the right side of x_0, using $z(x), x \in [x_0 + \varDelta, x_0 + s + \varDelta]$ to find $\hat{y}_R(x_0)$. The parameter s is the scale of the filter, and $\varDelta$ is the gap between the central point x_0 and the estimators.

For each approximator its error is estimated: $e_L(x_0)$ and $e_R(x_0)$, respectively. As the estimate of the result at the point of interest, $\hat{y}(x_0)$, the output of the filter which performs better, that is, gives a smaller error is used. This is the competitiveness feature of the filter. In the literature, the extrapolations at x_0 from the two filters were used. In the present implementation, the output at $x_0 - \varDelta$ from the left approximator, and for $x_0 + \varDelta$ for the right one, is used, to avoid using extrapolated values. This stabilizes the results, especially when robust approximators are used, which will be described further.

In [2], the linear least square approximators were used as filters, and their mean square errors were used as their approximation errors. The concept of using the difference of outputs from the two filters $\hat{y}_L(x_0)$, $\hat{y}_R(x_0)$, and their derivatives, as the estimates of the step and roof edge at point x_0 was introduced. The conditions for the existence of the step was that the graphs of the approximation errors crossed in such a way that for increasing x the error from the past increased and for decreasing x the error for the future increased. In the present paper it is assumed that at least one of the errors increases in this case. These conditions were expressed in [2] in a complicated way but they can be simply written down, respectively, as

$$\begin{aligned} &e_R(x_0 - \delta) > e_L(x_0 - \delta) \ \wedge \ e_R(x_0 + \delta) < e_L(x_0 + \delta)\,, \\ &e_R(x_0 - \delta) > e_R(x_0 + \delta) \ \vee \ e_L(x_0 - \delta) < e_L(x_0 + \delta)\,. \end{aligned} \tag{1}$$

Because the past error should be known for $x_0 + \delta$ then the measurements for $x_0 + D = x_0 + \varDelta + s + 2\delta$ should be known. The parameter δ can be called the neighborhood parameter. In [2] it was $\varDelta = \delta = 1$ and so it will be assumed in this paper. Therefore, $D = s + 3$.

The process of error graphs crossing is illustrated in Fig. 1[1]. Let us describe it in a figurative rather than in a rigorous way. Let us imagine that both approximators together with the central point are moved along the data from left to right. When a step is encountered, first the right approximator moves over it so the step enters the right approximator's support. Therefore, the error of the right approximator goes up. As the analyzed point is moved forward, the step leaves the support of the right approximator, so its error goes down, and enters that of the left one (this particular moment is shown in the figure). Now, the error of the left approximator increases.

At the crossing point, the edge intensity values E_0 for the jump of the function, and E_1 for the jump of its first derivative, respectively, are calculated as

$$\begin{aligned} E_0(x_0) &= \hat{y}_R(x_0 + \varDelta) - \hat{y}_L(x_0 - \varDelta)\,, \\ E_1(x_0) &= \hat{y}'_R(x_0 + \varDelta) - \hat{y}'_L(x_0 - \varDelta)\,, \end{aligned} \tag{2}$$

[1] The software and the graphs were developed in Matlab®.

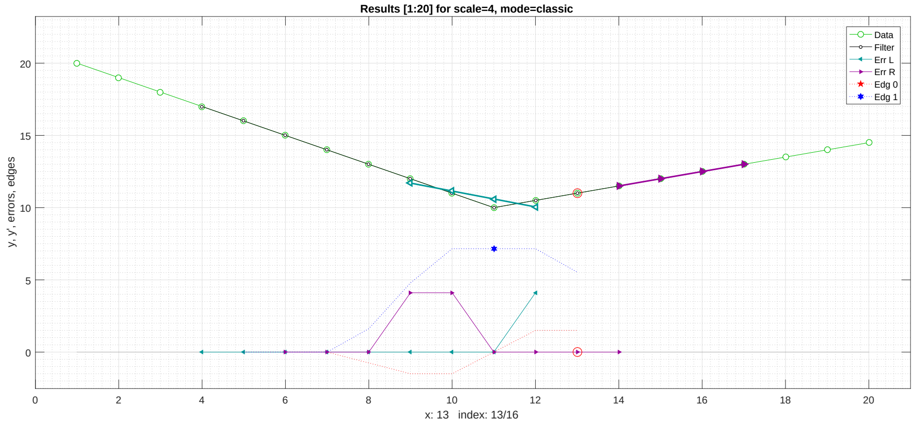

Fig. 1. Intermediate results for the two approximators up to $x_0 = 13$: Graphs of errors (thin magenta and cyan lines) cross between points 10 and 12. The meaning of types and colors of lines and marks partly explained in the legend: `Err L`: left error, `Err R`: right error; `Edg 0`: jump of the function, `Edg 1`: jump of its first derivative – this type of edge is marked at $x = 11$ with a blue star. The current central point x_0 marked with a red circle on the axis as well as on the graph of the function. The left and right approximators around the central point shown with thicker cyan and magenta lines. The approximator with zero error has full marks, the other one has marks filled with white. Angles shown in tens of degrees. Errors multiplied by 10 for better visibility.

where $\hat{y}'_{\cdot}(\cdot)$ can be found from the approximators due to that they are linear. In the present application, not the filtered value but the values of the steps will be of the primary interest.

In Fig. 2 the result is shown for data in which all the detectable changes are present: a roof edge, a step edge and a combined step and roof edge. The data are synthetic and clean. What is apparent is that the roof edge is detected in a single point, like this at $x = 1$, while the step edge, like this at $x = 20, 21$ is found at two points. This is correct due to that the jump of a discrete function appears between two points and can be assigned to both of them. It should also be stressed that there are no separate conditions for the two edge types detected. If one of the edge types is missing, then its intensity is zero.

2.2 Robustness

In [2] the use of the least median of squares [12] was mentioned as the possible approximation criterion, but in that paper it was not developed. Here we propose to use this criterion. We have used the algorithm from [6] with the correction pointed out in the discussion on this algorithm in its web site (given in [6]), entry of 06 Oct 2004. The issue of high time requirements of the algorithm does not constitute an important problem here because the scale of the approximators is not expected to be higher than 10 to 20.

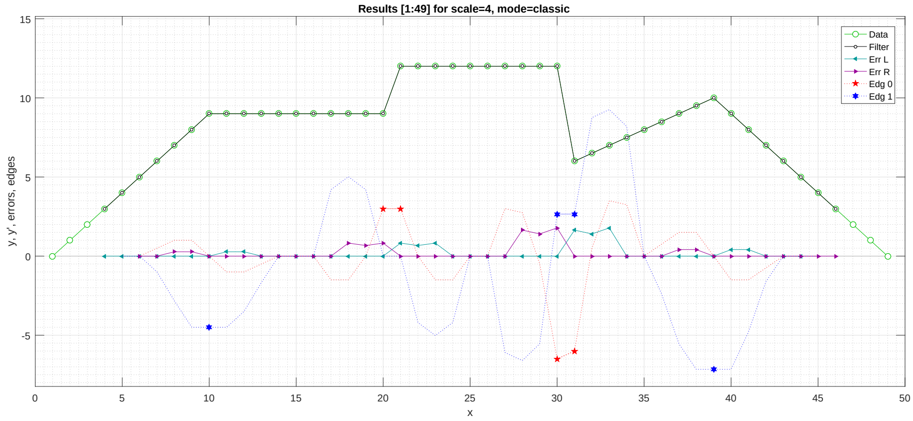

Fig. 2. Results for the classic version of the detector for synthetic data with all the detectable changes represented: function jump at $x = 20, 21$, derivative jump at $x = 10$ and 39 and combined jumps at $x = 30, 31$. Angles shown in tens of degrees.

In Fig. 3 the results for data containing all the detectable changes can be seen. Some outlying vales, or point noise-type artefacts, are added to illustrate the functioning of the robust detector. With the robust version of the detector, the influence of the outliers which are far from the edges, relatively to the scale, was eliminated. However, the outlier at $x = 26$ which is close to the step at 30 interfered with the detection of this step and prevented from finding one of its points (at $x = 30$). Also, the value of the function was slightly changed at $x = 29$. The outlier inside the roof edge at $x = 39$ gave rise to a spurious step at $x = 38$. The interferences would be greater if the scale were larger.

It is clear that the robust detector can not extinguish the influence of the outliers located in the vicinity of the edges to be detected. It should also be pointed out that it can be doubtful whether the extraneous value is an outlier or constitutes a valid part of the signal. The data point can be considered an outlier if it does not follow the trend, which is not an univocal statement. We shall come back to this problem in the next Section.

3 Real-life example

As an example let us consider the graph of free memory of some web server in 100 intervals of one minute each, starting from 04 July 2016, 10:43:01 GMT. The memory was averaged in each minute. The graph in Fig. 4 displays an uneven load so the used memory requirement jumps are significant.

The general question which should be asked is which data should be treated as valid and which should be dismissed as outliers or noise. The extraneous values of such values as memory requirement are all measured precisely, but it is probable that some of them do not follow the current trend. So, some of them

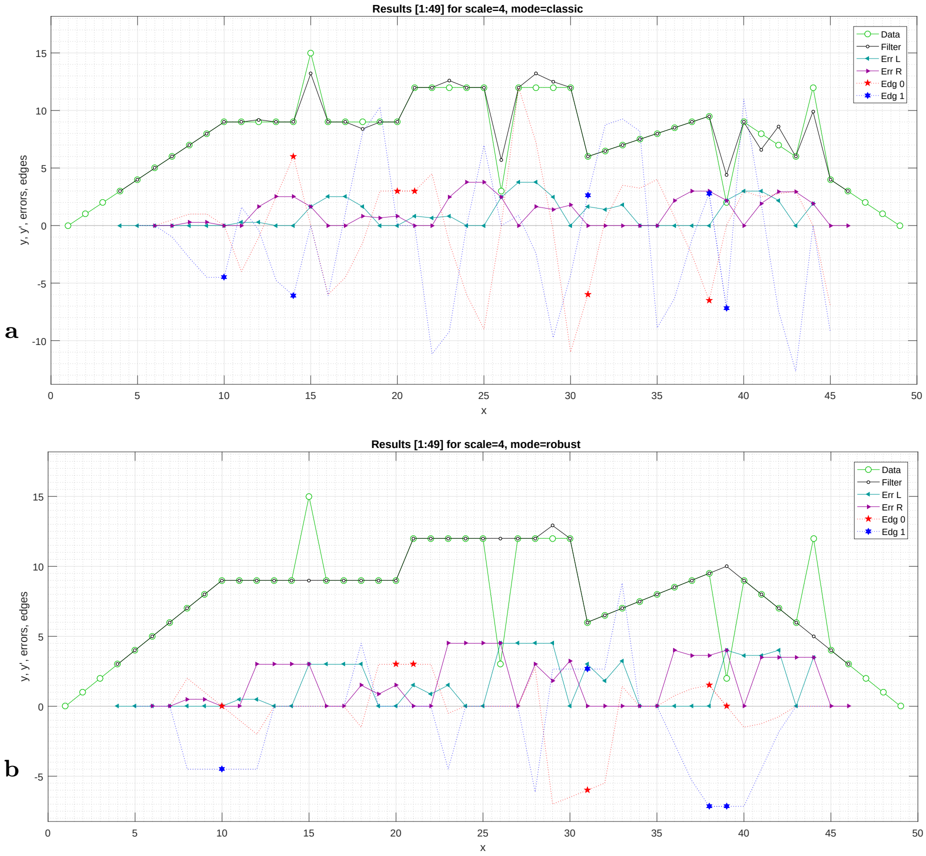

Fig. 3. Results for the data of Fig. 2 with some outlying data added. Versions of the detector: (**a**) classic; (**b**) robust. In (b) the influence of some outliers was eliminated but some interfere in the detection – see text. Angles shown in tens of degrees.

may be considered as untypical, but such values are significant. The question on the significance of single data items most probably can not be answered without looking at other measures of the state of the observed object.

The observation of approximators reveals their disadvantageous behavior, which could be expected solely on the basis of the design of the approximators. It consists in that the intensity of the tendency to minimize the approximation errors does not depend on which end of the support of the approximator is considered: the one closer to or farther from the central point x_0. However, it is more important to minimize the error at points closer to x_0. The adverse case is shown and explained in Fig. 6. Therefore, the weighting of the minimized errors would be a preferable feature of the algorithm. This is more important in the robust approximators than in the classic ones, because in the robust

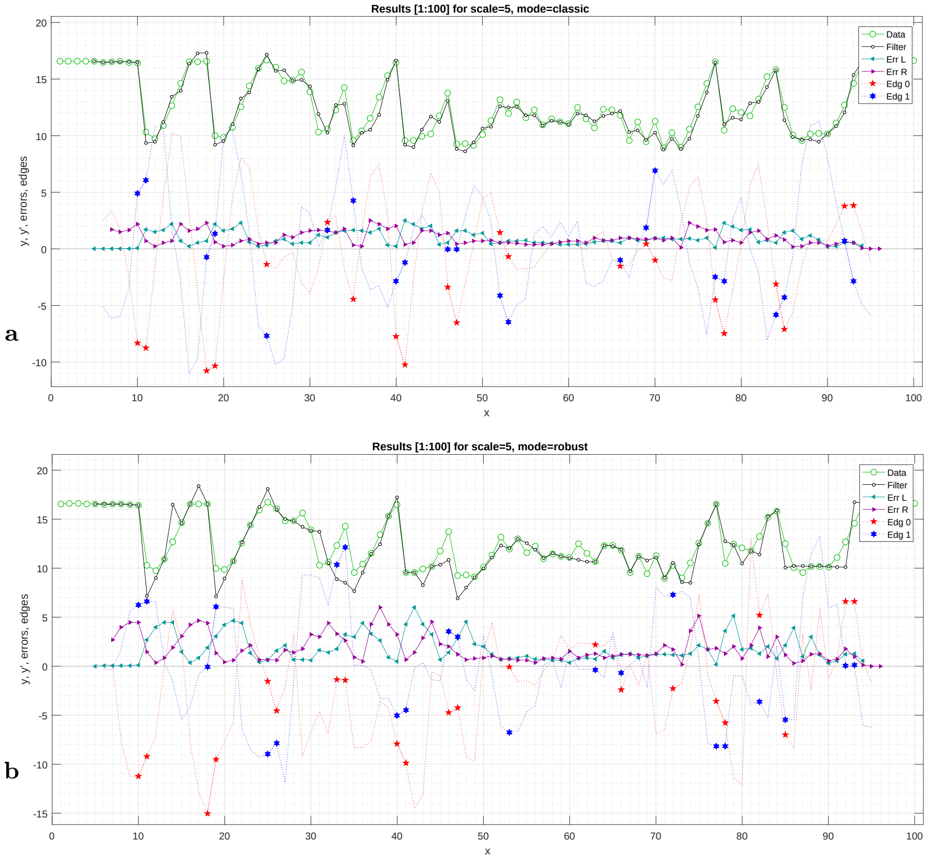

Fig. 4. Analysis of free memory of a web server averaged for 1 min intervals in 100 minutes (starting at 04 July 2016, 10:43:01 GMT) in GB, at scale $s = 5$ (angles shown in tens of degrees). Detectors: (**a**) classic; (**b**) robust. Some parts of the signal are deformed, for example, the vertices at the jumps are protruded in the classic as well as robust versions. The scale seems to be too large for the signal variability.

approximators some data points are postponed. It is vital that the points close to the central point were postponed more rarely than the other ones.

4 Efficiency

It can be safely stated that the time complexity of the algorithm is linear with respect to the number n of data points to be processed: $O(n)$, because the same analyzes are repeated for every part of the data (data for two approximators), and the size of this part does not depend on the data size. The complexity depends on the scale s in the same way as the approximation algorithm does. It would not be practical to set the scale to large values so the algorithm can be

In the Proceedings this section was erroneously numbered as a subsection of Sect. 3. Further sections are renumbered here, accordingly.

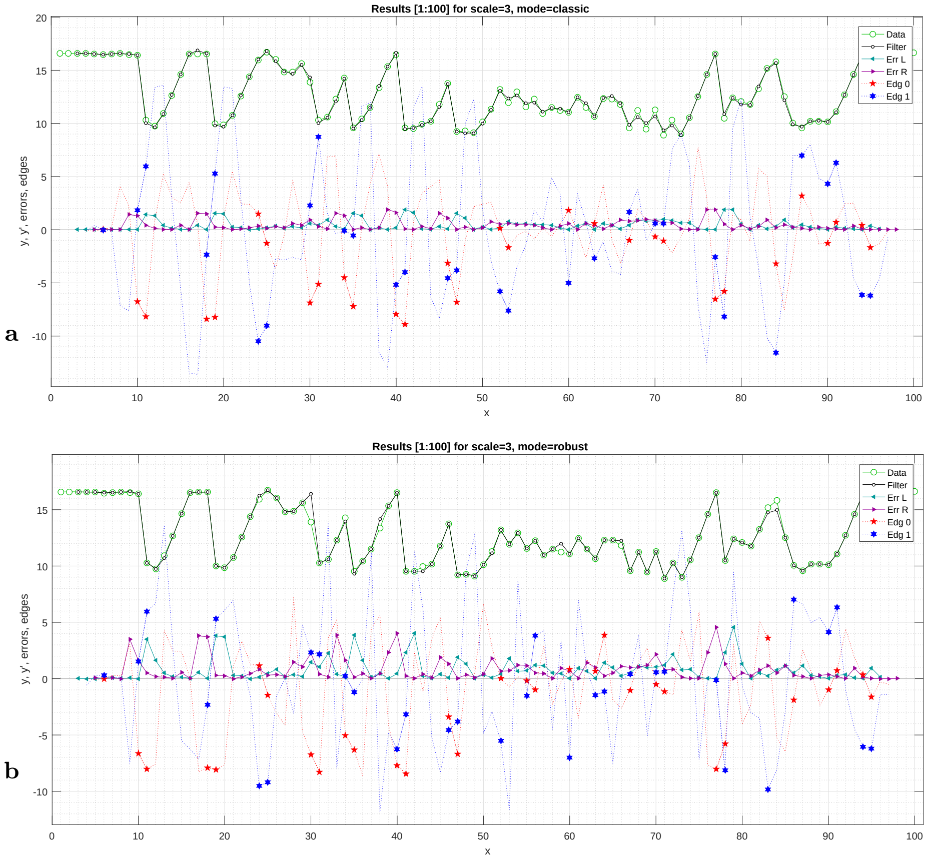

Fig. 5. Analysis of data of Fig. 4 at scale $s = 3$. The important effects – step and above all roof jumps positively detected in the majority of cases. The robust detector seems to better stick to small jumps at $x = 55 - 57$ but treats some peak points as noise at $x = 84, 85$. Detectors: (**a**) classic; (**b**) robust.

considered efficient. In the case of the least squares algorithm this is also linear: $O(s)$. With the least median of squares used at present it is $O(s^3 \log(s))$, but s is small and constant. Also, faster least median of square algorithms are available now. At the level of the proof of concept, and for linear complexity with respect to the data size, the time requirements are not the crucial factor.

In the case of least squares methods, the approximators could be implemented incrementally, so the dependence on the scale would be reduced. Moreover, in the basic implementation, one approximator performs the same operations as the other one, at the different locations of the central point, so the repeated calculations could be avoided.

5 Discussion

Section renumbered – see Sect. 4.

The proposed algorithm has a number of positive and negative features. In each of these groups of features some improvements are possible. The advantages, drawbacks and possible improvements will be summarized separately.

Advantages

- Detection is performed together with the filtering process.
- Higher order approximators can be developed to fit the data more accurately.
- Higher order derivatives can be estimated according to the order of the approximating functions.
- The complexity of the algorithm with respect to the size of the data is linear.

Drawbacks

- A considerable part of data on the future must be known before the detection can be made.
- Besides the scale s of the detector, which is a meaningful and useful parameter, there are other parameters to be tuned: gap $\varDelta$ and neighborhood δ.

Possible improvements

- The algorithm should minimize the errors at those ends of the supports of the approximators, which are closer to the central point, to a larger extent than at the opposite ones. Weighting of the errors would be a desirable solution.
- The gap $\varDelta$ and neighborhood δ are the parameters which could be optimized. At present they are both assumed to be equal to 1.
- The design of the error function can be improved, especially in the case of robust approximators. One of the possible measures of error could be the prediction accuracy of the approximators, for example estimated by the difference $|y(\cdot) - \hat{y}.(\cdot)|$.
- The existence of edges can be verified with statistical methods.
- Efficiency can be improved by incremental calculations and data reuse.

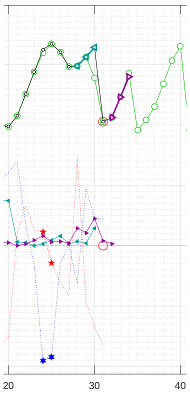

Fig. 6. Detail of Fig. 5 at $x = 31$. The left robust approximator omits the point $x = 30$ located at its end close to the central point $x_0 = 31$. This will cause the filtered value to be located far from the point $x = 30, y = 14$ and the edge value to be overestimated. This could be avoided if the approximator minimizes the error at its end close to the central point rather than at the other end.

6 Summary and prospects

Section renumbered – see Sect. 4.

The concept of the competitive filter was recapitulated and developed further by adding the robustness feature to the existing algorithm. This feature requires some care but can be profitable in the presence of impulsive noise. The filter was used as a detector of changes in the value and first derivative of the signal analyzed. The changes were detected with the set of conditions for the measures of errors made by the approximators, while the intensity of the jumps were found as differences of the values at the output of the approximators. Some positive results were obtained for data representing a strongly changing signal. The detector has a potential of being further developed and improved. It seems reasonable to work on higher degrees of approximating functions, on weighted error minimization and on statistical verification of changes.

References

1. Bhardwaj, S., Mittal, A.: A survey on various edge detector techniques. Procedia Technology 4, 220–226 (2012), doi:10.1016/j.protcy.2012.05.033
2. Chmielewski, L.: The concept of a competitive step and roof edge detector. Machine Graphics & Vision 5(1-2), 147–156 (1996),
3. Chmielewski, L.: Failure of the 2D version of the step and roof edge detector derived from a competitive filter. Report of the Division of Optical and Computer Methods in Mechanics, IFTR PAS (Dec 1997),
4. Furmańczyk, K., Jaworski, S.: Large parametric change-point detection by a V-box control chart. Sequential Analysis 35(2), 254–264 (2016), doi:10.1080/07474946.2016.1165548
5. Jaworski, S., Furmańczyk, K.: On the choice of parameters of change-point detection with application to stock exchange data. Quantitative Methods in Economics XII(1), 87–96 (2011)
6. Leontitsis, A.: LMS toolbox (from MathWorks® Comunity File Exchange) (Sept 2001), www.mathworks.com/matlabcentral/fileexchange/801-lms-toolbox. [Online; updated 05-January-2004, accessed 15 Nov 2016]
7. Liang, L.R., Looney, C.G.: Competitive fuzzy edge detection. Applied Soft Computing 3(2), 123–137 (2003), doi:10.1016/S1568-4946(03)00008-5
8. Niedźwiecki, M., Sethares, W.A.: New filtering algorithms based on the concept of competitive smoothing. In: Proc. 23rd ISCIE Symp. on Stochastic Systems and its Applications SSS'91. pp. 129–132. Osaka, Japan (1991)
9. Niedźwiecki, M., Suchomski, P.: On a new class of edge-preserving filters for noise rejection from images. Machine Graphics & Vision 1-2(3), 385–392 (1994)
10. Radke, R.J., Andra, S., Al-Kofahi, O., Roysam, B.: Image change detection algorithms: a systematic survey. IEEE Transactions on Image Processing 14(3), 294–307 (March 2005), doi:10.1109/TIP.2004.838698
11. Räty, T.D.: Survey on contemporary remote surveillance systems for public safety. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 40(5), 493–515 (Sept 2010), doi:10.1109/TSMCC.2010.2042446
12. Rousseeuw, P., Leroy, A.: Robust Regression and Outlier Detection. John Wiley (1987)