Wild Binary Segmentation for multiple change-point detection

Magnus Berg Sletfjerding
(Dated: March 6, 2019)

ARTICLE LINK

https://arxiv.org/abs/1411.0858

I. INTRODUCTION

The accurate treatment and analysis of noisy time series is an ubiquitous challenge across scientific disciplines. Within the field of single-molecule analysis of biomolecules, low signal-to-noise ratios are a constant problem when fitting models of stochastic change in piecewise-stationary data.

Binary Segmentation is a simple and powerful method for "slicing" data into pieces which come from the same distribution, by finding "change points" in a time series. While Binary Segmentation is widely utilized in the analysis of time series involving docking and binding on long (s) timescales, it performs significantly less well if the data exhibits multiple change points, especially if they are close to each other.

The article presented proposes an improvement on the classical Binary Segmentation algorithm, in order to analyze data with multiple changepoints accurately.

II. BINARY SEGMENTATION

The simplest time-series data model can be expressed as

$$X_t = f_t + \varepsilon_t, t = 1, \ldots, T$$

where f_t is a one-dimensional signal which has an unknown number of changepoints N, with unknown locations $\eta_1, \ldots, \eta_N$, and ε_t is a normal random variable centered at 0.

Classical Binary Segmentation utilizes the CUSUM statistic, which is 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 $s \leq b < e$ and $n = e - s + 1$.

The Binary Segmentation algorithm finds the first changepoint, b_0 by maximizing $|\tilde{X}^b_{s,e}|$, after which it will look for changepoints in the two new segments of the data.

Finally, the Binary Segmentation stops if the $|\tilde{X}^b_{s,e}|$ for any b_0 is lower than a set threshold.

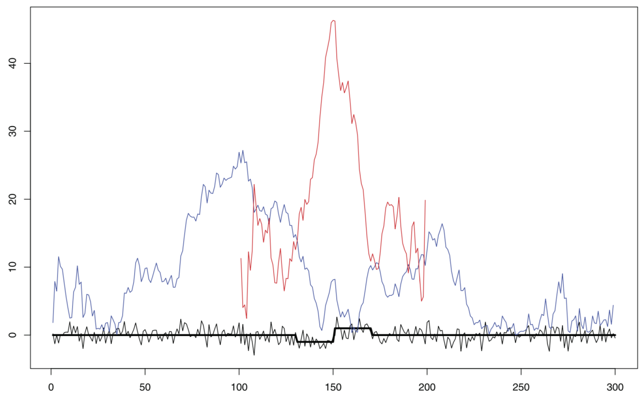

FIG. 1. True function in thick black, observed data in thin black, $|\tilde{X}^t_{0,300}|$ plotted in blue and $|\tilde{X}^t_{101,199}|$ plotted in red.

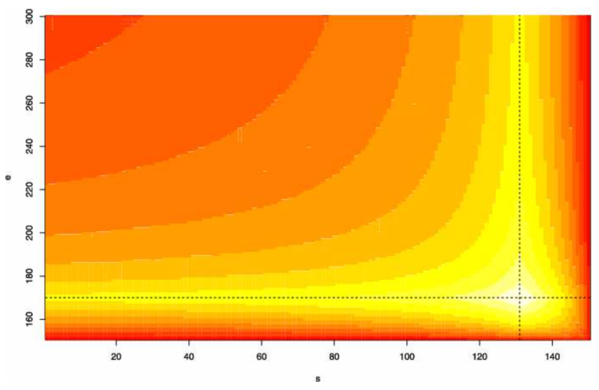

FIG. 2. Heat map of the values of $|\tilde{X}^t_{s,e}|$ as a function of s and e. Dashed lines indicate the maximum in either dimension, and in this case, the maximum is located at $(s, e) = (131, 170)$

A. Problems with the Binary Segmentation Algorithm

As seen in Figure 1, a large window (e.g. calculating $|\tilde{X}^t_{0,300}|$) will cause a misrepresentation of the data, and b_0 is estimated to be at $t = 100$. However, with a smaller window, the maximizer accurately finds the change point at $t = 150$. This point is further illustrated by Figure 2, where the values clearly show that the window size to find this particular change point is much smaller than 300 frames. The need for a more robust method for estimating the peaks is therefore obvious.

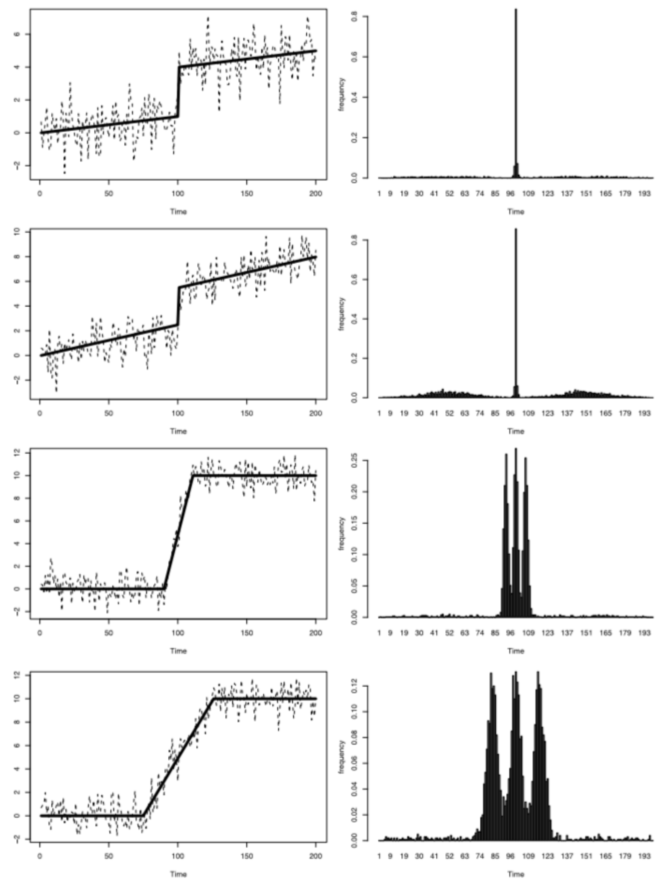

FIG. 3. Left: linear trend data (spots) and true functions (black lines). Right: frequency histograms of the found change points.

III. WILD BINARY SEGMENTATION

Wild Binary Segmentation works in a similar manner as Binary Segmentation, but differs in the way it finds b_0.

For Wild Binary Segmentation, F_T^M is a set of M randomly sampled intervals $[s_m, e_m], m = 1, \ldots, M$ where $[s_m, e_m]$ have been drawn from $1, \ldots, M$.

While Binary Segmentation uses a simple maximization of $|\tilde{X}_{s,e}^b|$ over the entire data, Wild Binary Segmentation instead maximizes $|\tilde{X}_{s_m,e_m}^b|$, hence forming a 2-dimensional space to maximize, with m on one axis, and b on the other.

Like the Binary Segmentation algorithm, the Wild Binary Segmentation also stops if $|\tilde{X}_{s,e}^b|$ is lower than a set threshold.

IV. PERFORMANCE

The Wild Binary Segmentation was compared to a series of publicly available datasets, and was shown to outperform classical Binary Segmentation in all cases. As seen in figure 3, Wild Binary Segmentation still misclassifies the middle of a linear (i.e. not instant) transition. Even so, the algorithm still manages to find the "true" change points, asserting its advantage over typical Binary Segmentation.

V. CONCLUSION

The Wild Binary Segmentation algorithm is a significant improvement upon the classical Binary Segmentation algorithm. While it does not perfectly find all change points, and misclassifies some points, it still does not mask points in the same way as the Binary Segmentation algorithm does, as its implementation ensures stochastic selection of segments. This prevents agains successive errors being repeated, as the repetition of random draws allows the user to withdraw a distribution of results, if they remember to rerun the experiments a series of times.