# Báo cáo về **Anomaly detection**

## Sử dụng package Twitter's Anomaly detection[8].

Bài viết chủ yếu đi vào ý tưởng của thuật toán, cũng như 1 số điểm mạnh, điểm yếu, cách sử dụng.
Các công thức cụ thể của giải thuật sẽ được đề cập ở phần tài liệu tham khảo.

## Mục lục

## Mục đích:

Phát hiện các điểm bất thường trong time series

## I. Ý tưởng chung:

+ Tách data thành 3 phần: trend, season, residual (dùng thuật toán Loess, thông qua hàm **stl()** trong **R**)

+ Tính lại residual = data - median- season

+ Chạy S-H ESD test trên residual (Dùng t-distribution, cải tiến bằng cách tính độ lệch chuẩn theo MAD,...)

## II. Cụ thể thuật toán

### II.1. Model Assumptions:

Giả sử data có normal distribution (giả thiết của S-H ESD test trong thuật toán)

### II.2. Tách data:

Trong R có 2 hàm hỗ trợ tách time series: decompose() và stl()

Hàm decompose sử dụng Moving Average, trong khi stl dùng Loess.

#### II.2.1. Ý tưởng chung của 2 thuật toán Moving Average và Loess:

##### II.2.1.1. Moving Average :

Ta tính data mới bằng cách, thay mỗi observation bằng trung bình cộng của 2 observations gần nó. (Các observation có mức độ quan trọng như nhau, hay trọng số của các observations được xem bằng nhau)[10]

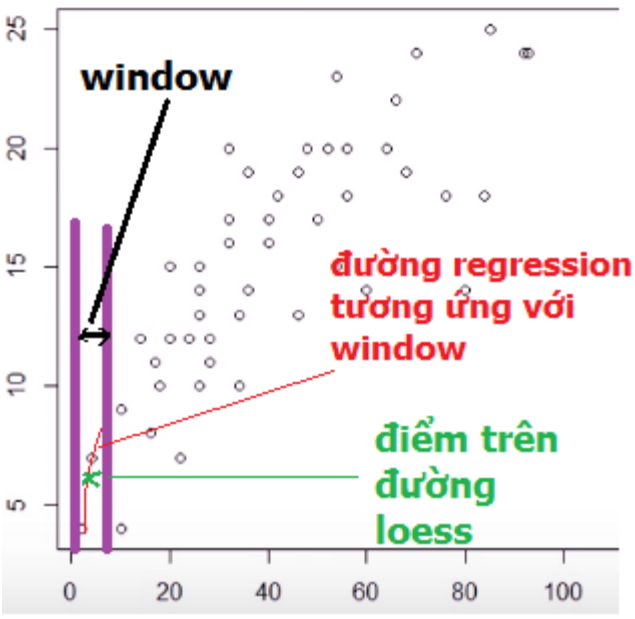
##### II.2.1.2: Loess: "locally-weighted scatterplot smoothing" [1]

Dùng local regression để tạo 1 đường smooth.

Ý tưởng: có 1 'window' chạy dọc hết dữ liệu, window càng rộng thì đường smooth càng mượt.

Với các dữ liệu nằm trong window đó, ta tạo 1 đường regression (đường thẳng hoặc đường cong). Chú ý là ta sẽ đánh trọng số, các điểm ở gần trung tâm sẽ có trọng số cao hơn các điểm ở ngoài rìa window => đường regression bị phụ thuộc vào các điểm trung tâm hơn. Trọng số này sẽ liên tục được tính lại, cập nhật.

Sau các bước trên, ta thu được đường regression, điểm chính giữa của đường regression đó sẽ là điểm trên đường loess ta cần vẽ.




Cụ thể hơn về Loess, xem thêm ở tài liệu tham khảo [1]

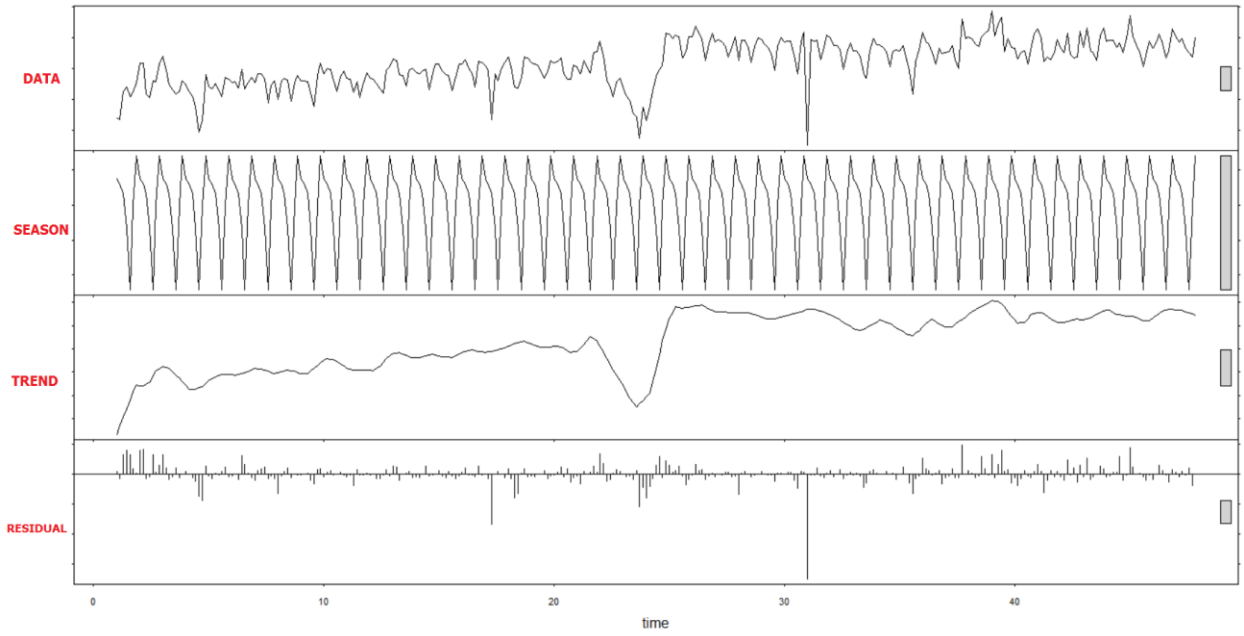
### II.2.2. So sánh 2 hàm stl() và decompose()

| | Decompose() | Stl() |
|---|---|---|
| Season | - Xem season lặp lại từ năm này qua năm khác | - Season có thể thay đổi theo timeseries |
| Mất dữ liệu | - Vài giá trị đầu và cuối không tính toán được (do dùng Moving Average) | - Không bị mất |
| Ảnh hưởng của outliers | - Nhiều hơn | - Ít hơn (do có trọng số) |
| Trading day[4] *(chu kì thị trường chứng khoán)* | - NA | - Không tự động xử lý được[3] |

### II.2.3. Các bước tách data

Trong giải thuật detect Anomaly này, tác giả dùng hàm stl() để tách data thành 3 phần: Trend, season, residual : (residual là phần còn lại của data sau khi tách trend và season)

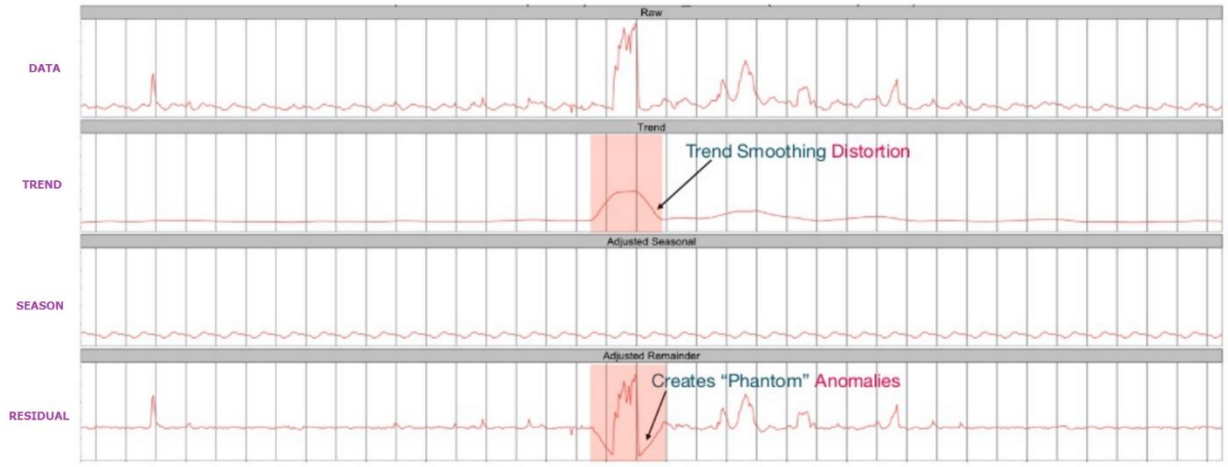
Hàm **stl**() trả về như sau:




### II.3. Tính lại residual

Thông thường thì **Residual = Data - trend - season** (Công thức này áp dụng với Additive Mode. Trong trường hợp Multiplicative Model, ta có thể chuyển về dạng này bằng cách lấy log của 2 vế, khi đó log(data)= log(trend) + log(season) + log(residual), rồi áp dụng công thức trên).

Với cách làm truyền thống, ta dễ bị rơi vào trường hợp sau:




Trend thường có xu hướng "mượt", nó tạo 1 đường lên khá đều (xem hình trên), lúc đó, Residual sẽ xuất hiện phần lõm, tạo ra điểm bất thường không mong đợi.

Để khắc phục vấn đề này, ta có công thức mới để tính Residual:

**Residual = Data – trend – median**[5]


Vấn đề được giải quyết ! Tác giả sử dụng công thức trên để tính Residual trong thuật toán.

### II.4. Chạy S-H ESD test trên Residual

H-S ESD là 1 biến thể của ESD.

Ta sẽ đi nghiên cứu ESD trước

**ESD test**[6] **(Extreme Studentized Deviate test)** được dùng để phát hiện điểm bất thường trong dữ liệu 1 biến. ESD là 1 cải tiến của **Grubbs test** và **Tietjen-Moore test.**

**Grubbs** test và **Tietjen-Moore** test yêu cầu nhập đúng số outliers k, rồi chỉ test với k đó. Trong khi **ESD** sẽ nhận một số $\mathbf{k_{max}}$, và kiểm tra từ 1 cho đến $\mathbf{k_{max}}$ để tìm ra số outlier trong data.

- Giả thiết:

**H0:** Không có outliers trong data

**Ha:** Có r outlier trong data

- Test statistic: $R_i = \frac{\max_i |x_i - \bar{x}|}{s}$

(với $\overline{x}$ và $s$ là trung bình cộng và độ lệch chuẩn của dữ liệu)

- Critical values:

$$\lambda_i = \frac{(n-i)\, t_{p,\, n-i-1}}{\sqrt{(n-i-1+t^2_{p,\, n-i-1})(n-i+1)}} \qquad i = 1, 2, \ldots, r$$

Với $t_{p,v}$ là 100p% từ t-distribution với bậc tự do v

*Chi tiết hơn, xem ở mục [6]*

**Số outliers được trả về là số i lớn nhất, thỏa $R_i > \lambda_i$.**

+ Ở đây, tác giả dùng **S-H ESD (**Seasonal-Hybrid Extreme Studentized Deviate) test. S-H ESD thực ra là ESD, nhưng được kết hợp với season. Thuật toán chia timeseries thành nhiều chu kì (period), sau đó áp dụng ESD vào từng chu kì.

Bản chất, giải thuật vẫn dùng ESD.

+ Trong khi hiện thực ESD, tác giả tiến hành robust statistics bằng cách tính Độ lệch chuẩn theo **MAD**.

MAD = $\mathrm{MAD} = \mathrm{median}(\ |X_i - \mathrm{median}(X)|\ )$

MAD giúp cải thiện phát hiện điểm bất thường trong data hơn công thức thông thường. Cách tính độ lệch chuẩn thông thường, sẽ có bình phương. Việc này khiến cho tác động của outliers lớn, kéo theo độ lệch chuẩn lớn, dẫn đến khó phát hiện outliers, gây ảnh hưởng tới phát hiện điểm bất thường.[7]

Từ đó, Ta tính **Độ lệch chuẩn** theo **MAD** là

$\sigma = 1.4826\ MAD$

(1.4826 ở đây là do giả thiết normal distribution, xem thêm [7])

## III. Hướng dẫn dùng package

### III.1. Cài đặt

```
install.packages("devtools")
devtools::install_github("twitter/AnomalyDetection")
library(AnomalyDetection) # attach the library
```

### III.2. Giới thiệu các hàm package cung cấp

Package cung cấp 2 hàm:

- **`AnomalyDetectionTs()`** : Áp dụng cho dữ liệu là 1 **data.frame**, gồm 2 cột. Cột 1 là thời gian, cột 2 là dữ liệu.

  **Trả về:**

  + Danh sách các điểm bất thường

  + plot 1 đồ thị biểu diễn timeseries, trong đó các điểm bất thường được khoanh tròn (màu xanh)

- **`AnomalyDetectionVec()`** : Tương tự như hàm trên, nhưng áp dụng cho data (vector) chỉ có cột 2.

## III.3. Hàm `AnomalyDetectionTs()`

Với hàm **`AnomalyDetectionVec()`** thì các tham số cũng tương tự, nên ta chỉ xét đại diện. Một số lưu ý riêng về AnomalyDetectionVec() sẽ được đề cập sau.

### III.3.1. Tham số của `AnomalyDetectionTs()`:

```
AnomalyDetectionTs <- function(x, max_anoms = 0.10, direction = 'pos',
                               alpha = 0.05, only_last = NULL, threshold = 'None',
                               e_value = FALSE, longterm = FALSE, piecewise_median_period_weeks = 2, plot = FALSE,
                               y_log = FALSE, xlabel = '', ylabel = 'count',
                               title = NULL, verbose=FALSE, na.rm = FALSE){
```

**Giải thích các tham số:**

**x:** data frame input (2 cột)

**max_anoms**: Tối đa số các điểm bất thường cần kiểm tra, tính theo % data

**direction**: Hướng phát hiện điểm bất thường. có 3 lựa chọn là 'pos' | 'neg' | 'both', tương ứng với "phát hiện điểm bất thường phía trên (giá trị bất thường cao hơn các điểm khác)", "phía dưới", "cả 2 phía" của data.

**alpha**: ngưỡng test (level of statistical significance)

**only_last:** chỉ kiểm tra trên chu kì (ngày, giờ) cuối cùng của timeseries.

**threshold:** Lọc lại các điểm bất thường. Chỉ nhận các điểm bất thường vượt ngưỡng này. Có 4 giá trị là 'None' | 'med_max' | 'p95' | 'p99'.

+ None: Không lọc lại.

+ med_max: Tính **periodic_maxs** = max của từng ngày (trong trường hợp timeseries tính theo giờ, hoặc phút, thì mỗi ngày sẽ gồm nhiều observations, ta lấy max của mỗi ngày. Còn khi dữ liệu biểu diễn theo ngày, thì max của từng ngày chính là nó), sau đó lấy median của các max vừa tìm được.

+ p99: lấy quantile 0.99 của **periodic_maxs** (ở trên), tức là quantile(periodic_maxs, 0.99)

**+** p95**:** tương tự, quantile(periodic_maxs, 0.95)

**evalue:** expected value, thêm cột expected value vào kết quả trả về.

**longterm:** Nếu TRUE, data sẽ được chia thành nhiều chu kì nhỏ. Mục đích giúp cải thiện thuật toán. Đây là một tham số quan trọng, ta sẽ đi sâu vào tham số này ở mục sau.

**piecewise_median_period_weeks:** Điều chỉnh số observation trong một chu kì, mặc định là 2. Sẽ được nói rõ hơn cùng với **longterm**.

**Plot:** Vẽ hình hiển thị với các điểm bất thường, nên chọn TRUE

**y-log:** Với 1 số timeseries có outliers quá lớn, sẽ khó để plot hình. Khi đó ta chọn y-log bằng TRUE, để lấy log(data)

**vesbose:** In ra từng bước lúc chạy thuật toán.

...

### III.3.2. Cách chọn tham số longterm và piecewise_median_period_weeks

Mấu chốt của việc dùng hàm này, là ở 2 tham số trên.

**Đầu tiên, ta xét tham số longterm.**

- **longterm = F**, tất cả data được xem thành 1 chu kì.

  Giải thuật sẽ tiến hành ESD test trên nguyên tập data.

- **longterm = T**, data được chia thành nhiều tập (chunks) nhỏ,

Mỗi tập gồm có (**piecewise_median_period_weeks * period + 1)** observations. Lúc nhận input, hàm sẽ kiểm tra cột 1 của dữ liệu nhập vào, để xác định timeseries được biểu diễn theo loại gì (Date, hour,...). Biến **period (chu kì)** ở đây được gán như sau:

+ Timeseries tính theo phút: **period** = 1440 (gom theo ngày, 1440 là số phút của 1 ngày. Chu kì sẽ được tính là 1440 observations)

+ Timeseries tính theo giờ: **period** = 24 (gom theo ngày)

+ Timeseries tính theo date: **period** = 7 (gom theo tuần)

Như vậy, với dữ liệu tính theo ngày (Date), ta có period = 7.

Mỗi tập sẽ có (**7* piecewise_median_period_weeks + 1)** ngày. (mặc định là 7*2+1 = 15 ngày)

Khi đó data sẽ bao gồm số tập (chunks) là:

**"Tổng số observations" / "số observation mỗi tập"** tập (chunks).

Sau khi chia data thành nhiều tập (ta tạm gọi là các **sub_data**), ta tiến hành phân tách time series và ESD test trên từng tập đó. Cụ thể:

Với mỗi tập, ta chạy hàm **detect_anoms().** Có thể hiểu đơn giản, hàm này giúp ta chạy kiểm tra từng tập sub_data được chia ra, còn hàm **AnomalyDetectionTs()** sẽ tổng hợp kết quả.

Hàm **detect_anoms()** nhận 2 tham số quan trọng là **sub_data**, và **period**.

+ **sub_data:** là 1 tập được chia ra từ data gốc

+ **period**: được truyền vào tham số **frequency** trong hàm **stl** (số phần tử trong mỗi season của time series)

**detect_anoms()** sẽ phân tách time series, sau đó tiến hành ESD test trên từng tập sub_data.

Kết quả sẽ được trả về cho hàm cha **AnomalyDetectionTs()** tổng hợp.

Đây là điểm khác biệt khi set giá trị **longterm** = T. Việc này giúp cải thiện độ hiệu quả của thuật toán với data lớn hơn 30 observations.

### III.3.3. Hàm `AnomalyDetectionVec()`

```
AnomalyDetectionVec = function(x, max_anoms=0.10, direction='pos',
                               alpha=0.05, period=NULL, only_last=F,
                               threshold='None', e_value=F, longterm_period=NULL,
                               plot=F, y_log=F, xlabel='', ylabel='count',
                               title=NULL, verbose=FALSE){
```

Tương tự như hàm **`AnomalyDetectionTs()`**, ta chỉ cần lưu ý vài điểm sau:

- Hàm nhận vào data chỉ gồm cột giá trị, không có cột thời gian

- Do không có cột thời gian, nên ta phải tự điều chỉnh các tham số chu kì, bao gồm **period**, **longterm_period:**

+ **period**: Tương đương với **period** của hàm **`AnomalyDetectionTs()`**. Khi data được biễn diễn theo Date, thì **period** của **`AnomalyDetectionTs()`** là 7. Ta ***chỉ có thể*** điều chỉnh nó khi dùng hàm **`AnomalyDetectionVec()`**.Việc dùng hàm **`AnomalyDetectionVec()`** mặc dù khiến ta điều chỉnh tham số nhiều hơn, những cũng giúp ta linh hoạt hơn.

+ **longterm_period**: Tương ứng với "**số observation mỗi tập (chunk)**" ở hàm `AnomalyDetectionTs()`, tức là **longterm_period = piecewise_median_period_weeks * period + 1.**

Ta không cần thiết lập ***longterm****=TRUE* như trước, thay vào đó, ta chỉ định giá trị cho **longterm_period** để kích hoạt chức năng chia data thành nhiều chunks.

Ví dụ: Hai đoạn code sau đây tương đương nhau:

```
res_Ts = AnomalyDetectionTs(data, max_anoms=0.1, direction='both', plot=TRUE, longterm = T)
```

```
res_Vec = AnomalyDetectionVec(data[,2], max_anoms=0.1, direction='both', plot=TRUE, period = 7, longterm_period = 15)
```

## IV. Chạy thử data mẫu

Kết quả chạy với file "**test_data.csv**":

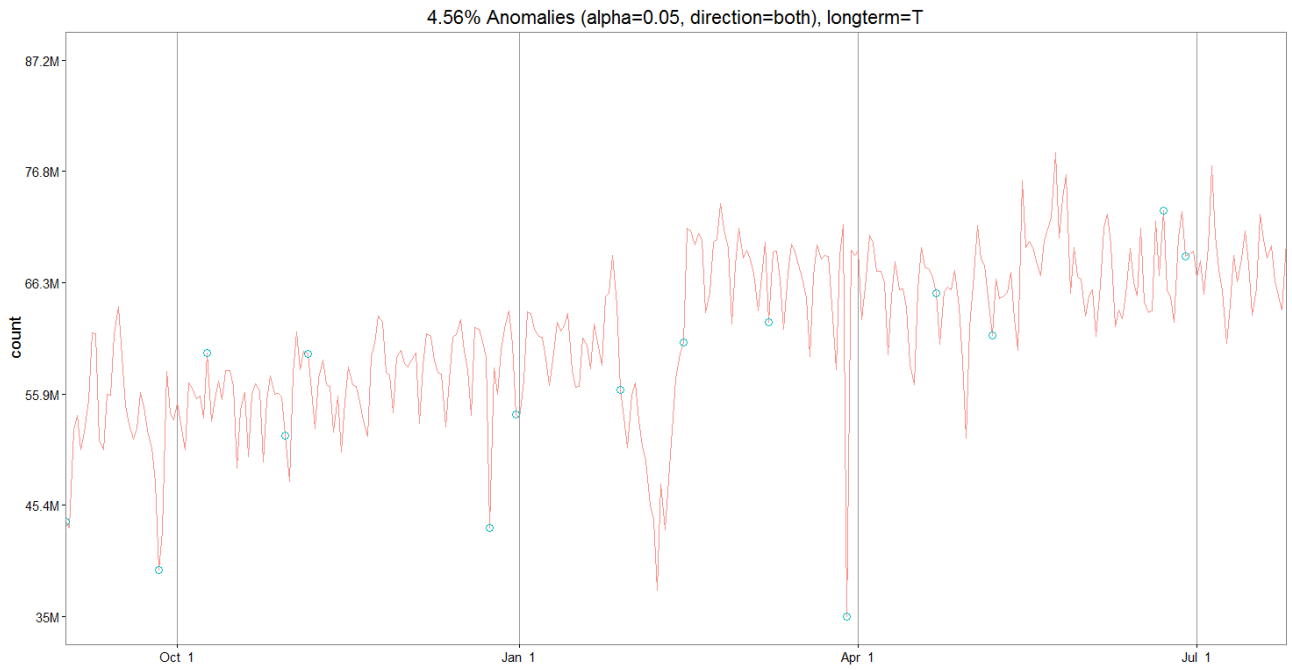
```
#install packages
install.packages("devtools")
devtools::install_github("twitter/AnomalyDetection")

library(AnomalyDetection)

#read the input data
data = read.csv("test_data.csv", sep="\t")

#convert the first col to Posix format
data[[1]] = as.POSIXlt(as.Date(x = data[,1],  "%m/%d/%Y"))

#use the package
res = AnomalyDetectionTs(data, max_anoms=0.1, direction='both',
plot=TRUE, longterm = T)
            # here we set 'longterm = T' and let
            'piecewise_median_period_weeks = 2' as default. By
            this way, we divided the data into multiple-chunks.
            It seems improve the result.
res$plot
res$anoms #return a data.frame() contain anomaly points.
```

- Kết quả:

# res$anoms : Trả về Bảng các điểm bất thường

| STT | timestamp | Anom values |
|---|---|---|
| 1 | 9/1/2015 | 43860155 |
| 2 | 9/26/2015 | 39336673 |
| 3 | 10/9/2015 | 59716553 |
| 4 | 10/30/2015 | 52012722 |
| 5 | 11/5/2015 | 59637770 |
| 6 | 12/24/2015 | 43316175 |
| 7 | 12/31/2015 | 53980630 |
| 8 | 1/28/2016 | 56304605 |
| 9 | 2/14/2016 | 60728844 |
| 10 | 3/8/2016 | 62602372 |
| 11 | 3/29/2016 | 34969512 |
| 12 | 4/22/2016 | 65318529 |
| 13 | 5/7/2016 | 61357086 |
| 14 | 6/22/2016 | 73094568 |
| 15 | 6/28/2016 | 68848608 |

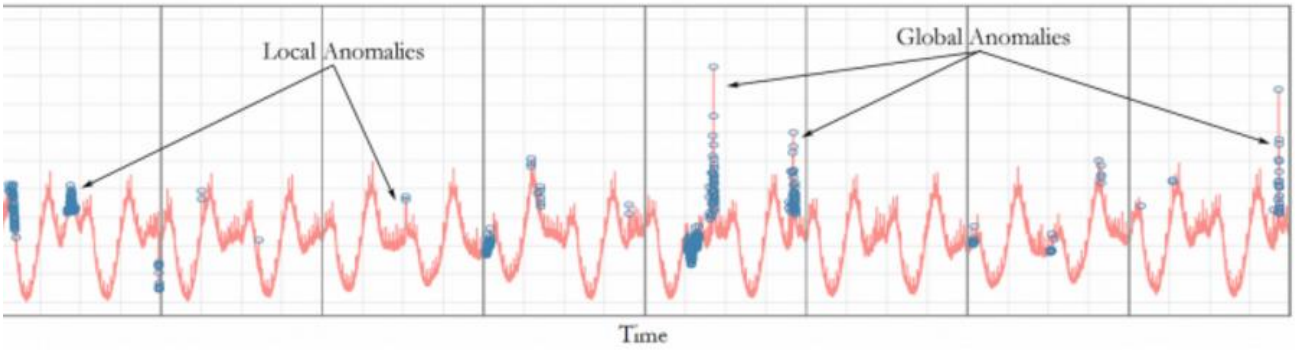
# res$plot: Biểu đồ. Các điểm bất thường được bôi tròn (xanh dương)



## V. Nhận xét:

### V.1. Điểm mạnh

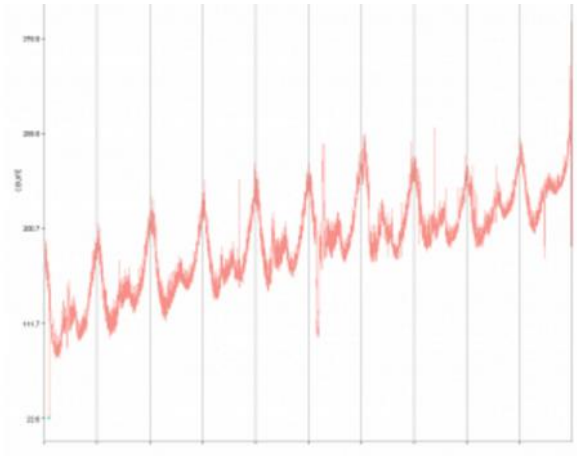
- Phát hiện được cả Local anomaly (điểm bất thường lẫn trong season)




- Tương đối hiệu quả với nhiều trường hợp

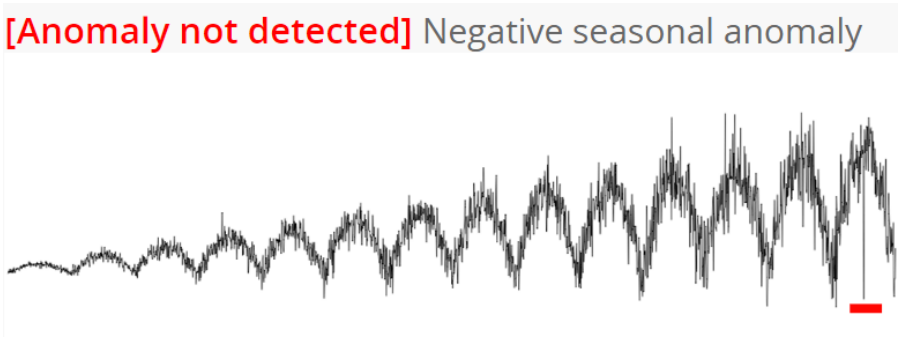
### V.2. Điểm yếu:

- Phải tự gỡ tính linear trong data ra, nếu không thuật toán sẽ rất tệ:



*SHESD identfies only a single outlier due to linear trend*

- Không phát hiện được season âm:




**Xem thêm ở tài liệu tham khảo [9].**

**Tài liệu tham khảo**

[1] http://align-alytics.com/seasonal-decomposition-of-time-series-by-loessan-experiment/
[2] http://cs.wellesley.edu/~cs315/Papers/stl%20statistical%20model.pdf
[3] https://www.otexts.org/fpp/6/5
[4] https://en.wikipedia.org/wiki/Trading_day
[5] http://www.slideshare.net/arunkejariwal/statistical-learning-based-anomaly-detection-twitter
[6] http://www.itl.nist.gov/div898/handbook/eda/section3/eda35h3.htm
[7] https://en.wikipedia.org/wiki/Median_absolute_deviation
[8] https://github.com/twitter/AnomalyDetection
[9] https://anomaly.io/anomaly-detection-twitter-r/
[10] http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc42.htm