International Journal of Statistics and Probability; Vol. 6, No. 5; September 2017
ISSN 1927-7032 E-ISSN 1927-7040
Published by Canadian Center of Science and Education

Applying Kolmogorov-Zurbenko Adaptive R-Software

Igor G Zurbenko[1] & Mingzeng Sun[1]

[1] Department of Epidemiology and Biostatistics, State University of New York at Albany, New York, USA

Correspondence: Mingzeng Sun, Department of Epidemiology and Biostatistics, State University of New York at Albany, New York, USA. E-mail: msun@albany.edu

Received: July 17, 2017 Accepted: August 13, 2017 Online Published: August 21, 2017

doi:10.5539/ijsp.v6n5p110 URL: https://doi.org/10.5539/ijsp.v6n5p110

Abstract

The Kolmogorov-Zurbenko Adaptive, kza package provides algorithms to deal with abrupt changes or breaks in the presence of heavy background noise. In a practical way, one-dimensional and high-dimensional simulated samples are generated to demonstrate signal recoveries and their accuracy evaluation by mean of squared error, mean difference and specificity index. Simulation investigation showed that smoothing window size need consider whenever applying kza package, and that kza could tolerate background noise about 10-folds heavier in higher-dimensional data compared to 1-dimensional data.

Keywords: Kolmogorov–Zurbenko filter, KZA software, Non-parametric, High-dimension data

1. What is Kolmogorov-Zurbenko Adaptive Algorithm?

Kolmogorov-Zurbenko Adaptive (KZA) algorithm (Close, B. & Zurbenko, I. G., 2016), is a family member of Kolmogorov-Zurbenko (KZ) R-software (Kolmogorov–Zurbenko filter). KZ software is a moving average technique, belongs to the class of low-pass filters (Zurbenko, I. G., 1986). It is designed to identify temporal features in time series data, such as trends, short-term fluctuations, seasonal and long-term variations; however, KZ filter tends to smooth out abrupt discontinuities (including spikes and ditches), in turn makes it difficult to estimate their properties (Rao, S. T. & Zurbenko, I. G., 1994). Extended from KZ filter, KZA enables to reveal any abrupt discontinuities in time series without distorting other existed patterns (Zurbenko, I. G. et al., 1996). Compared to KZ filter, the beauty of KZA filter is applying a dynamic smoothing window driven by data, while operate the moving average process (Zurbenko, I. G. et al, 1996; Yang, W. & Zurbenko, I. G., 2010). The KZA filter first identifies potential intervals when a break occurs; it then examines these intervals more accurately by reducing the smoothing window length to improve the resolution of the smoothed outcome.

2. When is KZA filter Used?

According to Zurbenko and his colleagues (Kolmogorov–Zurbenko filter; Yang, W. & Zurbenko, I. G., 2010), KZA filter was developed to detect breaks in nonparametric signals embedded in heavy background noise. It detects sudden changes over a low frequency signal of any nature submerged in heavy noise. For data with 1-dimensional setting, it can be used to identify intervention application, policy change and data quality issues as well (Rao, S. T., & Zurbenko, I. G., 1994; Solaun O., et al., 2013; Gonz áez, M. et al., 2013). As for data with 2-dimensional settings, such as 2D CT image data, 2D meteorological data, KZA can be applied to detect the breaks (change of color and/or change of density) (Kolmogorov–Zurbenko filter). Given data with 3-dimensional setting, like 3D spatial data, or 3D medical image, KZA can be applied directly to detect breaks (boundaries between layers) (Zurbenko, I. G., & Sun, M. 2016). In case of data with higher-dimensional setting, KZA can also be used to detect signal discontinuities with some modifications. KZA shows very high sensitivity for break detection, even with a very low signal-to-noise ratio.

3. How is KZA Algorithm Manipulated

The KZA algorithm acquires all the typical advantages of a nonparametric approach, and it does not require any specific model of the samples under investigation.

3.1 Install R packages

To run KZA algorithm, need to install ‘kza’ R package and the dependent package ‘polynom’.

3.2 Syntax

The basic syntax for executing KZA algorithm is:

y1<-kz(x1, m=c1, k=c2)

v1 <- kza(x1,m=c1,y=y1,k=c2, impute_tails=TRUE)

x1 – input data series, it can be in the form of 1D vector, 2D matrix or 3D array

m – smoothing window size, it can take value of c1 in the compatible form of input data x1

k – number of iterations, it can take any positive value of c2

y1 – the outcome from KZ algorithm

impute_tails - The default is to drop the tails.

v1 – the outcome of KZA

3.3 Output

To conduct KZA smoothing, it needs to initiate the program by providing input data "x1", window size 'c1' and iteration number 'c2'. After executing KZA algorithm, the smoothed data is 'cubed' in the same form as the input data. It can be extracted by command "v1$kza".

4. Example

In practice, we have 2 things to carefully consider before executing the algorithm: smoothing window size and iteration number, to avoid under-smoothing or over-smoothing. We suggest 2 or 3 iterations in general; Regarding the smoothing window size, it depends on the data; in general, the bigger the smoothing window size, the better / smoother the outcome (Figure 1), but we need balance the smoothness and error. To illustrate our points, using simulated data, we estimate the KZA filter's smoothing effect by window size and background noise in this article. While choosing window size, we also need pay enough attention to the meaningful logic for the real data, rather than pure statistics.

4.1 Window Size

Assumed that there is a vector, containing time series data with timepoint = 1 through 100, at all timepoints signal = 0 except for timepoint 42 through timepoint 59, 18 timepoints in total (top left in Figure 1), where signal = 1. This signal is embedded in background noise with sigma=0.5 (top middle, Figure 1). Applying KZA algorithm with different window size. Figure 1 displays a better smoothness with bigger window, as long as the window size is within the signal size (18 is the signal coverage size).

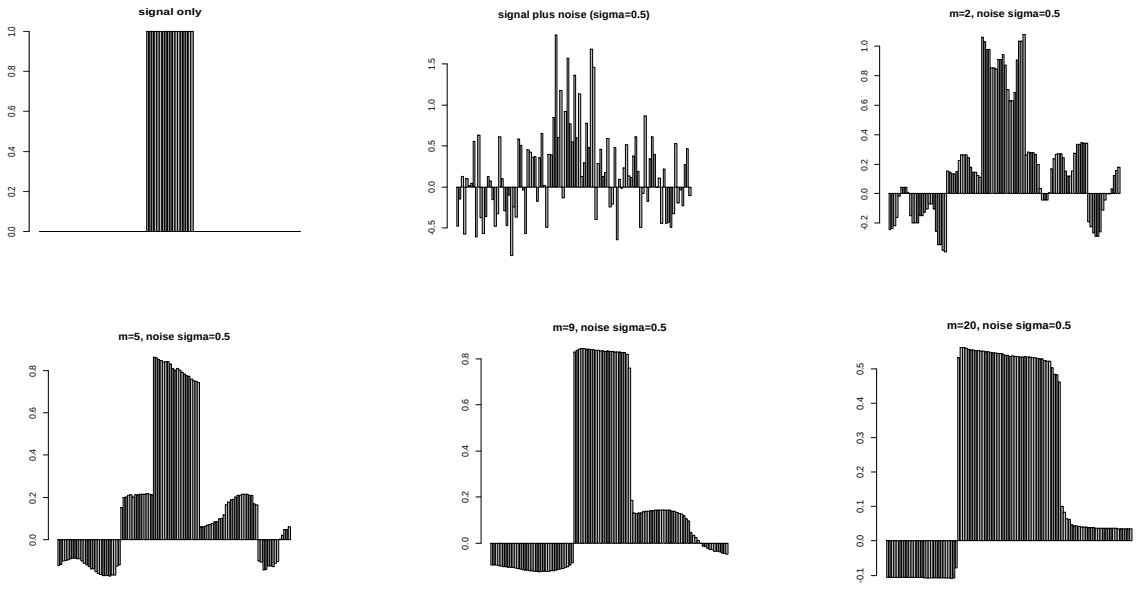


Figure 1. KZA smoothing effects by window size in 1-dimension space

4.2 Measures for Evaluating Smoothing Effects

To illustrate the effect of KZA algorithm, we introduce three statistics: mean squared error (MSE), mean difference (MD) and specificity index (SI) to evaluate how window size and background noise affect smoothness and fidelity.

$$\text{MSE} = \sum_{i,j,k} |Signal - v\$KZA|^2 \ / N$$

$$\text{MD} = \sum_{i,j,k} (Signal)/N - \sum_{i,j,k} (v\$KZA)/N$$

$$\text{SI} = \sum_{c(i,j,k)} |0 - v\$KZA|^2 \ / N_c$$

here, *i, j,* and *k* are the signal coordinate index in the sample space, *N* is the signal sample size; *c(i, j, k)* is the index for signal uncovered space, it is compensate to signal space; N_c is the difference between entire sample space and signal space, another word, it is the signal uncovered space size.

4.3 One - Dimensional Time Series Data

The simulated data are composed of non-signal area 1 from timepoint 1 through timepoint 41, signal area from timepoint 42 through timepoint 59, and non-signal area 2 from timepoint 60 through timepoint 100; this signal is embedded in background noise (σ=0.5). Therefore, N=18, N_c=41 + 41=82.

With different window size, we applied KZA algorithm to the simulated data. Smoothing effects by window size is summarized in figure 2. Overall, KZA could do a good job by using window size from 2 through 10, with the best window size falling at m=9, evidenced by a good combination of MSE=0.0297, SI=0.0627 and MD=0.171.

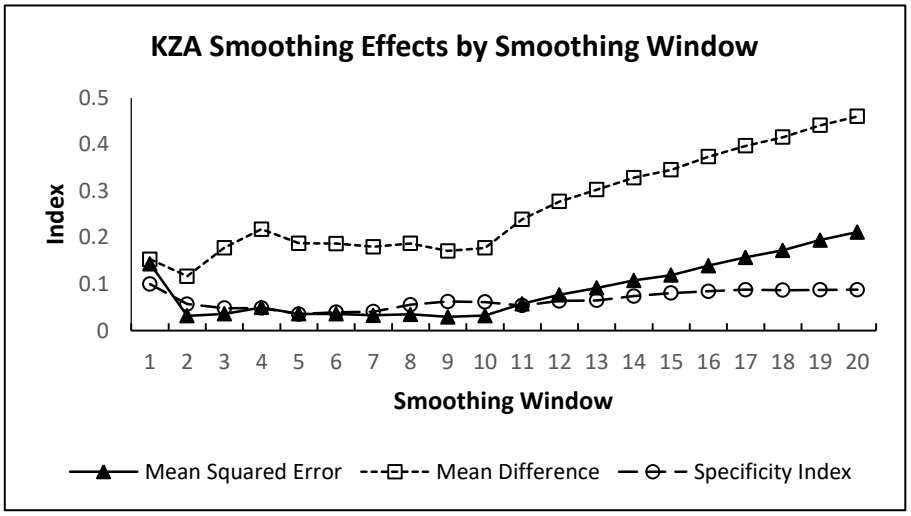


Figure 2. Influence of window size on KZA smoothing effects of 1D signal

We also looked at the influence of background noise on KZA smoothing effects. It is not surprise to see an increased smoothing error with the increase of background noise (Figure 3). Under 1-dimension setting, KZA could perform a decent job, while handling data with its background noise up to σ = 0.5. In terms of its power of handling background noise, KZA has a much better performance when smoothing higher-dimensional data, we will address this later.

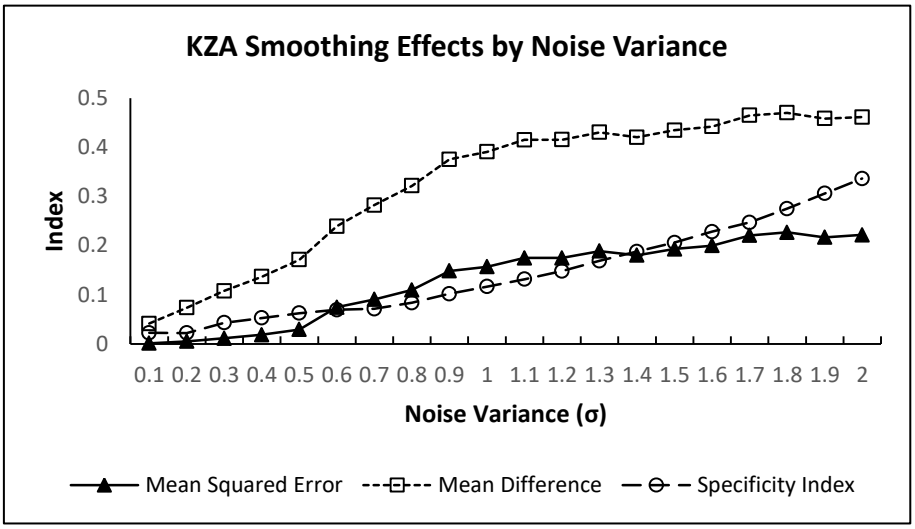


Figure 3. Influence of background noise on KZA algorithm

4.4 Two - Dimensional Data

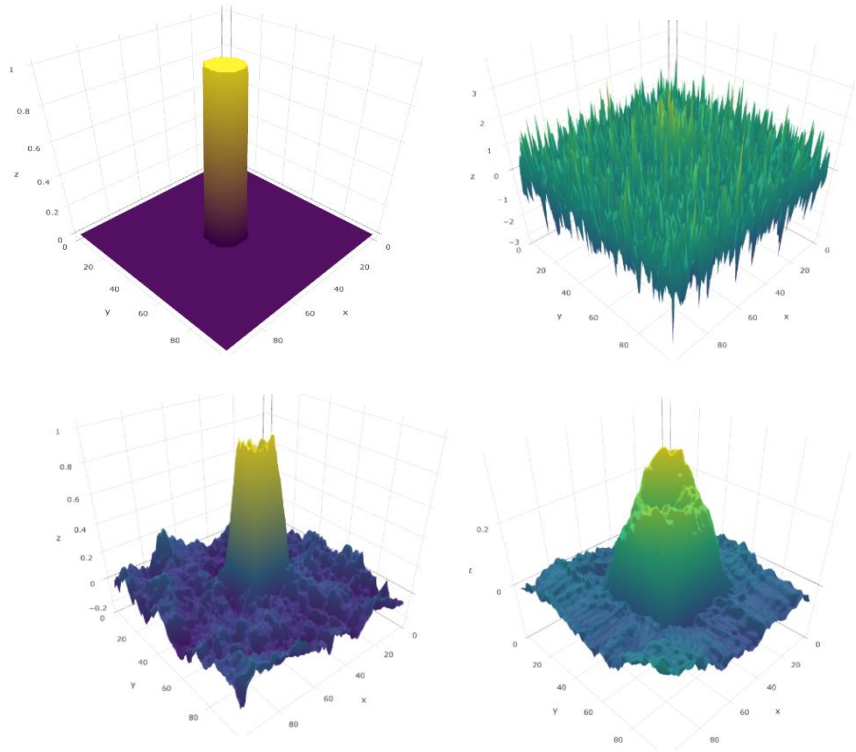


Figure 4. KZA algorithm unsmoothed and smoothed simulated signal data

Simulated 2-dimensional signal is a cylinder with diameter=18, located in a 100 x 100 space (top left, Figure 4), this signal is embedded in background noise with σ = 1 (top right, Figure 4). After applied KZA algorithm using window size 8 (left bottom, Figure 4) and window size 20 (right bottom, Figure 4), images of recovered signal indicate that window size plays an important role in performing KZA algorithm. Figure 5 showed KZA smoothing effects with different window size. It is possible for KZA to perform decently using window size from 3 through 10, with the best size m = 8 (MAE = 0.022, SI = 0.008, MD = 0.118), which is about the half signal size. Similarly, as KZA performed in 1-dimensional data, as long as the window size is not larger than half size of the signal dimension, KZA algorithm is capable of recovering the signal in a decent shape.

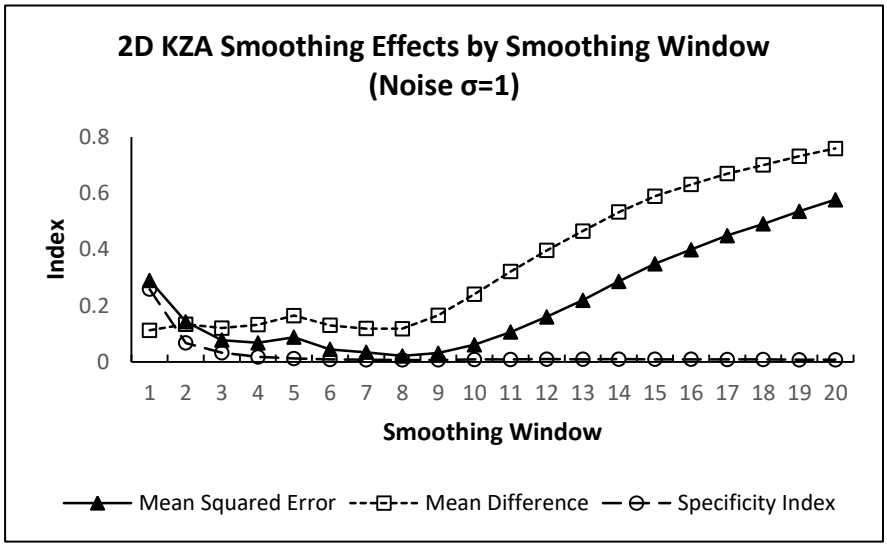


Figure 5. Influence of window size on KZA smoothing effects of 2D signal

To better understand the role of window size in KZA algorithm, we used rectangle-shaped signal, instead of cylinder-shaped signal. Since it is under 2D setting, so the window size applied is logically in the form of 2D: m = c(a, b). The simulated data is composed of signal = 1 covering a field with the dimension of 32 x 16 and noise ($\sigma = 1$) covering the entire field with the dimension of 100 x 100. We recovered the signal using 3 types of smoothing windows: (1), square window – j x j; (2), rectangle window – 2j x j; and (3), rotated rectangle window – j x 2j. Figure 6 showed that the rectangle smoothing window performed the best, and the best smoothing window is m = c(16, 8), which are the half size of signal dimensions.

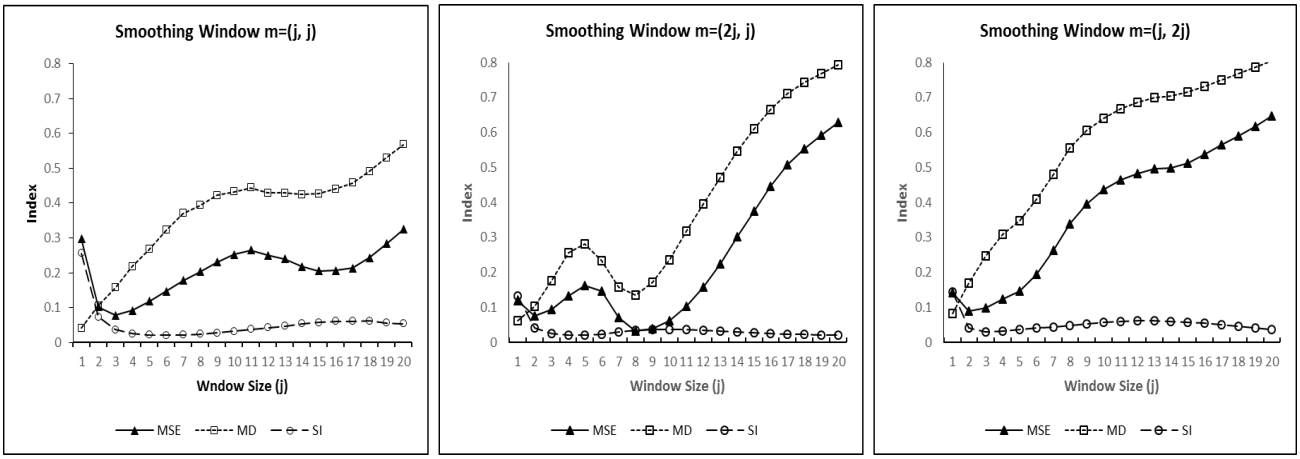


Figure 6. Comparison of window shape and window size for 2D signal

4.5 Three - Dimensional Data

Three-dimensional simulated data are composed of cylinder-shaped signal = 1 (diameter = 18) located in a 100 x 100 two-dimensional field; and in the time domain, there is 100 timepoints forming the 3rd dimension; this signal only existed from timepoint 33 through timepoint 68, and at all the rest timepoints signal = 0. The signal mentioned above is embedded in a three-dimensional (100 x 100 x 100) background noise ($\sigma = 1$). Using dimension compatible smoothing window, e.g. m = c(j, j, 2j), we applied the data to KZA algorithm, and showed the results in figure 7. When 2 <= j <= 9, the recovered signal is in a good shape (Figure 7, 8). To visualize the recovered signal, we respectively displayed the signals recovered by using smoothing window m = c(2, 2, 4), m = c(4, 4, 8) and m = c(8, 8, 16) at the timepoint 50 (Figure 8). Basically, the recovered signals were good, with slight difference of background and/or signal between each other. This indicates that KZA could perform relatively better, given much more information is available under three-dimensional settings than that under one -dimensional settings. Another word, KZA could tolerate heavier background noise while applying to higher-dimensional data.

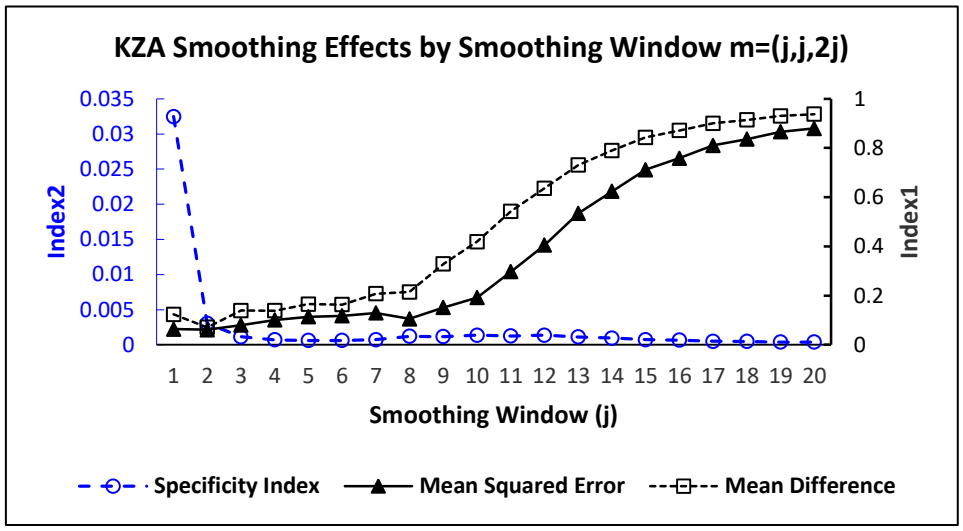
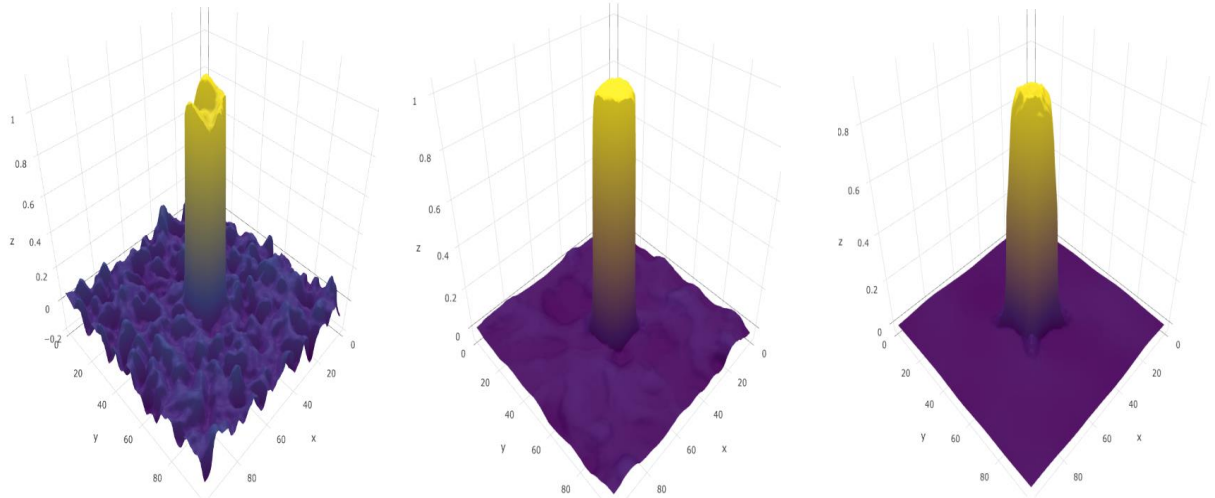


Figure. 7. Influence of window size on KZA smoothing effects of 3D signal



m=c(2,2,4) m=c(4,4,8) m=c(8,8,16)

Figure 8. Comparison of window size for smoothed 3D signal

To better understand that KZA algorithm can work more efficiently to recover 3D signals embedded in the heavy background noise, we applied KZA to 1D, 2D and 3D signals addressed in section 4.3, 4.4 and 4.5, and signals were respectively embedded in a background noise (σ is from 0.5 to 10) with compatible dimensions. We displayed the results in figure 9. In general, KZA algorithm can handle background noise 10 folds heavier under two - dimensional (in brown) and three - dimensional (in black) settings than that under one - dimensional (in blue) settings. Regarding the difference between two - dimensional and three – dimensional data, while the noise is not very heavy, for example with $\sigma < 3.5$, KZA performs better for two - dimensional data; otherwise, KZA algorithm would perform better for three - dimensional data.

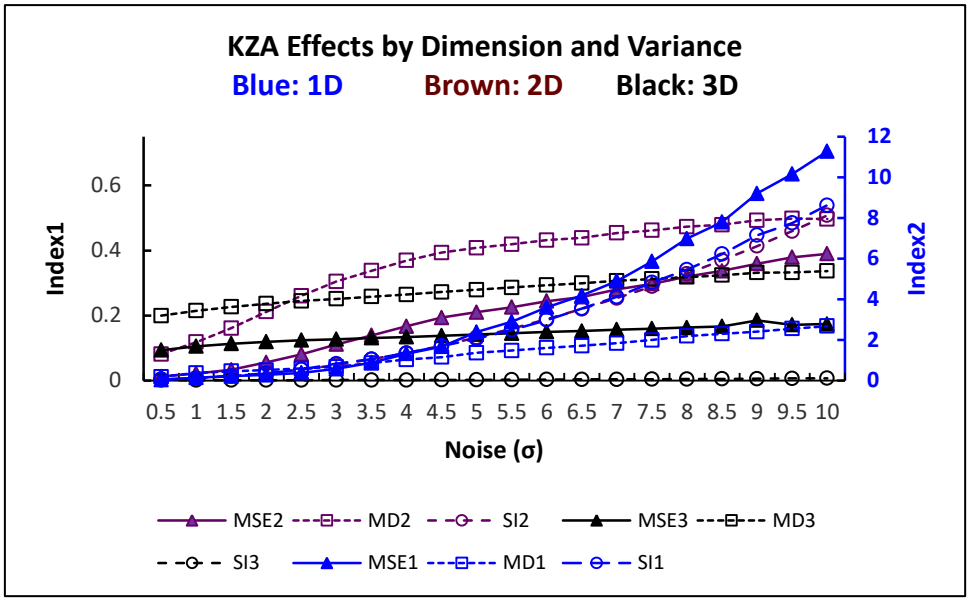
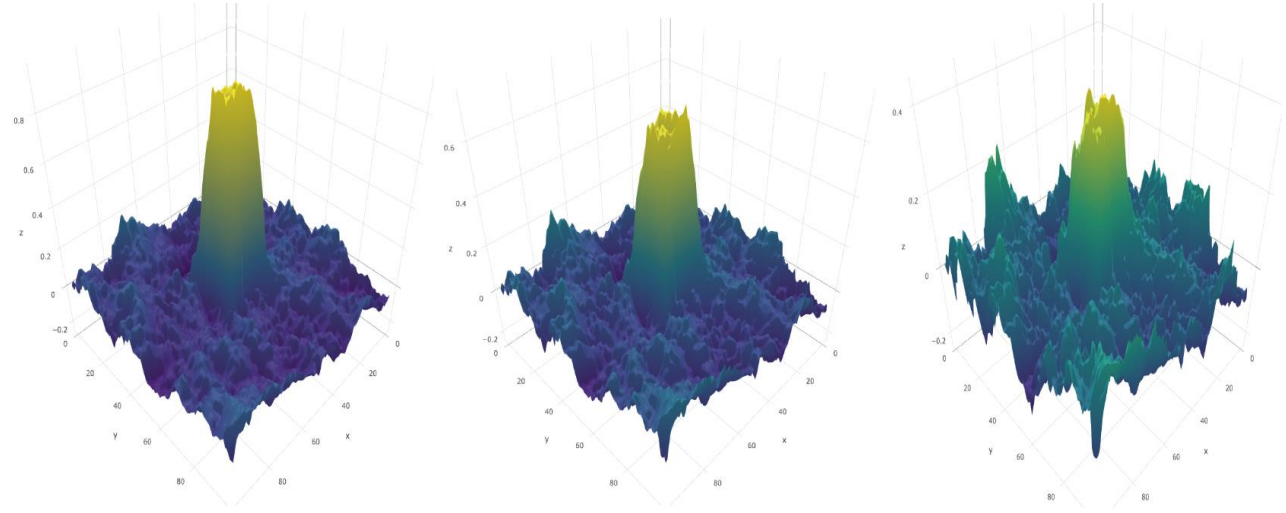


Figure 9. Influence of variance and dimension on KZA smoothing effects

5. Missing Data



m=9, σ=1, p=0 m=9, σ=1, p=25% m=9, σ=1, p=50%

Figure 10. Influence of missing values on KZA smoothing effects.

Another impressive feat of kza algorithm is its capability to handle missing data (Close, B. & Zurbenko, I. G., 2016; Zurbenko, I. G. et al., 1996), as visualized in figure 10. Two-dimensional simulated data (section 4.4) was made missing randomly with missing rate p, then applied to kza filter. Smoothing effects of kza algorithm on data with missing rate from 0 to 0.9, was displayed in figure 11. Overall, the specificity index was not affected by data missing rate; but mean squared error and mean difference were strongly affected as the missing rate went above 50%. However, this influence seemed only concentrated on reflecting changes of mean difference; for example, while using kza to recover data with 50% missing values (for actual data, 50% missing is very high), we still get a pretty fair result with mean squared error = 0.42 and mean difference = 0.64. When plotted the image (Figure 10), we could see a decent image (right, Figure 10); although its mean declined to 0.4 (original signal was 1), but the signal was very well identifiable.

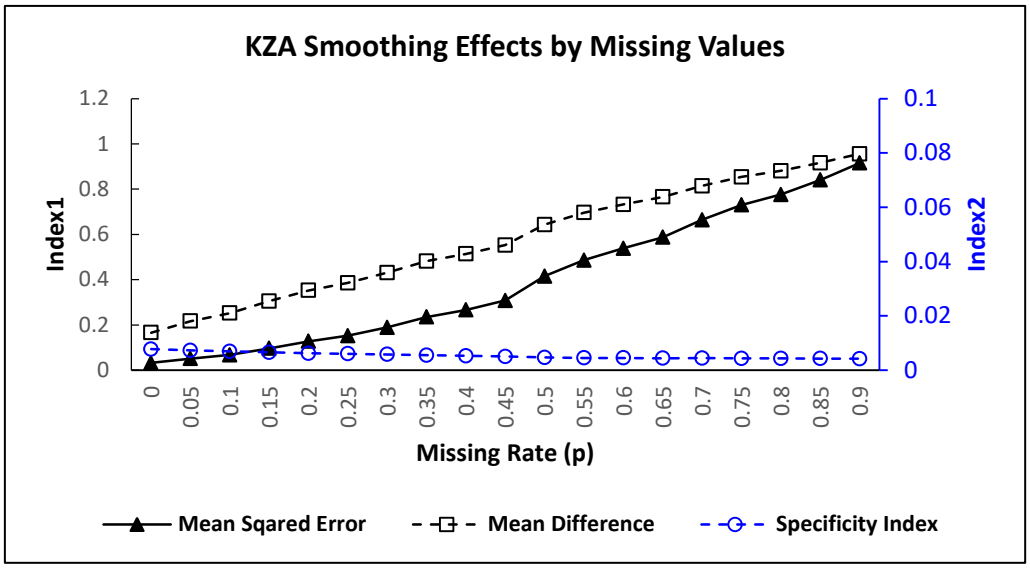


Figure 11. Influence of data missing rate on KZA smoothing effects

6. Discussion

KZ(m, k) filter is arranged by polynomial coefficients which can be interpreted as convolution of k uniform distributions length m. Sum of k independent uniform distributions is approximately normal distribution N(0, k m^2/12) with Gaussian coefficients. Time window of KZ(m, k) filter has finite support mk and effective width $m\sqrt{k}$ / $2\sqrt{3}$. It is a smooth function with continuous k-2 derivatives equal 0 outside mk /2. This property provides very small vulnerability to the noise (Kolmogorov–Zurbenko filter; Zurbenko, I. G., 1986; Yang, W., & Zurbenko, I. G. 2010). Truncated Gaussian window is discontinuous at the edges and vulnerable to the noise. KZ filter in other words repeat mathematical beauty of Gaussian infinite support, but transferring it to an inevitable finite support of computer applications. KZ filter provide suppression of the noise to the value $2\sqrt{3}\sigma$ / $m\sqrt{k}$ and has extraordinary good frequency resolution. It provides suppression of the noise by power function of the order 2k. MSE of KZ reconstruction is closest to the optimal compare with any other filtrations (Zurbenko, I. G., 1986). KZ adaptive filter is starting with KZ(m, k), then "zooming in" at the area of a possible break. Value of the suppressed noise obviously should be smaller than possible break. Larger values of m and/or k will provide better reconstruction of the break. Multidimensional KZ filter may provide better reconstruction because of multidimensional power of coefficient $2\sqrt{3}$ / $m\sqrt{k}$.

Without considering the actual meaning of data in this article, we addressed the questions, recommendations and concerns regarding performance of kza algorithm from a practical point of view. However, for real data in the actual world, we have to consider the logic meaning behind our data when we apply statistical techniques. Generally speaking, larger values of m, k are better for the reconstruction. Nevertheless, very big values of m, k may start to include specific frequencies/periods which hard to interpret as noise. For example, frequently we are searching for the breaks in the long term component (Zurbenko, I. G. et al., 1996; Close, B. & Zurbenko, I., 2008). Seasonality of the data may provide fast changing values which may wrongly be interpreted as break. To avoid such effect we may select m = 365 days (or 730 days). KZ filter with such m will remove completely annual component from initial consideration and will make KZA invulnerable from annual fluctuations. In the same way, we are removing strong variation cosine square law from spatial images to let opportunity to search for the regional breaks.

7. Conclusion

Understanding KZA algorithm, when it is used, what could affect its performance, and how to manipulate are very important in its application. This article has covered these very basics of KZA algorithm in a practical way, trying to convince readers: (1). KZA is simple, it does not require any specific model of your data. (2). It is practical and useful, it can handle data embedded in heavy noise. (3). It is powerful, it is non-parametric, does not require any assumptions of the data under study; it can apply to data in various of format, such as one – dimensional data vector, two – dimensional data matrix, three – dimensional data array, and even higher dimensional data (with modification). KZA algorithm even works better while applied to three - dimensional data.

References

Close, B., & Zurbenko, I. G. (2016). kza: Kolmogorov-Zurbenko Adaptive Filters.

Kolmogorov–Zurbenko filter.

Zurbenko, I. G. (1986) .The spectral analysis of time series. *Elsevier North Holland Inc. 248*,

Rao, S. T., & Zurbenko, I. G. (1994). Detecting and tracking changes in ozone air quality. *J. Air Waste Manag. Assoc., 44*, 1089–1095. https://doi.org/10.1080/10473289.1994.10467303

Zurbenko, I. G., Porter, P. S., Gui, R., Rao, S. T., Ku, J. Y., & Eskridge, R. E. (1996). Detecting Discontinuities in Time Series of Upper-Air Data: Development and Demonstration of an Adaptive Filter Technique. *Journal of Climate, 9*, 3548-3560.

Yang, W., & Zurbenko, I. G. (2010). Kolmogorov–Zurbenko filters. *WIREs Computational Statistics, 2*, 340–351. https://doi.org/10.1002/wics.71.

Solaun, O., Rodríguez, J. G, Borja, A., González, M., & Saiz-Salinas, J. I. (2013). Biomonitoring of metals under the water framework directive: detecting temporal trends and abrupt changes, in relation to the removal of pollution sources. *Mar Pollut Bull., 67*(1-2), 26-35. https://doi.org/10.1016/j.marpolbul.2012.12.005

González, M., Fontán, A., Esnaola, G., & Collins, M. (2013). Abrupt changes, multidecadal variability and long-term trends in sea surface temperature and sea level datasets within the southeastern Bay of Biscay. *Journal of Marine Systems*, 109-110, (Supplement), S144-S152. https://doi.org/10.1016/j.jmarsys.2011.11.014

Zurbenko, I. G., & Sun, M. (2016). Jet Stream as a Major Factor of Tornados in USA. *Atmospheric and Climate Sciences 6*(2), 236-253. https://doi.org/10.4236/acs.2016.62020.

Close, B., & Zurbenko, I. (2008). Time Series Analysis of the Fatal Accident Reporting System. *Proceedings of Annual ASA conference.*

Copyrights