Wild Binary Segmentation: A Case Study

Applying WBS to Detect Changepoints in Stationary Stock Return Data

Ishan Muzumdar IHM5018@PSU.EDU
Time Series Analysis
Pennsylvania State University
Department of Statistics

Abstract

Wild Binary Segmentation (WBS), developed and published in 2014 by Piotr Fryzlewicz, is a novel changepoint detection algorithm which builds upon previous changepoint detection methods, most notably standard binary segmentation. Fryzlewicz, intrigued by the low computational complexity and intuitive structure of standard binary segmentation, sought to build upon its foundation while fixing some of its faulty performance issues. Thus, he introduced a "wild" component to the algorithm, which provides random subset selection and CUSUM localization within each subset. In this paper, we seek to verify the claims made about WBS' performance in identifying changepoints, apply WBS to a variety of datasets, comment on the algorithm's shortcomings, and offer potentials solutions and extensions to the results presented in the original WBS paper. We find that the algorithm, though largely successful on reliable datasets, experiences some irregular results when applied to more complex, volatile time series.

Keywords: Wild Binary Segmentation, Changepoint Detection, Stationarity

1 Introduction

Changepoint detection is a broad, powerful tool used to identify changepoints, or data values when the underlying distribution of a stochastic, often time-based, process changes. Several statistical models rely on the assumption of stationary to yield accurate and generalizable results. However, in examining data, it is possible that external forces, or internal instability, can affect the distribution of the data. These distributional shifts can have significant ramifications in how forecasting is performed. More importantly, they can negatively influence the reliability of these forecasts. As a result, many statisticians have gained an interest in exploring changepoint detection methods and such analysis has emerged as a popular area of research. This field is particularly applicable in time series modeling – having a keen understanding of changepoints is crucial in building comprehensive, practical results when evaluating time series data.

Historically, changepoint detection has presented itself in several disciplines, including medicine, finance, anthropomorphic studies, computer science, and many more. Nonetheless, the rigor with which statisticians approached this problem did not arise until the 1950s with the revitalization of Bayesian modeling. Changepoint detection algorithms made noteworthy strides between the 80s and 2000s, when researchers developed least square estimators, information criterions, and segmentation/edge detection methods. Today, many changepoint detection methods intersect with common classifier problems in supervised learning. These procedures have been used to solve a variety of interesting problems in climate change, stock analysis, streaming algorithms, and speech recognition.

The benefits of changepoint detection within the realm of time series are two-fold. First, identifying changepoints allows statisticians to fit more accurate models. For a particular segment of data, knowing if and where changepoints occurs can signal that certain data should be removed from the model-estimation process, thereby eliminating extra "noise" in the model. Second, in rare instances, changepoint detection can potentially detect seasonal trends within the data. This can enable statisticians to predict future transformations of the data's distribution.

Though many changepoint procedures, especially those that leverage machine learning, can provide context for each changepoint within the entire time series, fundamental issues arise in the scalability of such techniques. When examining granular, expansive time series datasets, it is important to mitigate the accuracy of changepoint methods with their parsimony and efficiency. This trade-off is of particular intrigue to many computational statisticians as they aim to formulate algorithms which run quickly and correctly.

One of the most recent developments in changepoint detection is the wild binary segmentation algorithm, a nonparametric, unsupervised process which aims to capture all changepoints within a time series. Constructed by Piotr Fryzlewicz, professor at the London School of Economics, the algorithm harnesses the strength of the CUSUM (cumulative sum) statistic to monitor changepoints throughout randomly selected subintervals of the data. We discuss the procedure of the algorithm in the next section, as well as the CUSUM statistic. However, what is most notable about the algorithm is its low computation complexity (runtime of O(n log n)). Hence, if this algorithm is indeed effective in finding changepoints, it could set the standard for future changepoint detection methods.

2 Algorithm

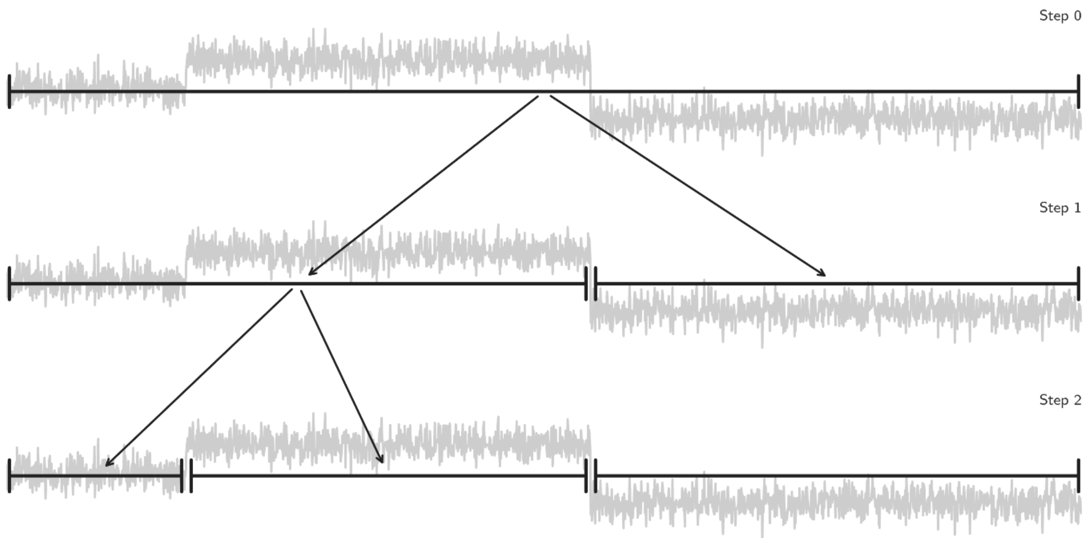
In this section, we examine the WBS algorithm in detail and provide some remarks about the CUSUM statistic. The algorithm operates very similarly to standard binary segmentation, but incorporates a "wild" component by randomly selecting certain subsamples of the time series and applying the CUSUM statistic to each interval. According to Fryzlewicz, localizing the CUSUM statistic to the random intervals eliminates some of the issues that arise within multiple change point detection when using a global CUSUM value. He also remarks that doing so results in a more optimal heuristic for changepoint detection.

Below is a high-level outline of the algorithm (pseudocode available in original paper):

Denote F_M^T as the set of M independently sampled intervals of the time series X_t drawn with replacement. The endpoints of interval i are s_i and e_i. We note the intervals are of varying length.

1. Randomly select M subsamples from time series.
2. Calculate CUSUM statistic on each subinterval and find global maximizer of CUSUM statistic between all intervals.
3. Compare this CUSUM value to the Strengthened Schwarz Information Criterion (establishes whether given changepoint exceeds significance threshold). Note that at this stage, no changepoints have been identified.
    a. If this value is greater than the SSIC, add to the list of changepoints.
    b. Else, stop the algorithm. No changepoints were detected.
4. If the maximizer in step 3 was significant, divide the time series at this point and iteratively run WBS on each newly formed segment of data.
    a. In particular, for each of these segments of data, M_j, calculate the CUSUM on all of the randomly selected subintervals whose endpoints are entirely within M_j, i.e., intervals where s_i and e_i are both within M_j.
    b. Find the CUSUM maximizer point and determine if its CUSUM score is higher than the SSIC. If yes, split the segment M_j at this point and iterate WBS again.
5. Repeat this process until algorithm terminates, namely when the segments contain no intermediate points or the maximizer found at each iteration becomes insignificant.

End of Algorithm



In the above process, we mentioned the CUSUM statistic and strengthened Schwarz information criterion (SSIC) without explicitly defining these terms. We choose not to address the SSIC in great detail, as it is beyond the scope of this paper. However, we note that the SSIC criterion is unique in that it is determined by the dataset and algorithm itself. Namely, it depends on the length of the time series as well as the locations of adjacent, previously identified changepoints.

The CUSUM measure, short for cumulative sum, is a well-known metric to calculate and identify changepoints. The statistic was first introduced by E.S. Page, a Cambridge computer scientist, and later modified by George Alfred Barnard, a renown quality control statistician. In general, the CUSUM statistic seeks to find points where some parameter of a given probability distribution changes. It achieves this goal by calculating a cumulative sum over points in a data series and marking points whose CUSUM exceeds some specified threshold as changepoints. The configuration of the CUSUM statistic varies depending on the parameter of interest.

3 CUSUM Statistics

In WBS, Fryzlewicz defines the CUSUM as follows:

Let X be the vector (X_s,…,X_e) of observations in some subinterval of the time series. Now let Y, a vector of contrasts, be defined as follows:

$$\tilde{X}^b_{s,e} = \sqrt{\frac{e-b}{n(b-s+1)}}\sum_{t=s}^{b} X_t - \sqrt{\frac{b-s+1}{n(e-b)}}\sum_{t=b+1}^{e} X_t,$$

Where $s \leq b < e$, with $n = e - s + 1$. Then we defined the CUSUM of this interval as the inner product of X and Y.

More intuitively, for an interval of data *(s,e)*, the CUSUM statistic finds the intermediate value *b* which maximizes the difference in mean between the data contained in the interval *(s,b)* and the data contained in the interval *(b,e)*. Furthermore, as a consequence of its construction – as well as the nature of the SSIC – as the interval size shrinks, theoretically, the likelihood of detecting changepoints decreases.

4 Results and Findings

In this section we assume the reader is familiar with some of the preliminary results and analyses discussed in the paper's corresponding presentation (slides attached below for reference). As such, we focus on some of the outcomes of WBS and possible explanations of these phenomena. Later, we also address some future of areas of research that may be of interest to those seeking to explore WBS in greater depth.

Before running WBS on Microsoft's stock return data, we first applied the algorithm to two simpler data sets. First, we generated 200 data points from an AR(3) model and inserted a mean shift at X_{100}. On this time series, WBS successfully identified a changepoint at time index 99, which was extremely close to the true changepoint value. We next ran WBS on Microsoft Stock Index data from 2000 to 2019 and again, the algorithm yielded exceptional output. In fact, the piecewise mean function that R generated when plotting WBS was almost perfectly superimposed on top of the actual data plot. The overarching results of these tests were nothing short of outstanding.

The next phase of analysis, then, was to test WBS on a more complex data set – in particular, one that was more erratic and "stationary" in nature (note: We use stationary here in a very crude sense. The data does not demonstrate true stationary, but rather, appears more stationary in nature). Since these datasets tend to have more subtle changepoints, assessing WBS' performance on such data would provide for an interesting reference point to the algorithm's strengths and weaknesses.

Hence, we applied WBS to Microsoft Stock Return data, which – though seems stationary on first glance – is highly dynamic and varying. As expected, the algorithm was more muddled in its conclusions. Many of the algorithm's outcomes were promising, but there were some confusing accompanying results as well.

Overall, the algorithm detected several known historical changepoints in Microsoft stock returns, including the 2000 and 2008 recession, the Microsoft acquisition of LinkedIn, and Microsoft's successful ventures into the cloud and IoT space in 2017. We claim that the only significant changepoint that the algorithm failed to identify was the 2009 resurgence of Microsoft after the 2008 recession. We believe that this is a direct consequence of the algorithms somewhat greedy approach; on each iteration, WBS fails to consider previously detected changepoints. This implies that as the algorithms runs, it loses contextual information about the whole time series, leading to some changepoints going unrecognized.

In this specific case, the 2008 recession was one of the first changepoints detected by WBS. And, because the segments on either side of the changepoint were sufficiently large, the algorithm encountered difficulty trying to identify true changepoints which existed close to previously selected changepoints (recall that the SSIC depends on the distance to adjacent changepoints). To counter this issue, we propose a hybrid Sliding Window and WBS algorithm to improve the algorithm's performance in these instances. Under this revamped version of WBS, at each changepoint detected in the algorithm, the algorithm runs a Sliding Window subroutine for some sufficiently large interval w centered around the identified changepoint. In doing so, the algorithm can view data points which lie outside the segment that WBS is currently processing. We note that this algorithm would also maintain a log-linear runtime, keeping its computational complexity on par with that of WBS. The major drawback of this method is that it introduces a new hyperparameter to the algorithm, namely w. As such, empirical testing or more thorough analysis would have to be performed in order to find optimal values of w.

Indirectly, this observation about the algorithm's execution on intervals of certain length also clarifies why there were strong changepoint clusters near the ends of the time series, an event which Fryzelwicz claims is unlikely to occur under WBS. As the intervals become more and more granular throughout WBS, the volatility of the time series rendered WBS largely inadequate in distinguishing between extreme data points and true changepoints.

More explicitly, one of the significant pitfalls of WBS is that as the subinterval size decreases, the algorithm's robustness to outlier data significantly dampens. Again, we remark that this is a direct result of having fewer intermediate data points, and consequently, an inability to discern true changepoints from outliers in the distribution. Such conflation of changepoints and outliers is one of the more concerning results of WBS, which directly ensues from the poor adaptability of the algorithm's Schwarz Information Criterion. It would be interesting to examine whether incorporating a penalty for the iteration number, thereby increasing the SSIC threshold on each iteration, would rectify this issue. In turn, this would hopefully result in the algorithm placing stricter requirements for future changepoints, optimizing its correctness in the presence of outliers.

Interestingly, although WBS positively affected the model selection process for each segmented window of data, it failed to change the model's forecasting ability in any substantial measure. This was a more peculiar result that we stumbled upon, and it was unclear why the model produced similar predictions. We consider three possible explanations for this trend, outlined below:

1. There was little to no change in the mean. Though WBS detected changepoints throughout the time series, the shifts in the mean value were still fairly small. Perhaps the relative centering around zero made it unlikely that the model's predictive power would improve dramatically.
2. The noise of previous data points was fairly weak. We consider this possibility, especially for the first three constructed windows, because the model selection process chose MA models for the first three windows. The only difference between each model was the order, parameter q. As a result, it is plausible that, thought changepoints existed, they were not particularly influential in estimating the full model. We do not believe this to be a likely scenario (reason outlined below) but we felt it was worth mentioning to the reader.
3. The volatility of the data made it impractical to perform forecasting. We believe that this is the most likely reason for such outcomes in the predictive power for each model. Reliable forecast methods are contingent on reliable data. However, after fitting each of our models, we confronted significantly high sigma values that accordingly generated wide confidence intervals.

Of course, it is also possible that some combination of these factors led to the forecasting outcomes for both models. Testing WBS on other datasets with fluctuating values might help better answer this question.

Overall, the analyses described in this paper largely serve as an introductory exploration into WBS. In the future, researchers may wish to apply WBS on a diverse collection of datasets to gain a more comprehensive understanding of the algorithm's statistical underpinnings and applications. Moreover, running WBS with a variety of CUSUM statistics may also provide statisticians with more holistic grounds for identifying changepoints. In particular, it would be interesting to see if combining the CUSUM statistics used to detect changes in the mean and variance and testing them simultaneously would generate trustworthy results. This idea seems reasonable on face value, but thorough analysis would verify such an approach is indeed credible.

Acknowledgements

I would like to thank Dr. Danning Li for all her contributions to the making of this paper.

Appendix A: R Script

Below is a copy of relevant R Code used in analyzing Microsoft stock data.

```
window1 = msft_returns[1:384]
window2 = msft_returns[385:1965]
window3 = msft_returns[1966:3977]
window4 = msft_returns[3978:length(msft_returns)]


# model selection on whole msft returns data
select(MA(32),msft_returns,include.mean = TRUE)
select(ARMA(3,3), msft_returns, include.mean = TRUE)
select(AR(25), msft_returns, include.mean = TRUE)

#analyzing residuals for four windows
corr_analysis(window1)
corr_analysis(window2)
corr_analysis(window3)
corr_analysis(window4)

# performing model selection on four windows and summing AIC scores
select(MA(16),window1,include.mean=TRUE) #MA(5)

select(MA(16),window2,include.mean = TRUE) #MA(1)
select(ARMA(4,4),window2,include.mean=TRUE)

select(MA(18),window3,include.mean = TRUE) # MA(11)

select(AR(8),window4,include.mean = TRUE) #AR(8)

# performing prediction using full model WINDOW 2
corr_analysis(msft_returns[1:1935])
select(ARMA(3,3),msft_returns[1:1935],include.mean=TRUE)
msft_returns.half = gts(msft_returns[1:1935])
full.data.model = estimate(ARMA(3,2),msft_returns.half)

forecast.full = predict(full.data.model,n.ahead=25,level = .95)
mape.full = median(abs(forecast.full$pred - msft_returns[1935:1959]))


# performing prediction using reduced model WINDOW 2
msft.returns.reduced = gts(msft_returns[385:1935])
reduced.data.model = estimate(MA(1),msft.returns.reduced)

forecast.reduced = predict(reduced.data.model,n.ahead=25,level = .95)
mape.reduced = median(abs(forecast.reduced$pred -
                            msft_returns[1935:1959]))

# performing prediction using full model WINDOW 3
corr_analysis(msft_returns[1:3947])
select(ARMA(3,3),msft_returns[1:3947],include.mean=TRUE)
msft_returns.half.3 = gts(msft_returns[1:3947])
full.data.model.3 = estimate(ARMA(3,3),msft_returns.half.3)

forecast.full.3 = predict(full.data.model.3,n.ahead=25,level = .95)
mape.full.3 = median(abs(forecast.full.3$pred -
                           msft_returns[3948:3972]))
```

```
# performing prediction using reduced model WINDOW 3
msft.returns.reduced.3 = gts(msft_returns[1966:3947])
reduced.data.model.3 = estimate(MA(11),msft.returns.reduced.3)

forecast.reduced.3 = predict(reduced.data.model.3,n.ahead=25,
                              level = .95)
mape.reduced.3 = median(abs(forecast.reduced.3$pred -
                             msft_returns[3948:3972]))
```

References

Aminikhanghahi, Samaneh, and Diane J Cook. "A Survey of Methods for Time Series Change Point Detection." *Knowledge and Information Systems*, U.S. National Library of Medicine, May 2017, www.ncbi.nlm.nih.gov/pmc/articles/PMC5464762/.

Fryzlewicz, Piotr. "WILD BINARY SEGMENTATION FOR MULTIPLE CHANGE-POINT DETECTION." *Annals of Statistics*, Institute of Mathematical Statistics, 4 Nov. 2014, www.jstor.org/stable/pdf/43556493.pdf?seq=1#page_scan_tab_contents.

"Window-Based Change Point Detection." *Window-Based Change Point Detection – Ruptures Documentation*, 3 Jan. 2018, ctruong.perso.math.cnrs.fr/rupturesdocs/build/html/detection/window.html.

CHANGE POINT DETECTION

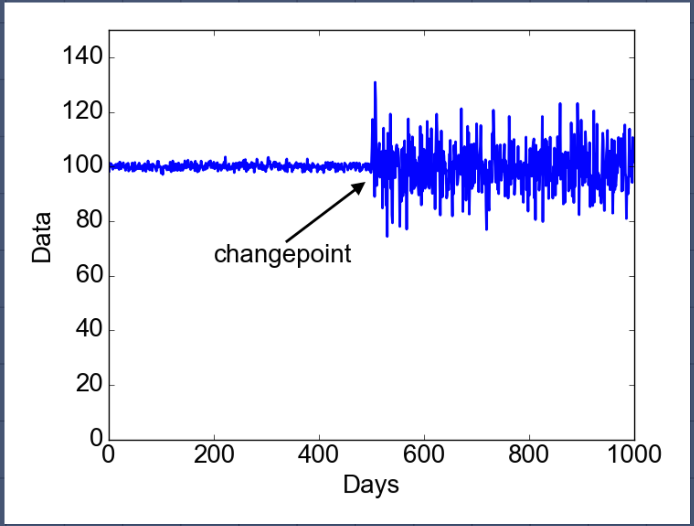
WILD BINARY SEGMENTATION METHODS

BACKGROUND

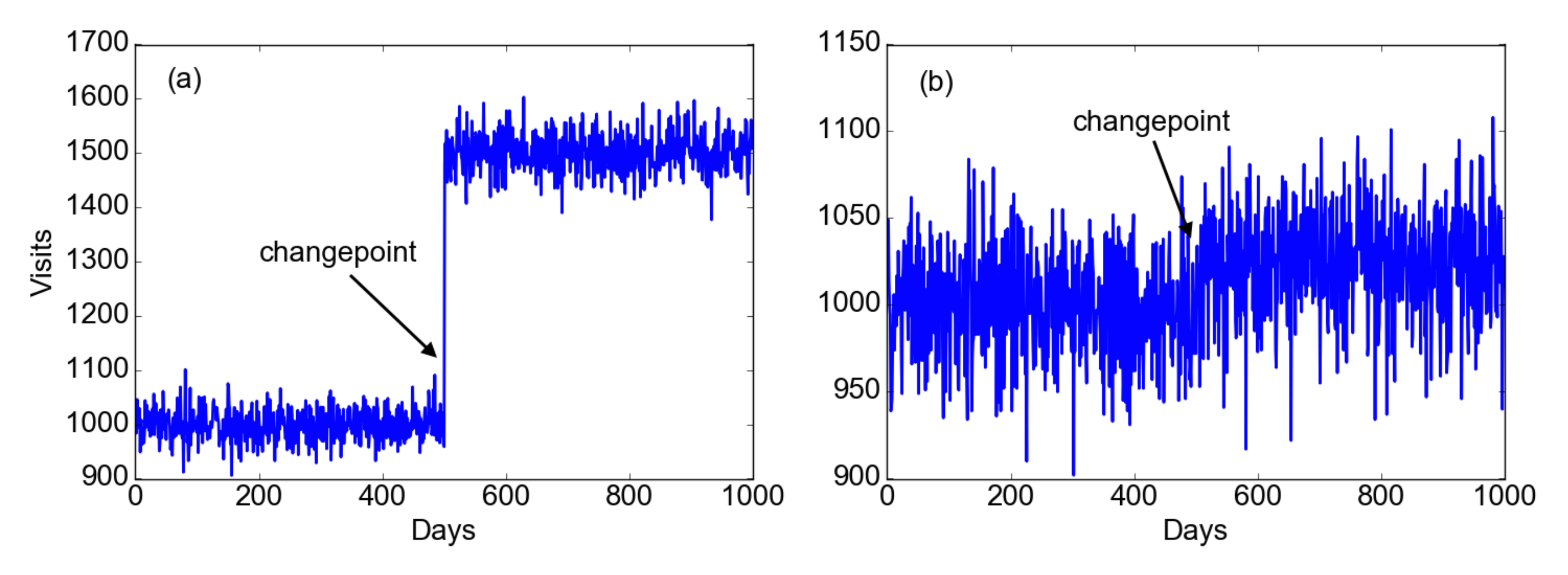
Introduction to changepoint detection and WBS

1

CHANGEPOINT DETECTION



PROCESS OF IDENTIFYING POINTS IN TIME SERIES DATA WHEN DISTRIBUTION CHANGES



EXAMPLES: EASY AND HARD TO DETECT CHANGEPOINTS

WILD BINARY SEGMENTATION

- Author: Piotr Fryzlewicz
- Introduces “Wild” Component to algorithm
- Holistic view of data context

PROCEDURE

FORM SUBSAMPLES

Randomly draw M subsamples with replacement

CALCULATE CUSUM

Calculate CUSUM statistic for each subsample and find data point with max CUSUM value

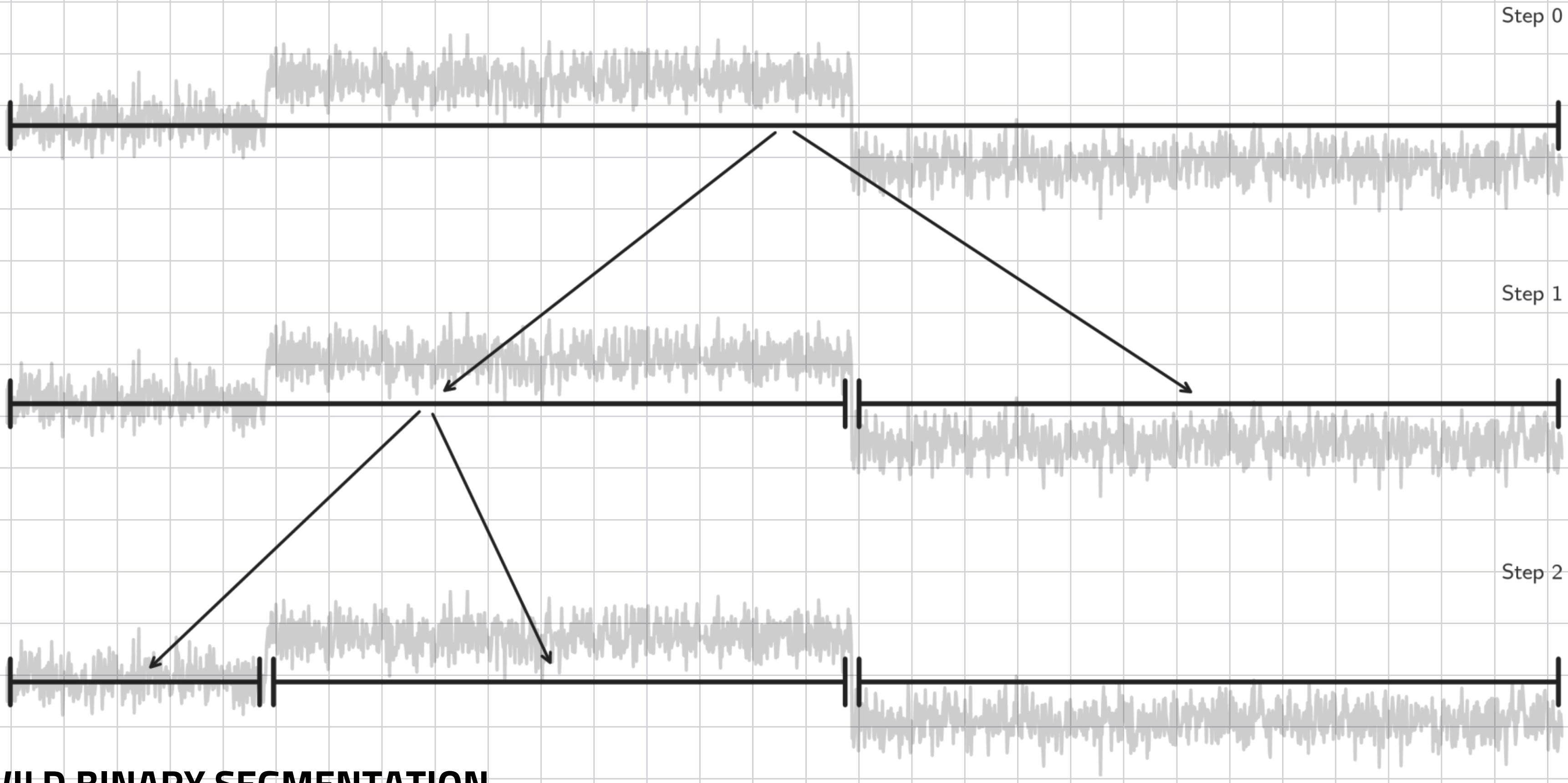
ITERATE

If CUSUM maximizer is significant, divide data and iteratively run WBS

CUSUM STATISTIC

- CUmulative SUM statistic aims to detect contrasts in time series by calculating cumulative sum for all data

$$\tilde{X}^{b}_{s,e} = \sqrt{\frac{e-b}{n(b-s+1)}}\sum_{t=s}^{b} X_t - \sqrt{\frac{b-s+1}{n(e-b)}}\sum_{t=b+1}^{e} X_t,$$



WILD BINARY SEGMENTATION

EXAMPLES

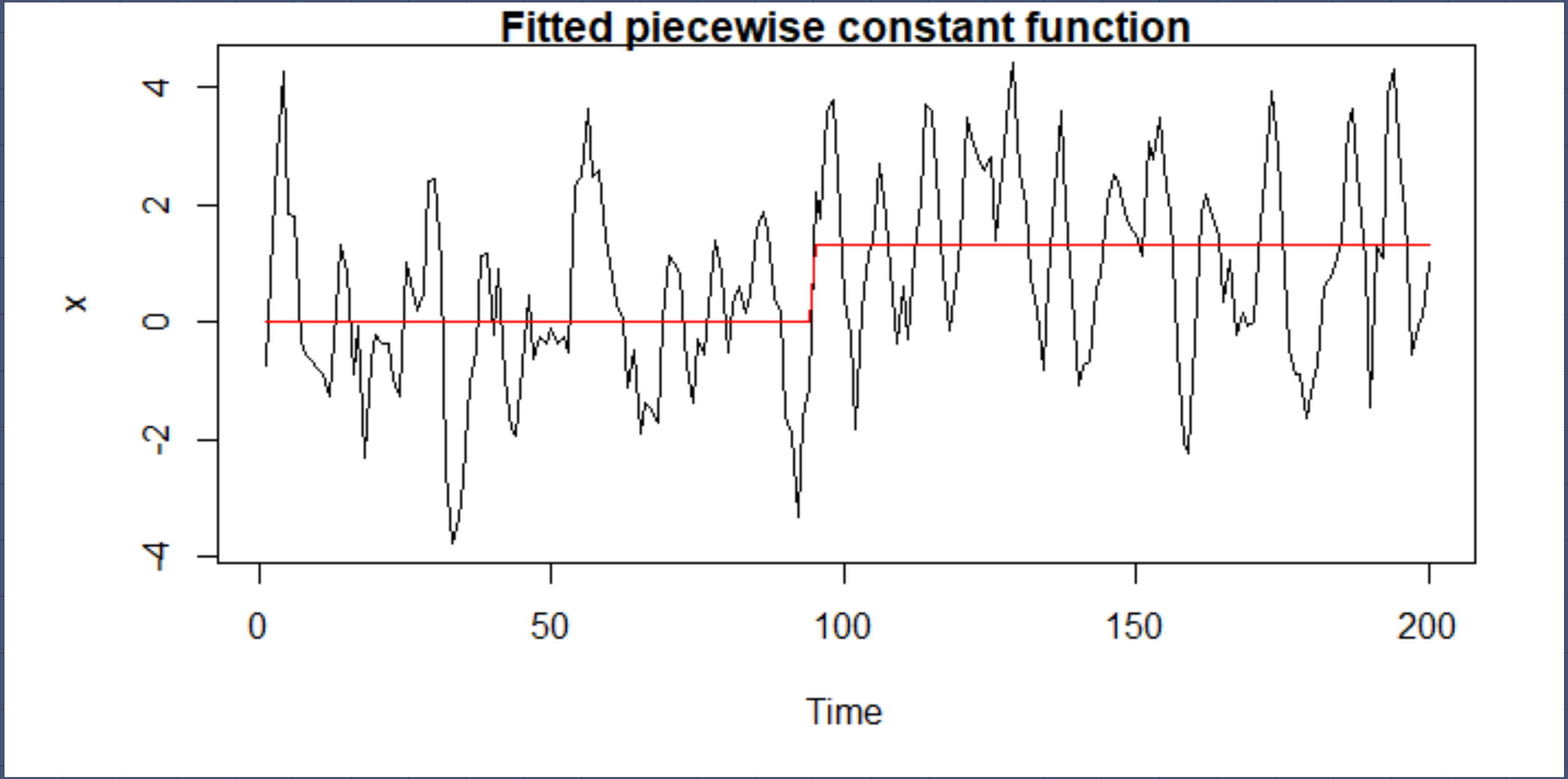
2

Applying WBS to Generated and Real Data

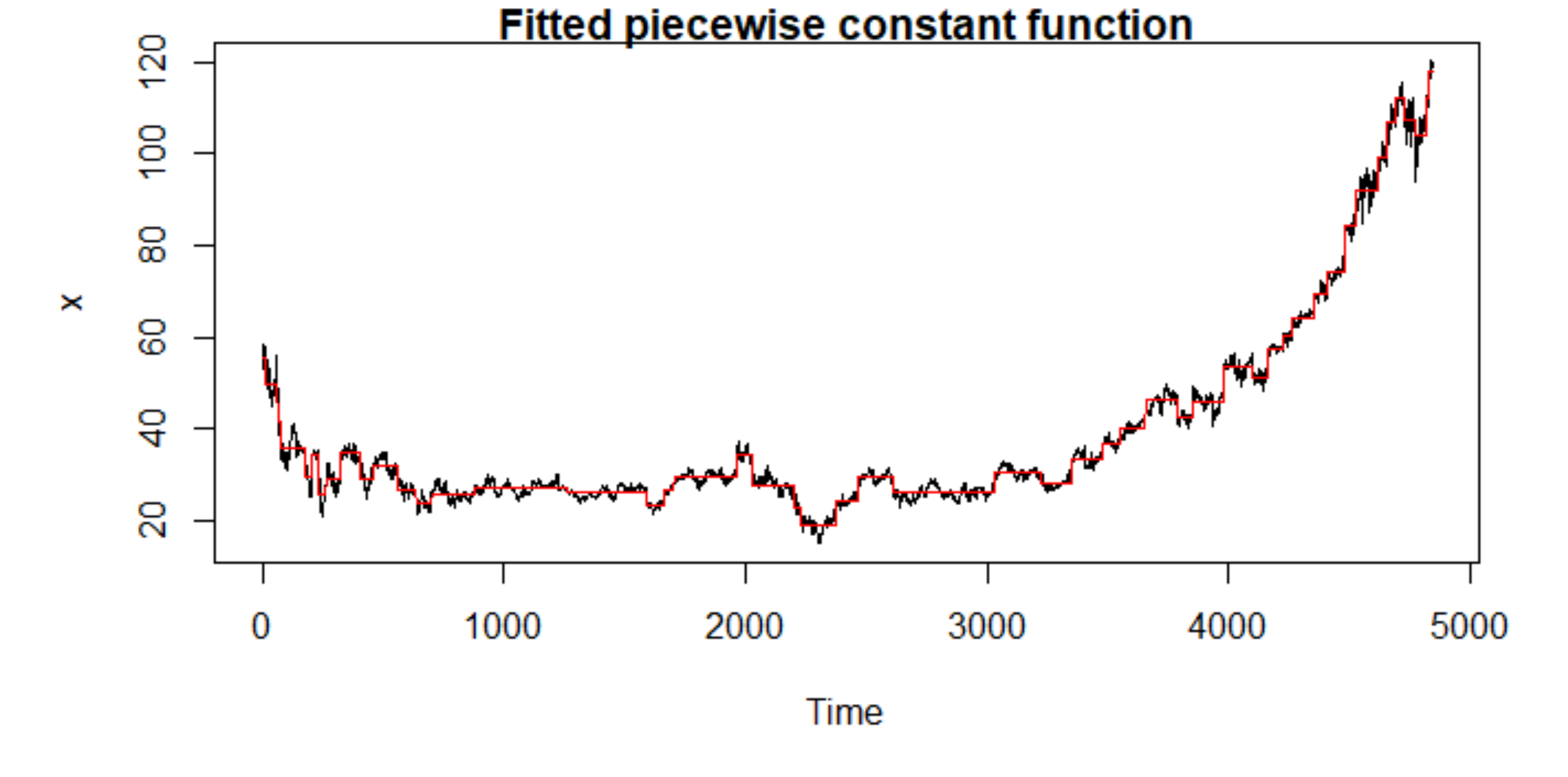
TWO APPLICATIONS

1. Generated 200 observations of AR(3) model with inserted mean shift at X_{100}

2. Ran WBS on Microsoft Stock Index 2000 - 2019



AR(3) WBS OUPUT



MICROSOFT STOCK INDEX OUTPUT

QUESTION: HOW WILL WBS RUN ON “STATIONARY” DATA?

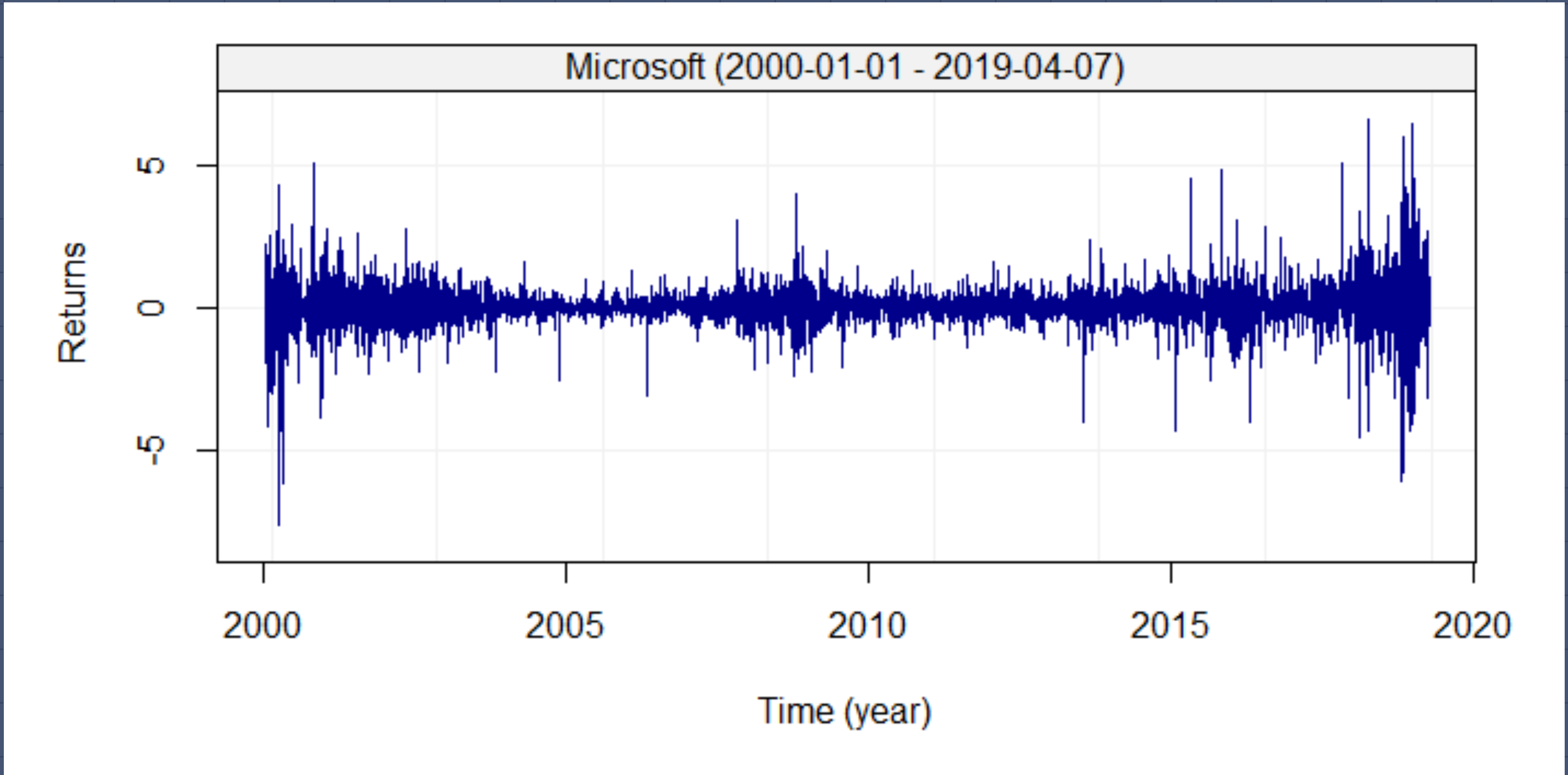
ANALYSIS

Evaluating WBS on Microsoft Data

3

STOCK DATA

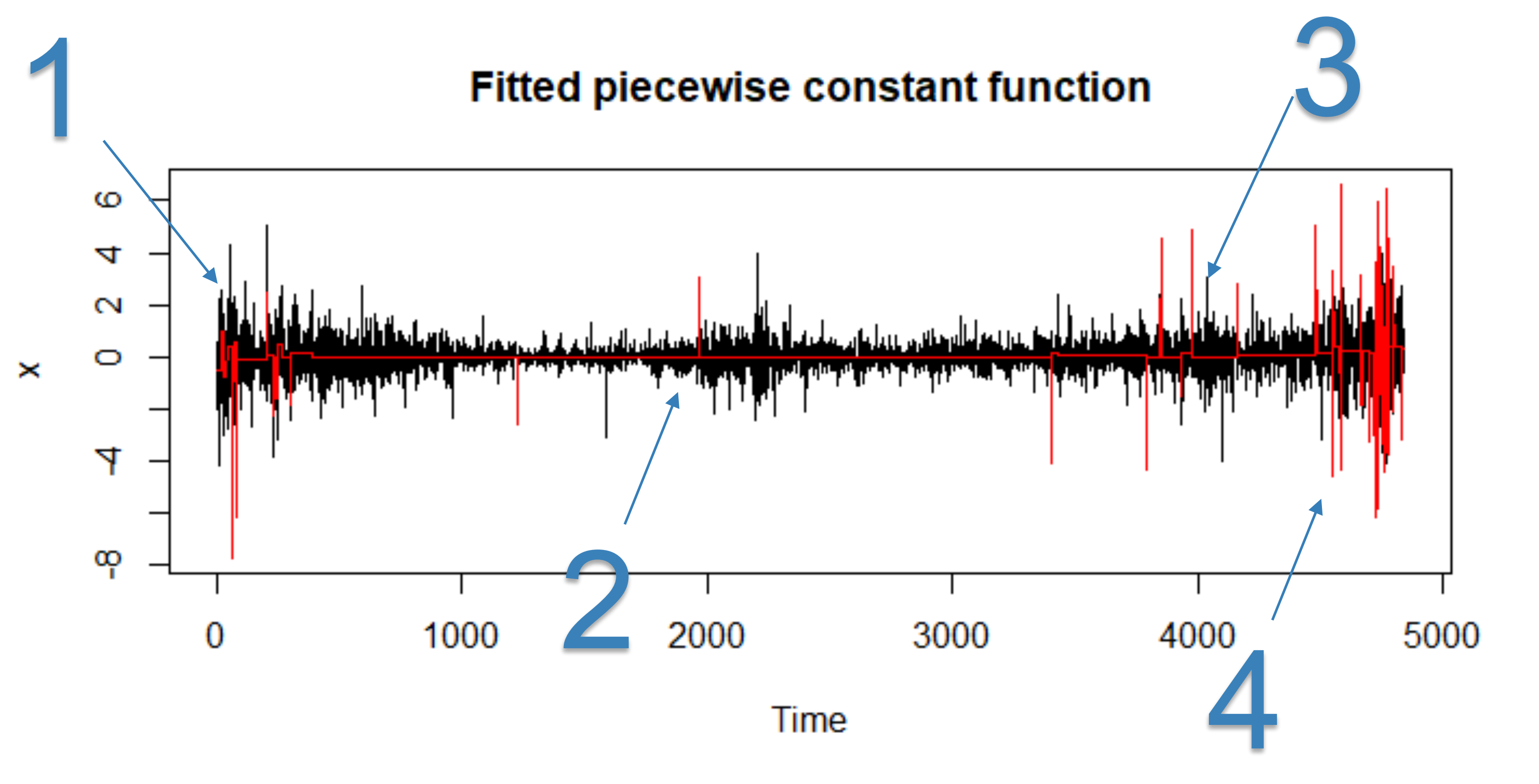
- **Apply to data on stock returns from Microsoft**
- **Compare changepoints to real-world events**



TIME SERIES: MICROSOFT RETURNS

PHASES OF ANALYSIS

RUN WBS	PARTITION DATA	FIT MODELS
Apply WBS algorithm onto data to detect changepoints	Divide data into discrete intervals based on detected changepoints	Estimated time series models for each section of partitioned data and analyze residuals



WBS PLOT OF MICROSOFT STOCK RETURNS

TIMELINE

2000

1 2000s recession caused several stock indices to fall rapidly

2008

2 Great recession – enough said

2015

3 Satya Nadella oversees several successful purchases (LinkedIn)

2018

4 Significant moves in IoT and Cloud space, Azure Sphere

OBSERVATIONS

WBS Output

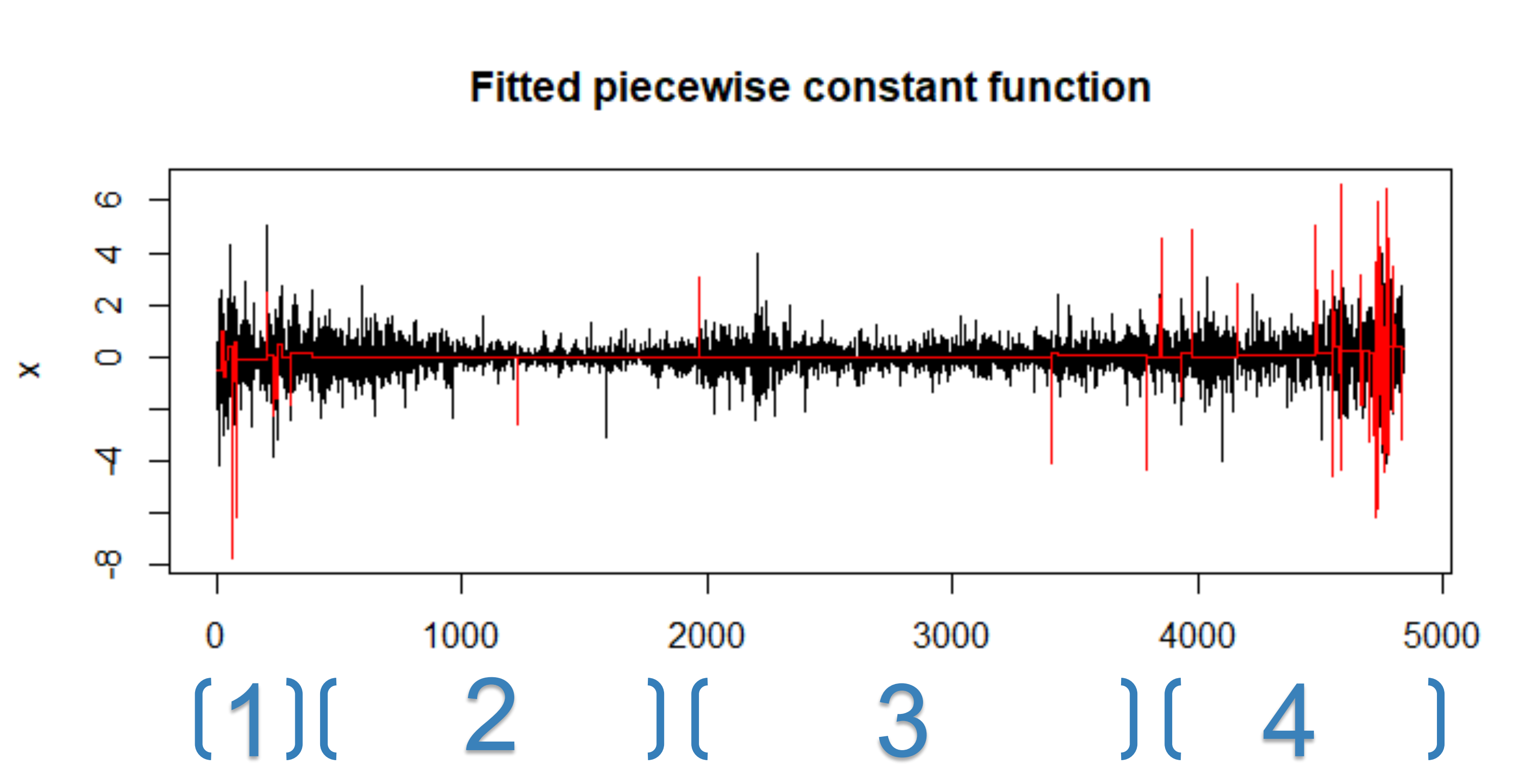
- Sharp spikes in piecewise function (no mean change)
- Significant changepoint clustering at endpoints
  - Granularity of volatile data

Real World Events

- Fairly strong correspondence to real world events
- Resurgence after recession not captured (exception)

METHODOLOGY

- Divided data based on 4 most significant changepoint regions
- Fit models for each segment of data
- Compare AIC, forecasting ability of piecewise model to full model



PARITION OF DATA

PLOT ACF/PACF

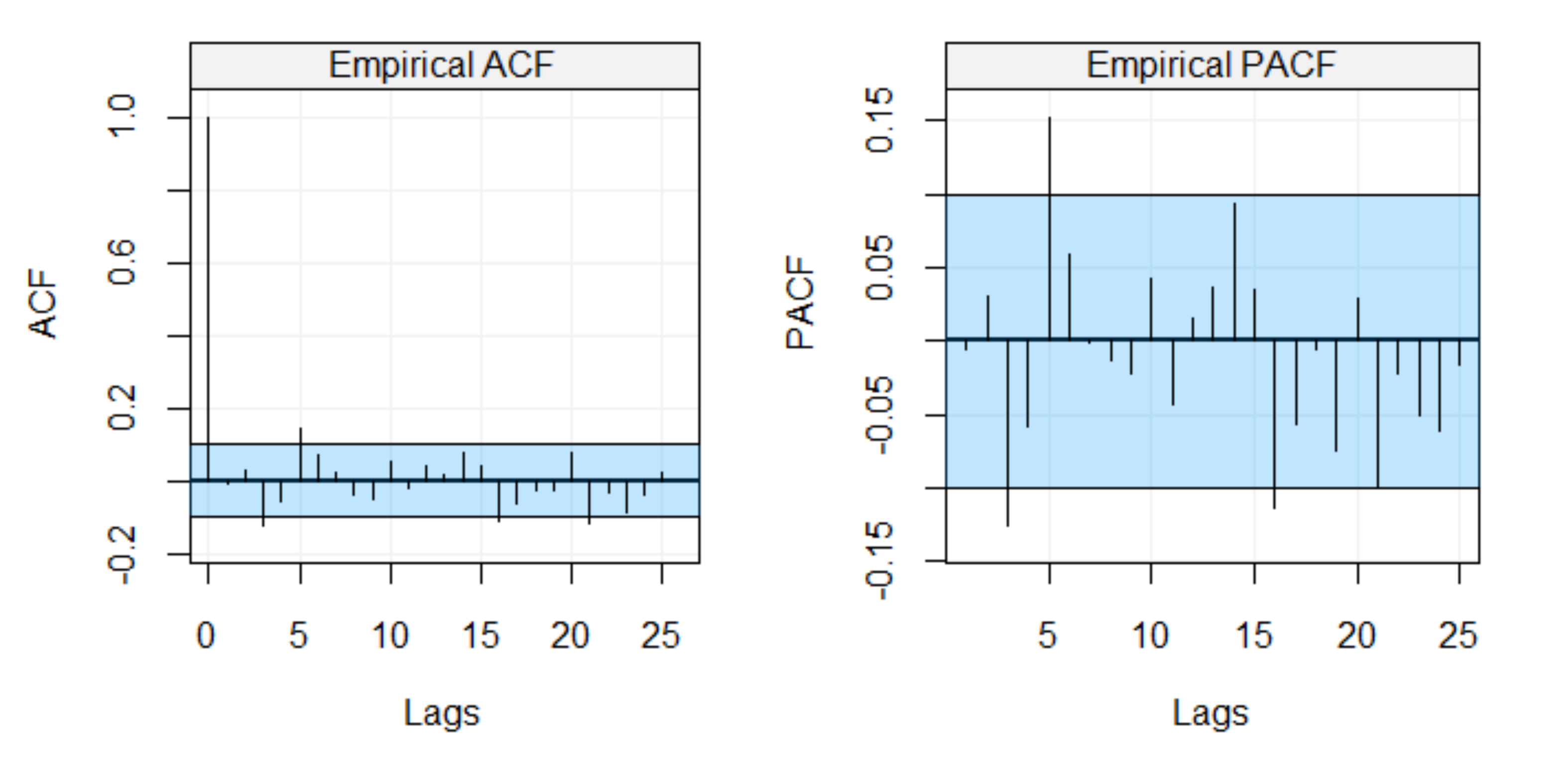
Gauge preliminary model

MODEL SELECTION

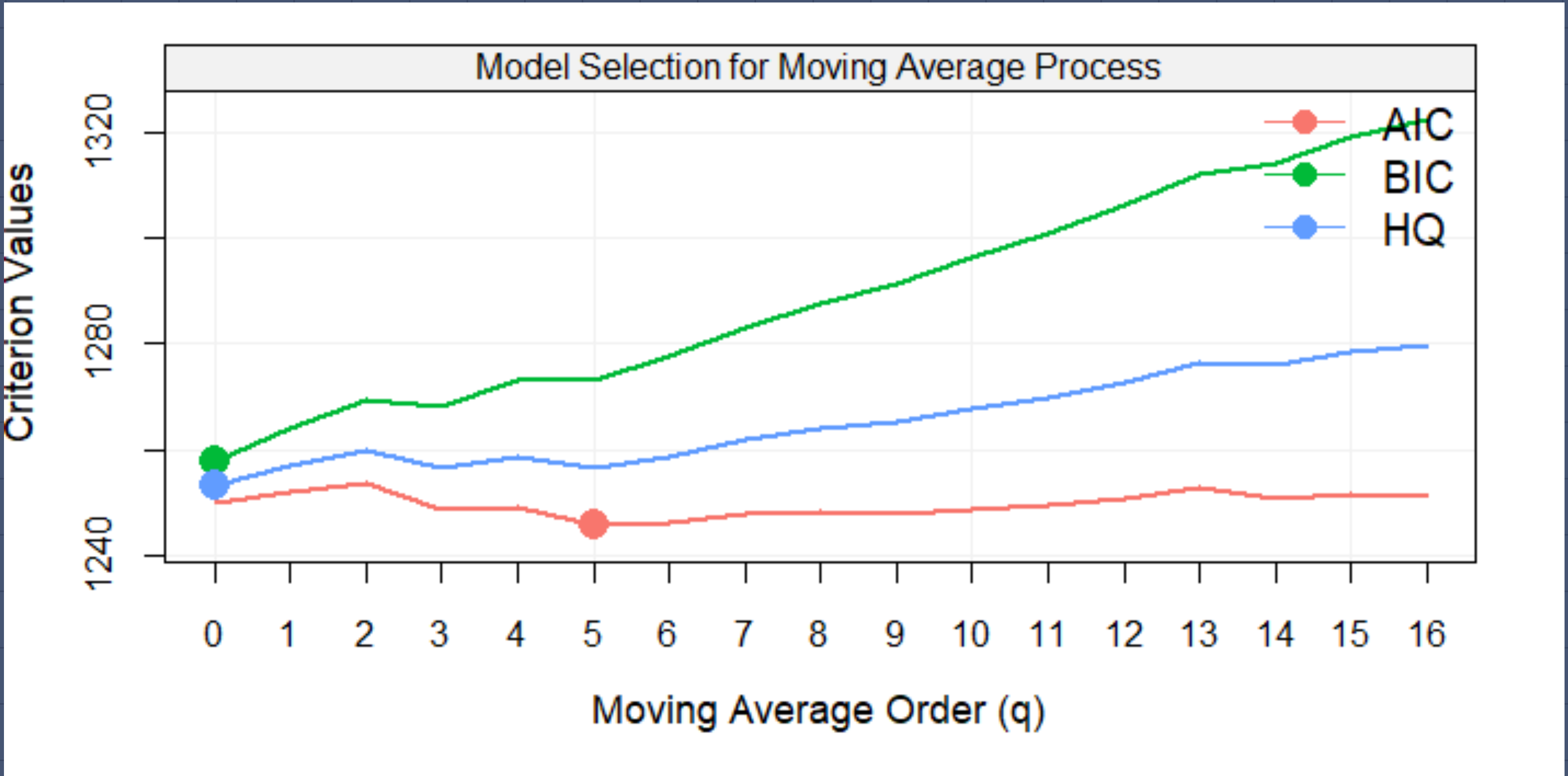
Use AIC Criterion to choose from candidate pool

ANALYZE RESIDUALS

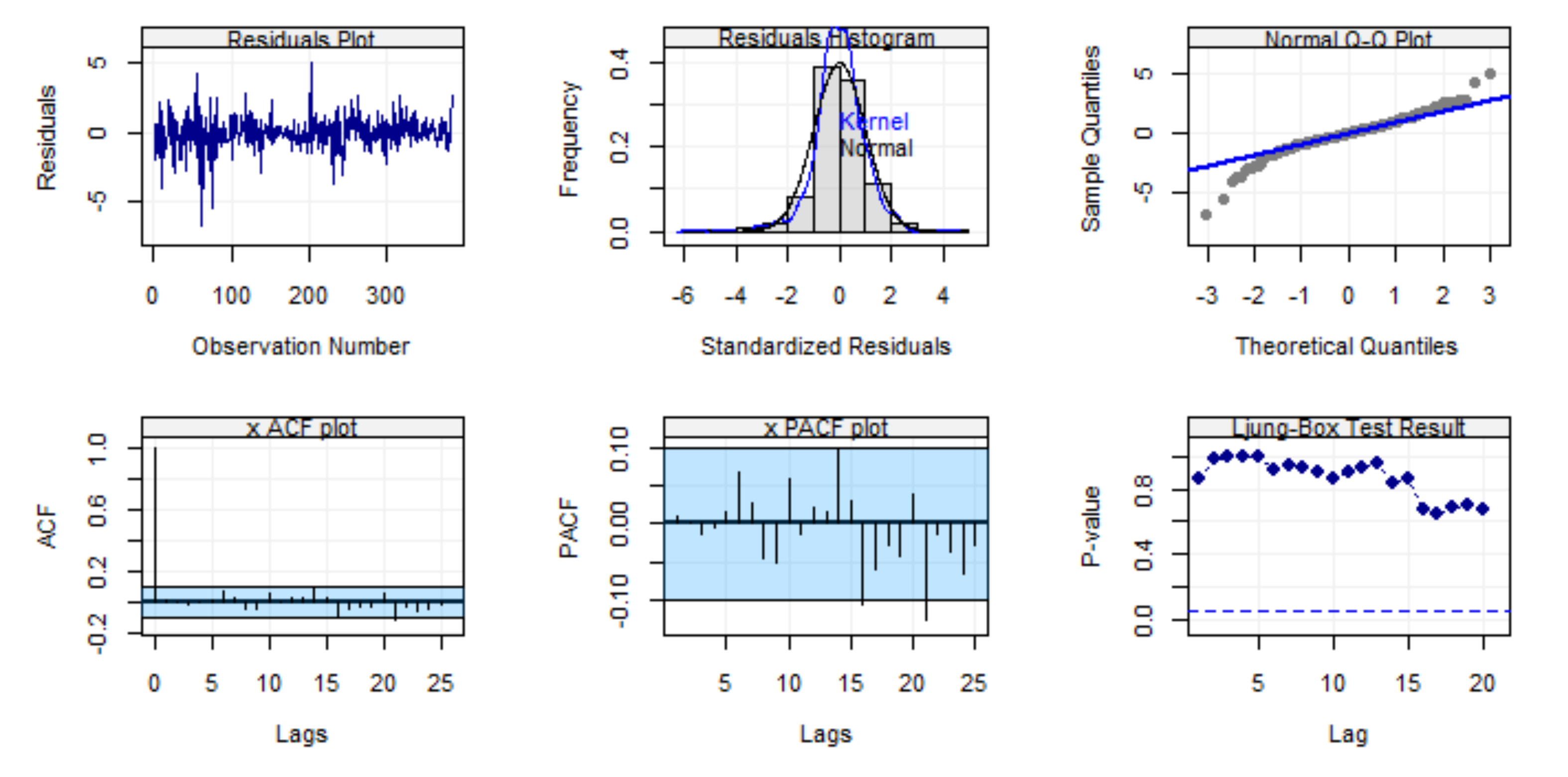
Ensure model captures dependency



CORRELATION ANALYSIS: ACF AND PACF CHARTS OF WINDOW 1



MODEL SELECTION FOR WINDOW 1

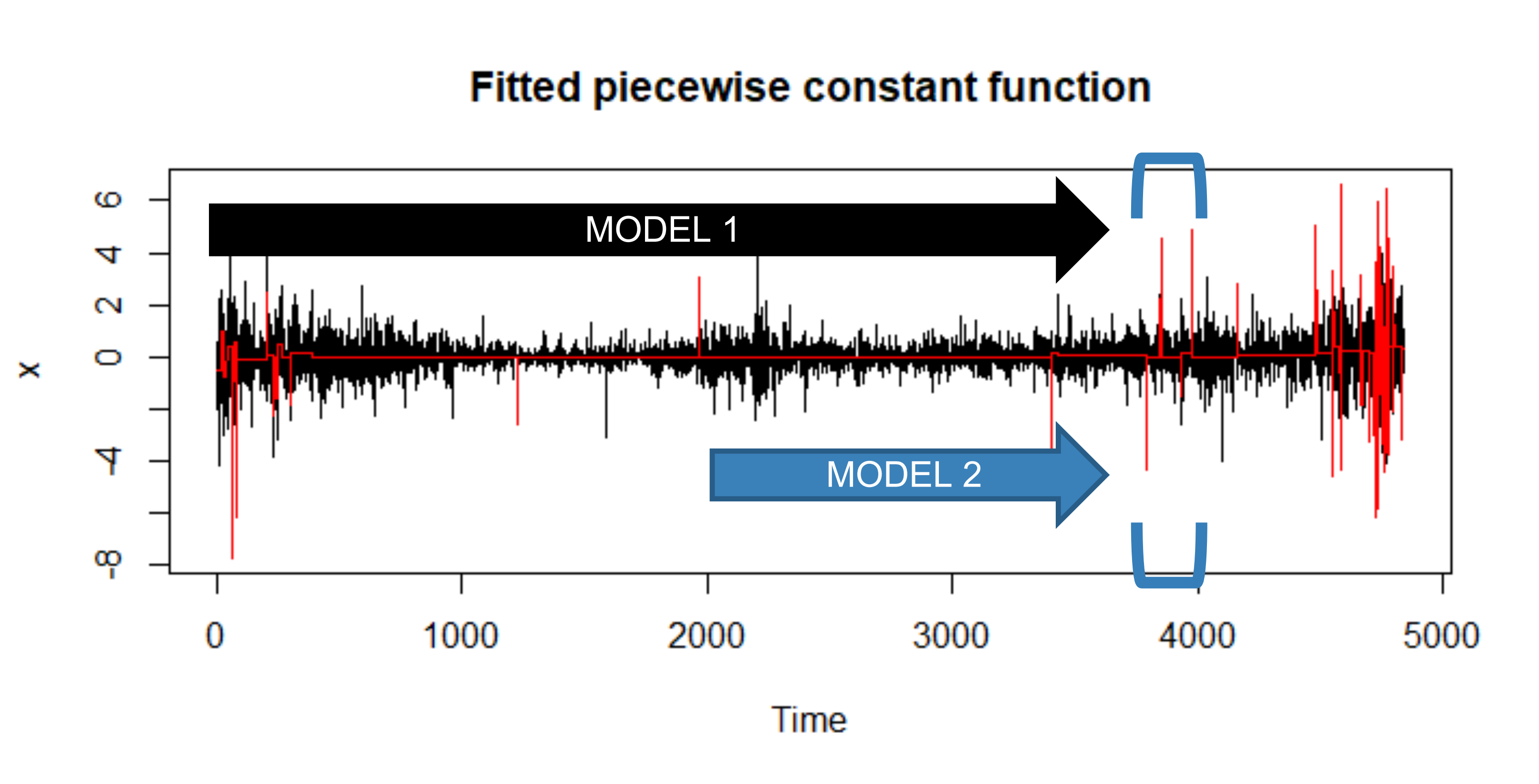


RESIDUAL ANALYSIS OF MA(5) MODEL

PIECEWISE MODEL

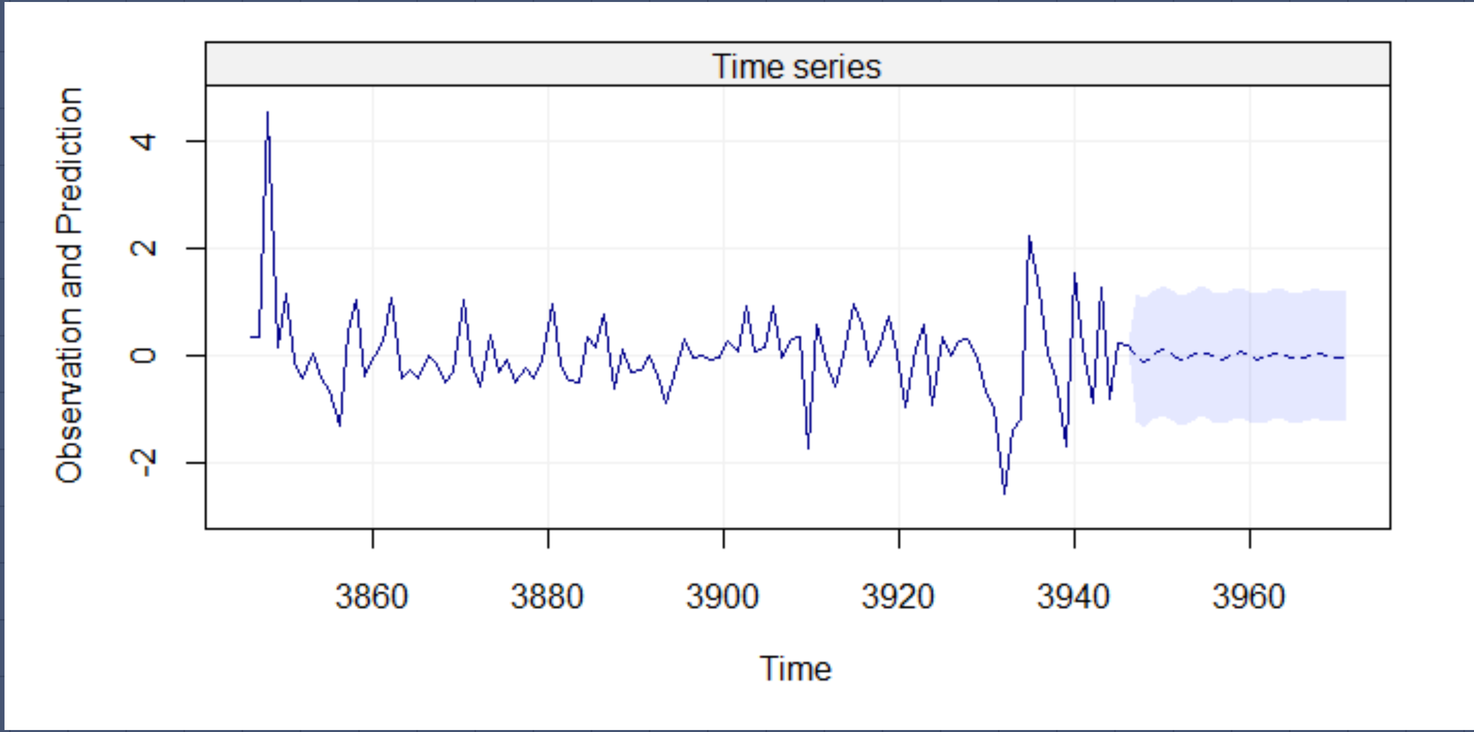
WINDOW	MODEL	ORDER	AIC
1	MA	5	1245.58
2	MA	1	2084.76
3	MA	11	3278.55
4	AR	8	2724.39

ARMA(3,3) MODEL WAS **NOT** CHOSEN FOR ANY WINDOW

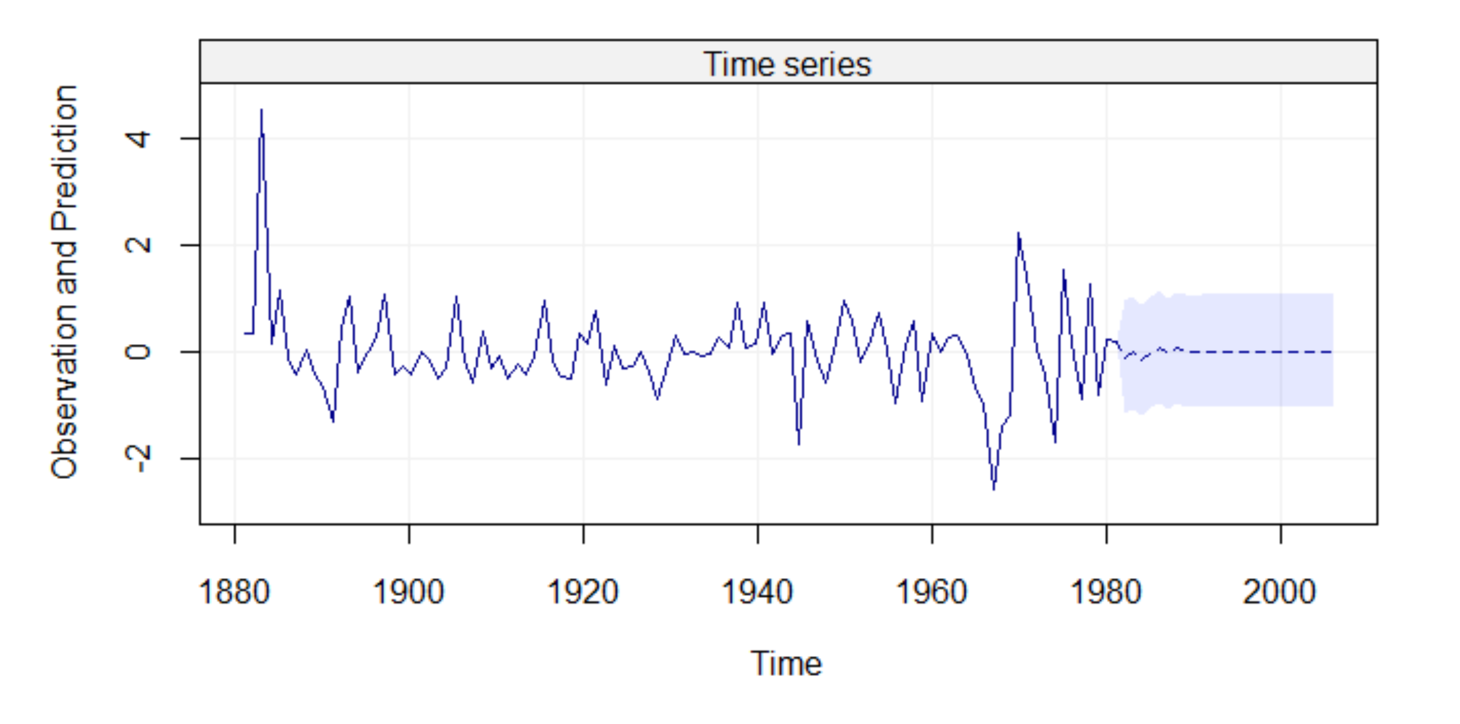


COMPARING PREDICTIVE ABILITY

FORECASTS: MODEL 1



FORECASTS: MODEL 2



COMPARISON

Model 1

- 100 percent empirical coverage
- MAPE = .32

Model 2

- 100 percent empirical coverage
- MAPE = .34

CONCLUSION → Predictive Abilities are comparable (Model 1 slight edge)

REASONS FOR MINIMAL IMPROVEMENT

- Little-to-no change in mean
- “Noise” of previous data may be weak
- Difficulty of predicting volatile data

CONCLUSIONS

Overall findings, remarks, and future work/improvements

4

FINDINGS

- Model fit seems to improve significantly
- Predictive power remains unchanged
- Reasonable correspondence to real-world events
- Clustering of certain data points

AREAS OF IMPROVEMENT

- Incorporate information of previous iterations into algorithm
  - Hybrid Window Screening – WBS approach

- Adjusting threshold for significant changepoints

FUTURE WORK

- Testing WBS on less volatile data
- Comprehensive analysis of WBS ability to detect changes of different parameters
- Develop WBS version with window screen method

SOURCES

https://money.cnn.com/2000/04/24/markets/markets_newyork/

https://www.fool.com/investing/2019/04/21/microsoft-makes-an-iot-push-with-a-new-acquisition.aspx

https://arxiv.org/pdf/1411.0858.pdf

QUESTIONS?