Vignette Draft

Names

2022-12-01

```
# load any other packages and read data here
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✔ ggplot2 3.3.6      ✔ purrr   0.3.5
## ✔ tibble  3.1.8      ✔ dplyr   1.0.10
## ✔ tidyr   1.2.0      ✔ stringr 1.4.0
## ✔ readr   2.1.2      ✔ forcats 0.5.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(knitr)
library(tibbletime)
```

```
##
## Attaching package: 'tibbletime'
##
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(anomalize)
```

```
## ══ Use anomalize to improve your Forecasts by 50%! ═════════════════════════════
## Business Science offers a 1-hour course - Lab #18: Time Series Anomaly Detection!
## </> Learn more at: https://university.business-science.io/p/learning-labs-pro </>
```

```
library(timetk)
library(zoo) #dates
```

```
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(Rbeast) #used for change-point detection
library(changepoint) #used for change-point detection
```

```
## Successfully loaded changepoint package version 2.2.4
##  See NEWS for details of changes.
```

```
library(changepoint.np) #used for change-point nonparemetric
```

Executive summary

Time series anomaly detection and change-point on the univariate (potentially multivariate case) for time series economic data from LA concerning unemployment.

Task Question 1:

TEXTHERE

Task Question 2:

The primary interest in task 2 was to find significant periods of changes, whereas task 1 focused on anomaly detection of certain points, our interest lies in finding significant increments of time where there may have been upwards or downwards shifts in unemployment rates.

Data description

```
unemployment <- read_csv(here::here('data/processed_data.csv'))
```

```
## Rows: 385 Columns: 23
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): DATE
## dbl (22): unemploy_rate_la, avg_price_electr_kwh_La, avg_price_gasolone_la, ...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
unemployment
```

```
## # A tibble: 385 × 23
##    DATE   unemp…¹ avg_p…² avg_p…³ civil…⁴ unemp…⁵ new_p…⁶ home_…⁷ allem…⁸ allem…⁹
##    <chr>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 1990…      5.6   0.108   0.988     0.2    -4.3   4649.     0.4     0.1    -1.2
##  2 1990…      5.4   0.108   1.01      0.1    -3.8   3628.     0.2    -0.1    -1.1
##  3 1990…      5.5   0.108   1.03     -0.4     1     3833.    -0.1    -0.2    -3.7
##  4 1990…      5.4   0.107   1.08      0.4    -0.9   3466.    -1.1    -0.2    -1.3
##  5 1990…      5.4   0.108   1.10      0      -0.5   3497.    -0.3    -0.3    -0.9
##  6 1990…      6.2   0.108   1.13      0.5    15.2   2796.    -0.9    -0.4    -1.2
##  7 1990…      6.2   0.108   1.28     -0.4     0.2   2466.    -0.6    -0.4    -1.8
##  8 1990…      6.3   0.108   1.35     -1.4     0.4   2357.    -0.9    -0.4    -0.4
##  9 1990…      6.2   0.109   1.42     -0.1    -2.3   2173.    -0.6    -0.1    -1.5
## 10 1990…      6.5   0.11    1.42     -0.5     3.7   2267.    -1.2    -0.2    -1.2
## # … with 375 more rows, 13 more variables: allemployee_manu_la_pch <dbl>,
## #   allemployee_finan_la_pch <dbl>, allemployee_leisure_la_pch <dbl>,
## #   govn_social_insu_pch <dbl>, compen_employee_wage_pch <dbl>,
## #   real_disp_inc_per_capital_pch <dbl>, bbk_real_gdp <dbl>,
## #   pers_consum_expen_pch <dbl>, pers_saving_rate_us <dbl>,
## #   pers_current_tax_chg <dbl>, govn_social_ben_toperson_pch <dbl>,
## #   federal_fund_eff_rate_us <dbl>, X30_year_fixed_mortgage_us <dbl>, and …
```

Above we can see that the data collected is essentially a wrap-up on the 1st of a given month from the years 1900 to 2021 recorded from the city of Los Angeles. These variables encapsulate various city markers that range from unemployment to government benefits. Notably, belows are some interesting variables of interest:

- The date (Year-Month-Day) of each observation/recording
- The unemployment rate
- Average price of electricity
- Average price of gasoline

Since we are only concerned with the time-series aspect of unemployment, we are only focusing on the date and the unemployment rate of each month, therefore we will not show much interest in the other variables, yet.

Methodology Descriptions

Anomaly Detection

In this vignette, you'll learn how to conduct anomaly detection for a single time-series data.

What is Anomaly Detection?

A time series is the sequential set of values tracked over a time duration. An anomaly is something that happens that was unexpected or was caused by an abnormal event. (more contents to go….)

Set up

We are going to work with "anomalize" and "timetk" packages in R.

```
# import dataset
processed_data <- read.csv(here::here("data/processed_data.csv"))

# view dataset
head(processed_data, 5)
```

```
##       DATE unemploy_rate_la avg_price_electr_kwh_La avg_price_gasolone_la
## 1 1990/2/1              5.6                   0.108                 0.988
## 2 1990/3/1              5.4                   0.108                 1.014
## 3 1990/4/1              5.5                   0.108                 1.030
## 4 1990/5/1              5.4                   0.107                 1.080
## 5 1990/6/1              5.4                   0.108                 1.103
##   civilian_labor_force_la_pch unemployed_num_pch
## 1                         0.2               -4.3
## 2                         0.1               -3.8
## 3                        -0.4                1.0
## 4                         0.4               -0.9
## 5                         0.0               -0.5
##   new_private_housing_structure_issue_la home_price_index_la
## 1                               4648.972                 0.4
## 2                               3628.443                 0.2
## 3                               3833.476                -0.1
## 4                               3466.321                -1.1
## 5                               3496.684                -0.3
##   allemployee_nonfarm_la_pch allemployee_constr_la_pch allemployee_manu_la_pch
## 1                        0.1                      -1.2                     0.0
## 2                       -0.1                      -1.1                    -0.4
## 3                       -0.2                      -3.7                    -0.4
## 4                       -0.2                      -1.3                    -0.4
## 5                       -0.3                      -0.9                    -0.7
##   allemployee_finan_la_pch allemployee_leisure_la_pch govn_social_insu_pch
## 1                     -0.1                       -0.4                  0.0
## 2                     -0.7                       -0.1                  1.0
## 3                     -0.2                       -0.6                  0.0
## 4                     -0.8                       -0.1                  0.3
## 5                     -0.3                        0.3                  1.3
##   compen_employee_wage_pch real_disp_inc_per_capital_pch bbk_real_gdp
## 1                      1.2                           0.1    6.1814509
## 2                      0.7                          -0.1    2.9195562
## 3                      0.9                           0.5   -0.5634379
## 4                     -0.3                          -0.2    0.7507924
## 5                      0.8                           0.0    1.1771073
##   pers_consum_expen_pch pers_saving_rate_us pers_current_tax_chg
## 1                  -0.1                 8.6                  8.1
## 2                   0.7                 8.3                  5.2
## 3                   0.4                 8.8                  4.2
## 4                   0.2                 8.7                  0.6
## 5                   0.8                 8.6                  3.9
##   govn_social_ben_toperson_pch federal_fund_eff_rate_us
## 1                         -0.2                 8.237143
## 2                          0.7                 8.276774
## 3                          0.6                 8.255000
## 4                         -0.5                 8.176452
## 5                          1.2                 8.288667
##   X30_year_fixed_mortgage_us
## 1                    3.05710
## 2                    0.69135
## 3                    0.99338
```

```
## 4                        1.03664
## 5                       -2.99213
```

```
# data processing
str(processed_data)
```

```
## 'data.frame':    385 obs. of  23 variables:
##  $ DATE                                  : chr  "1990/2/1" "1990/3/1" "1990/4/1" "199
0/5/1" ...
##  $ unemploy_rate_la                      : num  5.6 5.4 5.5 5.4 5.4 6.2 6.2 6.3 6.2
6.5 ...
##  $ avg_price_electr_kwh_La               : num  0.108 0.108 0.108 0.107 0.108 0.108
0.108 0.108 0.109 0.11 ...
##  $ avg_price_gasolone_la                 : num  0.988 1.014 1.03 1.08 1.103 ...
##  $ civilian_labor_force_la_pch           : num  0.2 0.1 -0.4 0.4 0 0.5 -0.4 -1.4 -0.1
-0.5 ...
##  $ unemployed_num_pch                    : num  -4.3 -3.8 1 -0.9 -0.5 15.2 0.2 0.4 -
2.3 3.7 ...
##  $ new_private_housing_structure_issue_la: num  4649 3628 3833 3466 3497 ...
##  $ home_price_index_la                   : num  0.4 0.2 -0.1 -1.1 -0.3 -0.9 -0.6 -0.9
-0.6 -1.2 ...
##  $ allemployee_nonfarm_la_pch            : num  0.1 -0.1 -0.2 -0.2 -0.3 -0.4 -0.4 -0.
4 -0.1 -0.2 ...
##  $ allemployee_constr_la_pch             : num  -1.2 -1.1 -3.7 -1.3 -0.9 -1.2 -1.8 -
0.4 -1.5 -1.2 ...
##  $ allemployee_manu_la_pch               : num  0 -0.4 -0.4 -0.4 -0.7 0.1 -0.9 -0.8 -
0.4 -0.5 ...
##  $ allemployee_finan_la_pch              : num  -0.1 -0.7 -0.2 -0.8 -0.3 0.7 0.1 0.2
-0.3 -0.5 ...
##  $ allemployee_leisure_la_pch            : num  -0.4 -0.1 -0.6 -0.1 0.3 0.5 -0.3 0 0.
1 0.2 ...
##  $ govn_social_insu_pch                  : num  0 1 0 0.3 1.3 0.7 -0.1 0.8 -0.3 0.3
...
##  $ compen_employee_wage_pch              : num  1.2 0.7 0.9 -0.3 0.8 0.7 -0.4 0.7 -0.
8 0 ...
##  $ real_disp_inc_per_capital_pch         : num  0.1 -0.1 0.5 -0.2 0 0.2 -0.8 -0.2 -0.
9 -0.1 ...
##  $ bbk_real_gdp                          : num  6.181 2.92 -0.563 0.751 1.177 ...
##  $ pers_consum_expen_pch                 : num  -0.1 0.7 0.4 0.2 0.8 0.5 0.7 0.6 0 0
...
##  $ pers_saving_rate_us                   : num  8.6 8.3 8.8 8.7 8.6 8.7 8.1 8.1 7.8
7.9 ...
##  $ pers_current_tax_chg                  : num  8.1 5.2 4.2 0.6 3.9 2.4 -0.3 3.3 -1.8
-0.5 ...
##  $ govn_social_ben_toperson_pch          : num  -0.2 0.7 0.6 -0.5 1.2 -0.7 -0.3 2 -1
0.4 ...
##  $ federal_fund_eff_rate_us              : num  8.24 8.28 8.26 8.18 8.29 ...
##  $ X30_year_fixed_mortgage_us            : num  3.057 0.691 0.993 1.037 -2.992 ...
```

```
# change chr to Date format for 'DATE' column
# and select only the unemployment rate in LA
df <- processed_data %>%
  mutate(DATE, DATE = as.Date.character(DATE)) %>%
  select(DATE, unemploy_rate_la) %>%
  as_tibble(df) # convert df to a tibble

str(df)
```

```
## tibble [385 × 2] (S3: tbl_df/tbl/data.frame)
##  $ DATE            : Date[1:385], format: "1990-02-01" "1990-03-01" ...
##  $ unemploy_rate_la: num [1:385] 5.6 5.4 5.5 5.4 5.4 6.2 6.2 6.3 6.2 6.5 ...
```

Uni-variate Time Series Anomaly Detection

The entire process of Anomaly detection for a time-series take place across 3 steps:

1. Decompose the time-series into the underlying variables: *trend, seasonality, remainder*
2. Create upper and lower thresholds based on certain algorithms
3. Identify the data points which are outside the thresholds as anomalies

```
# using 'anomalize' package
#The R ‘anomalize’ package enables a workflow for detecting anomalies in data. The # mai
n functions are time_decompose(), anomalize(), and time_recompose().

df_anomalized <- df %>%
  time_decompose(unemploy_rate_la, merge = TRUE) %>%
  anomalize(remainder) %>%
  time_recompose()
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

```
## frequency = 12 months
```

```
## trend = 60 months
```

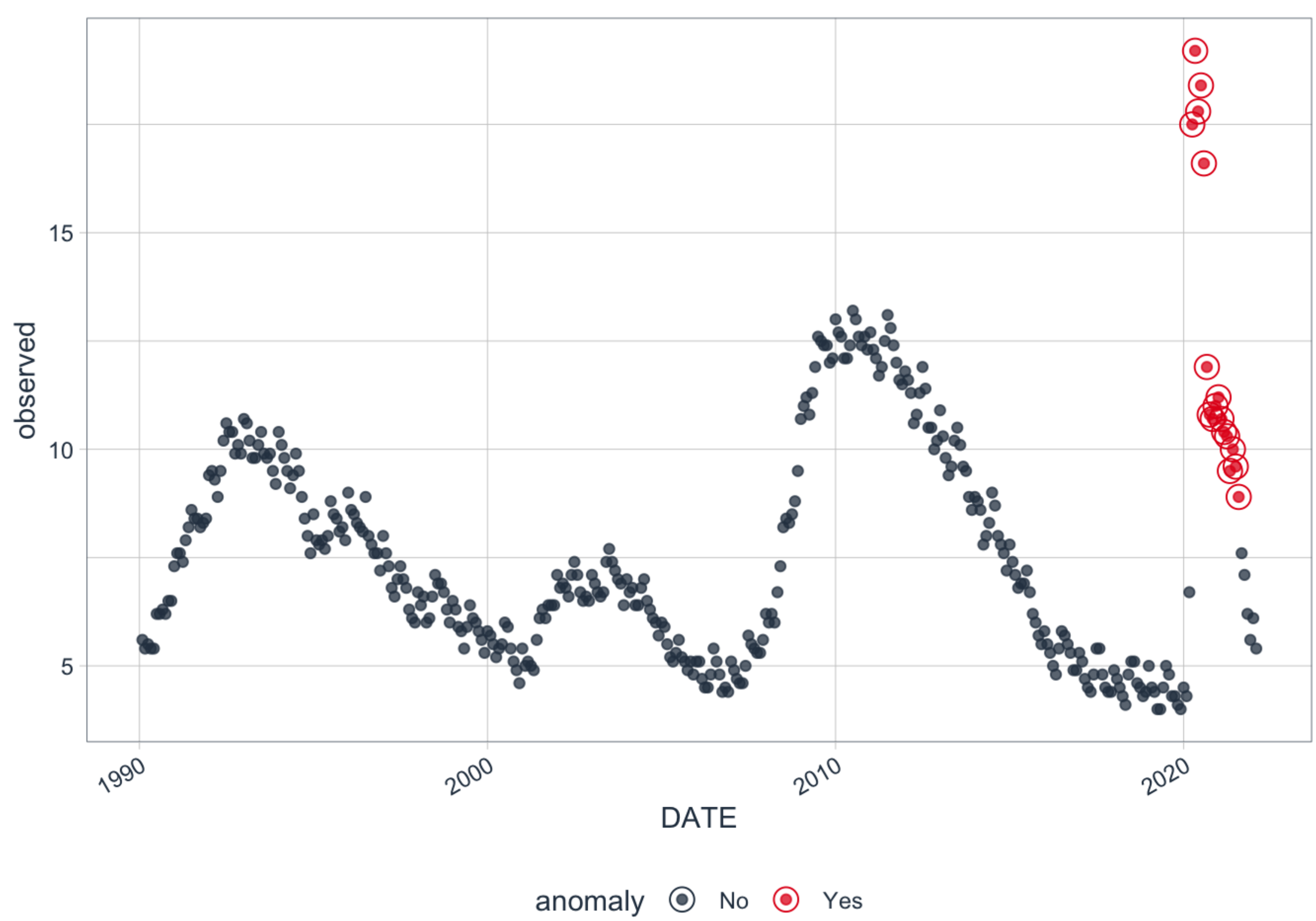
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
df_anomalized %>% glimpse()
```

```
## Rows: 385
## Columns: 11
## Index: DATE
## $ DATE              <date> 1990-02-01, 1990-03-01, 1990-04-01, 1990-05-01, 1990…
## $ unemploy_rate_la <dbl> 5.6, 5.4, 5.5, 5.4, 5.4, 6.2, 6.2, 6.3, 6.2, 6.5, 6.5…
## $ observed          <dbl> 5.6, 5.4, 5.5, 5.4, 5.4, 6.2, 6.2, 6.3, 6.2, 6.5, 6.5…
## $ season            <dbl> 0.01079240, -0.10196977, -0.37829521, -0.32774673, 0.…
## $ trend             <dbl> 5.769614, 5.895768, 6.021921, 6.148075, 6.274228, 6.4…
## $ remainder         <dbl> -0.18040676, -0.39379797, -0.14362593, -0.42032780, -…
## $ remainder_l1      <dbl> -2.314905, -2.314905, -2.314905, -2.314905, -2.314905…
## $ remainder_l2      <dbl> 2.471808, 2.471808, 2.471808, 2.471808, 2.471808, 2.4…
## $ anomaly           <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No",…
## $ recomposed_l1     <dbl> 3.465501, 3.478893, 3.328720, 3.505422, 4.058509, 4.6…
## $ recomposed_l2     <dbl> 8.252215, 8.265606, 8.115434, 8.292136, 8.845223, 9.4…
```

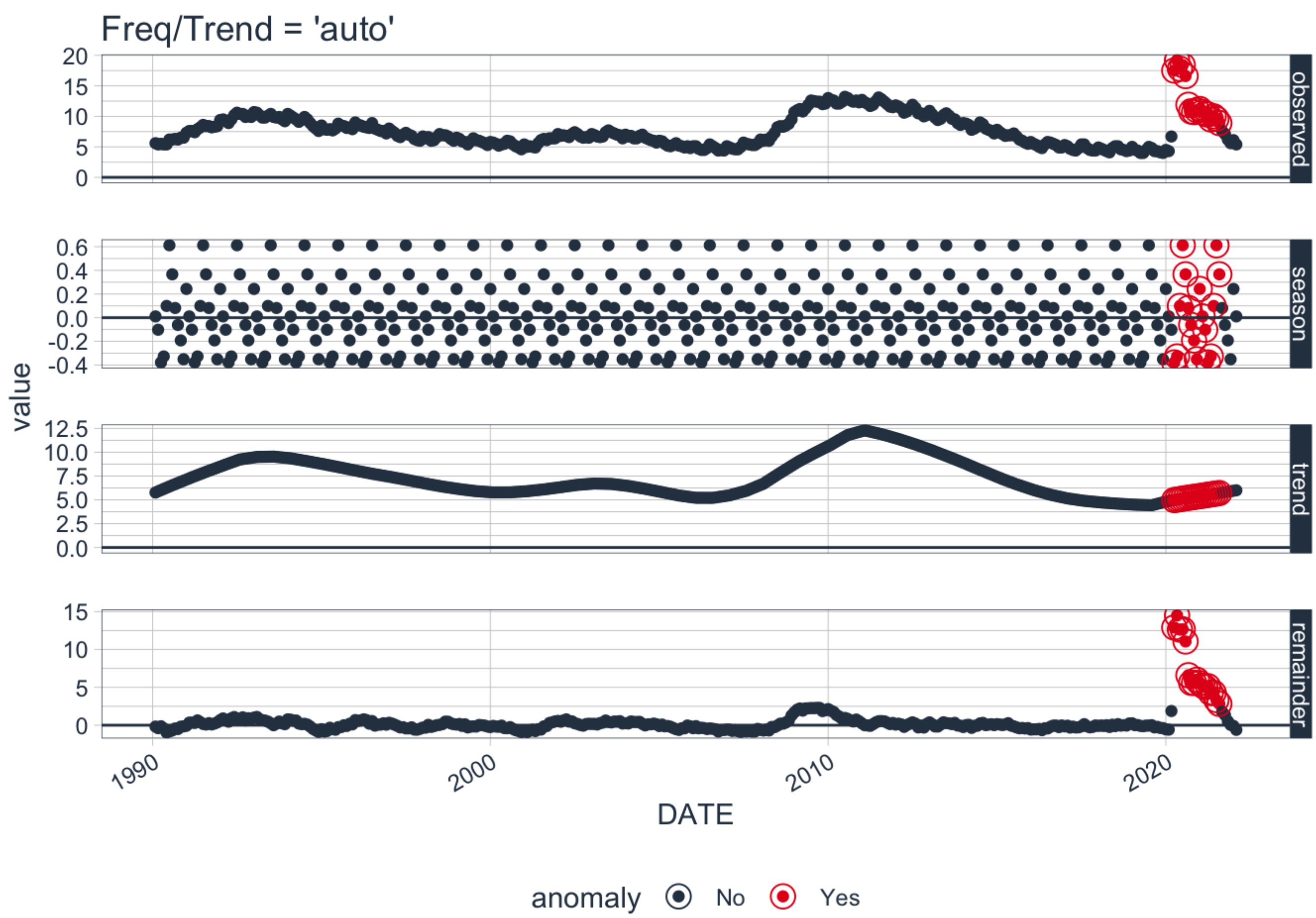
In the output tibble, there is a character column labeling if a time value is an anomalies or not. we can guess that anomalies are determined by “remainder” and the interval formed by “remainder_l1” and “remainder_l2”. Then, we can visualize those anomalies.

```
df_anomalized %>% plot_anomalies(ncol = 4, alpha_dots = 0.75)
```



What is the time period when anomalies are detected? We can observe how those anomalies lie in seasonal, trend, and remainder component.

```
p1 <- df_anomalized %>%
  plot_anomaly_decomposition() +
  ggtitle("Freq/Trend = 'auto'")

p1
```



Then, we can adjust the default trend and seasonality to see what the difference. Let's check what is the default frequency trend for our seasonal decomposition method. This implies that if the scale is 1 day (meaning the difference between each data point is 1 day), then the frequency will be 7 days (or 1 week) and the trend will be around 90 days (or 3 months).

```
get_time_scale_template()
```

```
## # A tibble: 8 × 3
##   time_scale frequency trend
##   <chr>      <chr>     <chr>
## 1 second     1 hour    12 hours
## 2 minute     1 day     14 days
## 3 hour       1 day     1 month
## 4 day        1 week    3 months
## 5 week       1 quarter 1 year
## 6 month      1 year    5 years
## 7 quarter    1 year    10 years
## 8 year       5 years   30 years
```

We can adjust local parameters to see what will happen. You will find the Covid-19 period is so odd upon the whole time period. You can try to exclude years after 2019 to see the difference.

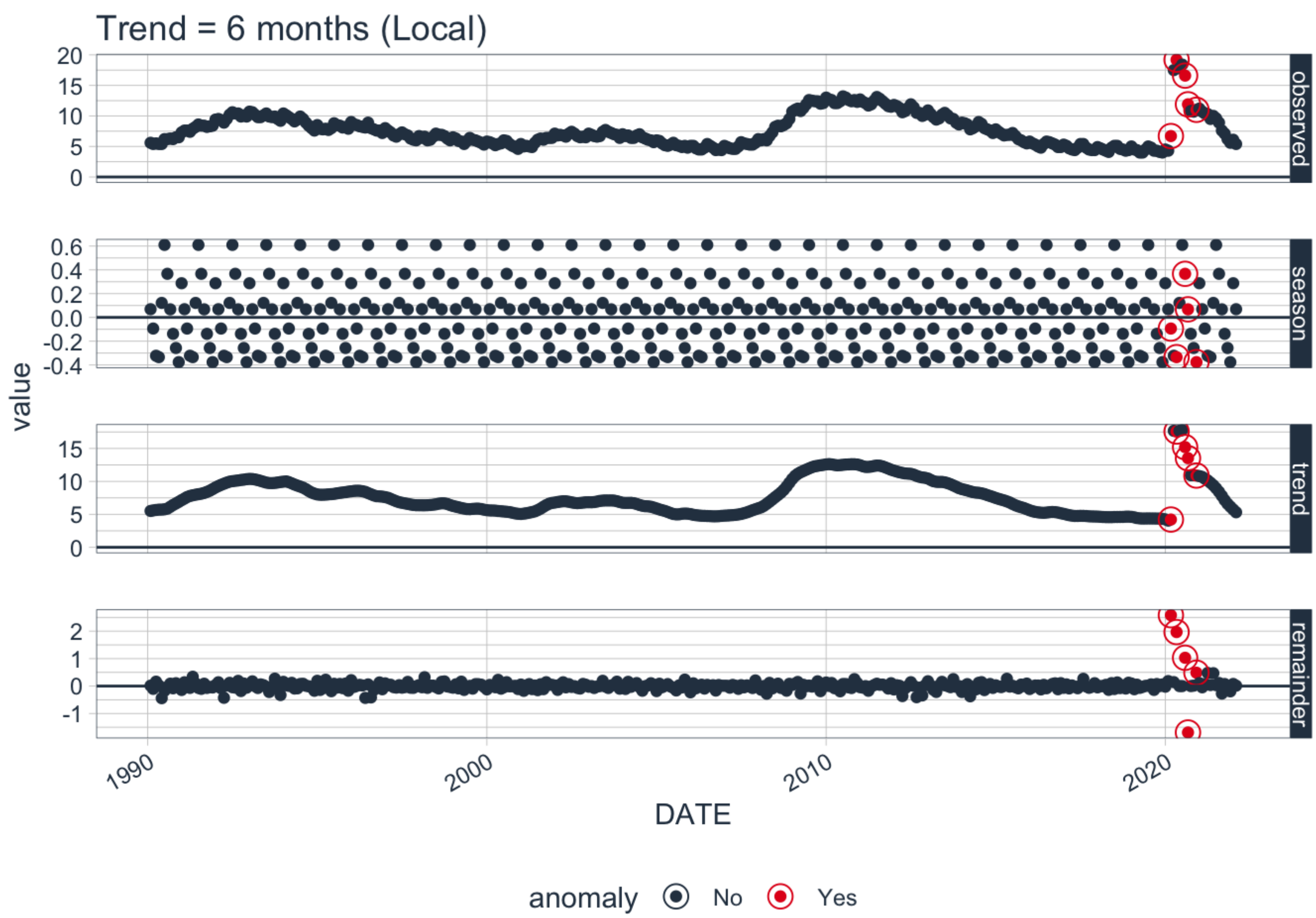
```
p2 <- df %>%
  time_decompose(unemploy_rate_la,
                 frequency = "auto",
                 trend     = "6 months") %>%
  anomalize(remainder) %>%
  plot_anomaly_decomposition() +
  ggtitle("Trend = 6 months (Local)")
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

```
## frequency = 12 months
```

```
## trend = 6 months
```

```
p2
```



Can we detect other economic recession? The answer is Yes. The alpha and max_anoms are the two parameters that control the *anomalize()* function. If we decrease alpha, it increases the bands making it more difficult to be an outlier. The max_anoms parameter is used to control the maximum percentage of data that can be an anomaly.

Please alter two parameters to see what will output.

```
# Adjusting Alpha and Max Anoms

p4 <- df %>%
  time_decompose(unemploy_rate_la) %>%
  anomalize(remainder, alpha = 0.025, max_anoms = 0.2) %>%
  time_recompose() %>%
  plot_anomalies(time_recomposed = TRUE) +
  ggtitle("alpha = 0.05")
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

```
## frequency = 12 months
```

```
## trend = 60 months
```

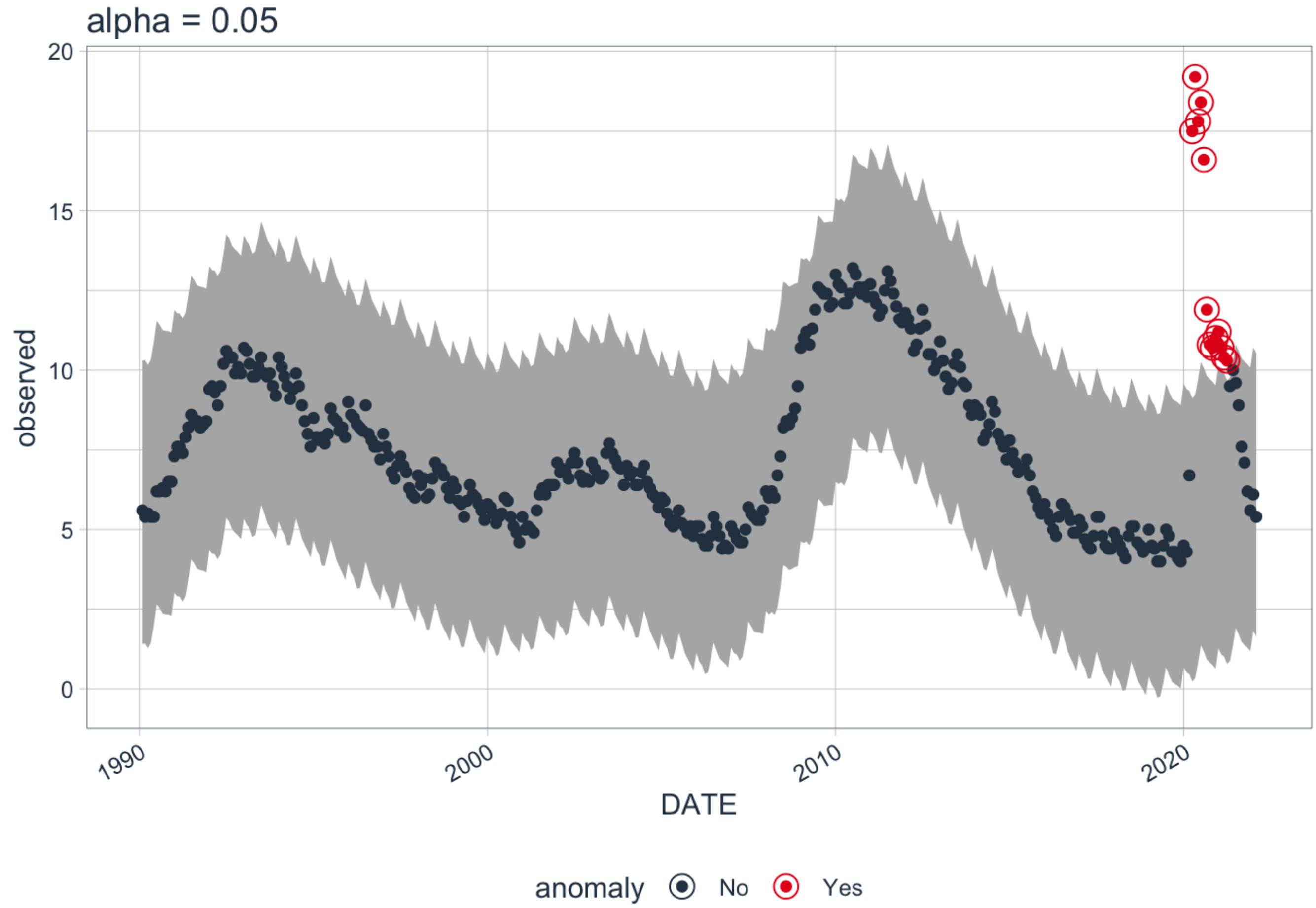
```
#> frequency = 7 days
#> trend = 91 days
p5 <- df %>%
  time_decompose(unemploy_rate_la) %>%
  anomalize(remainder, alpha = 0.6, max_anoms = 0.2) %>%
  time_recompose() %>%
  plot_anomalies(time_recomposed = TRUE) +
  ggtitle("alpha = 0.05")
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

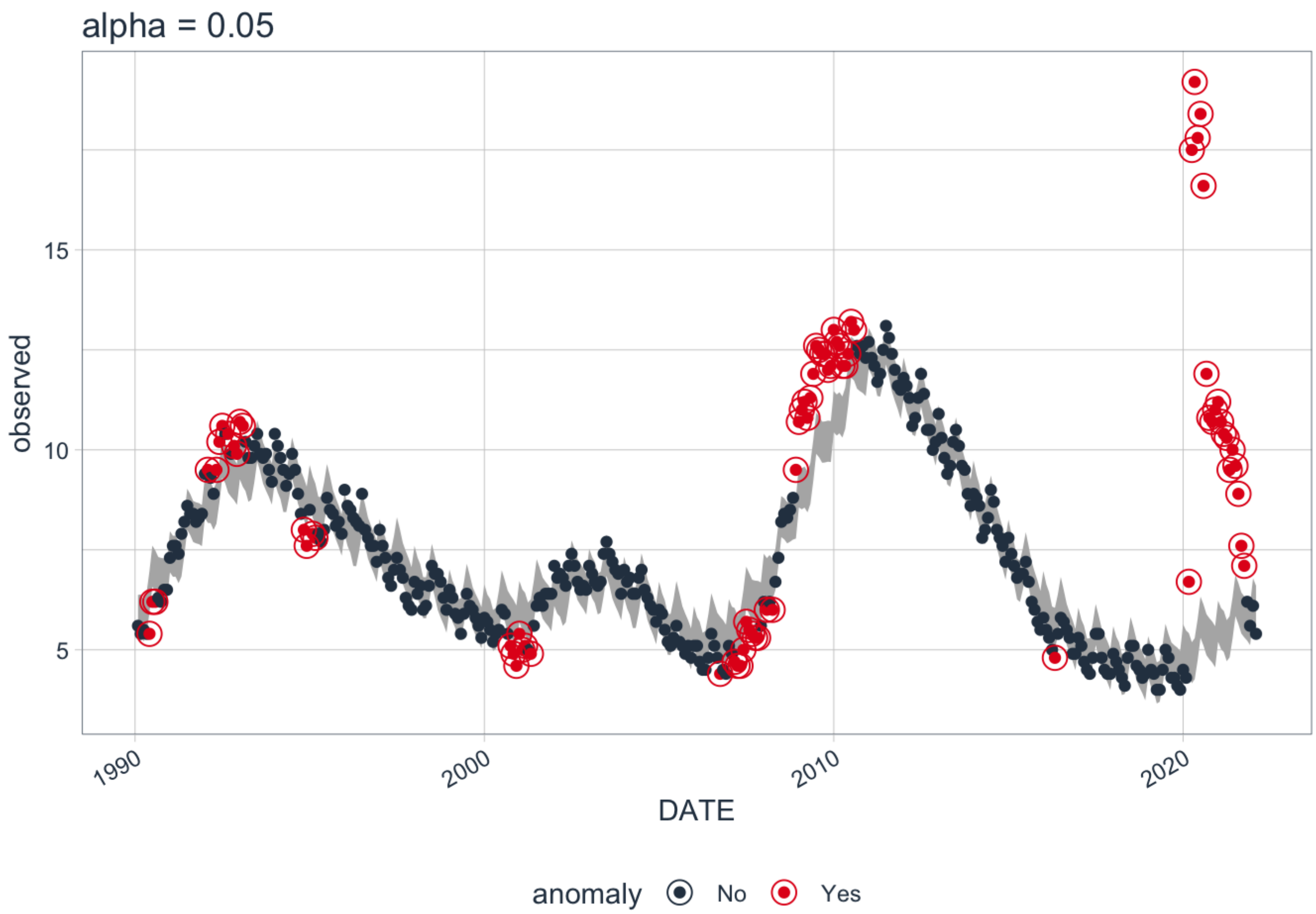
```
## frequency = 12 months
```

```
## trend = 60 months
```

```
#> frequency = 7 days
#> trend = 91 days
p4
```


alpha = 0.05
20
15
10
5
0
observed
1990
2000
2010
2020
DATE
anomaly
No
Yes

```
p5
```




```
p6 <- df %>%
  time_decompose(unemploy_rate_la) %>%
  anomalize(remainder, alpha = 0.3, max_anoms = 0.2) %>%
  time_recompose() %>%
  plot_anomalies(time_recomposed = TRUE) +
  ggtitle("20% Anomalies")
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

```
## frequency = 12 months
```

```
## trend = 60 months
```

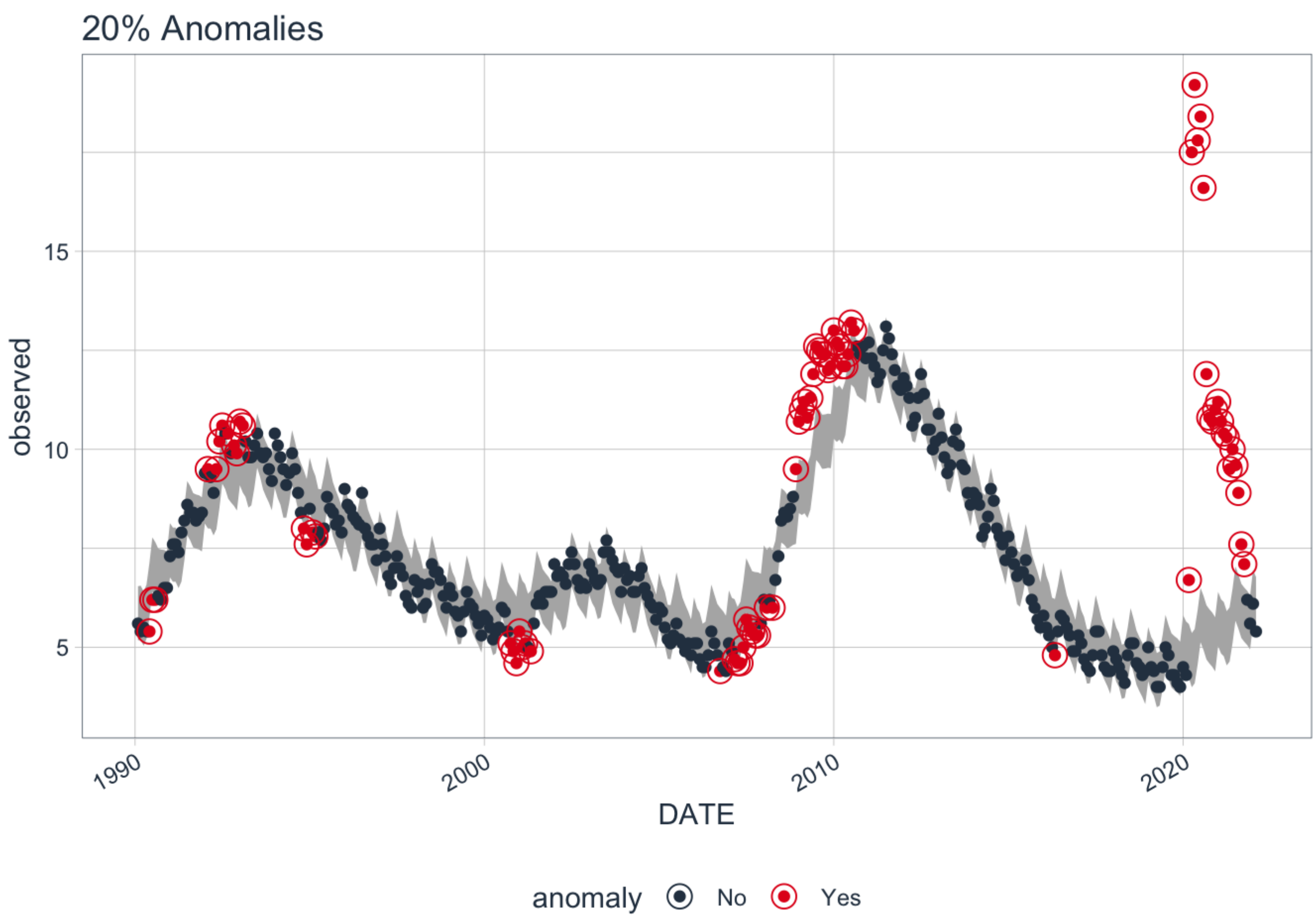
```
#> frequency = 7 days
#> trend = 91 days

p7 <- df %>%
  time_decompose(unemploy_rate_la) %>%
  anomalize(remainder, alpha = 0.3, max_anoms = 0.05) %>%
  time_recompose() %>%
  plot_anomalies(time_recomposed = TRUE) +
  ggtitle("5% Anomalies")
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

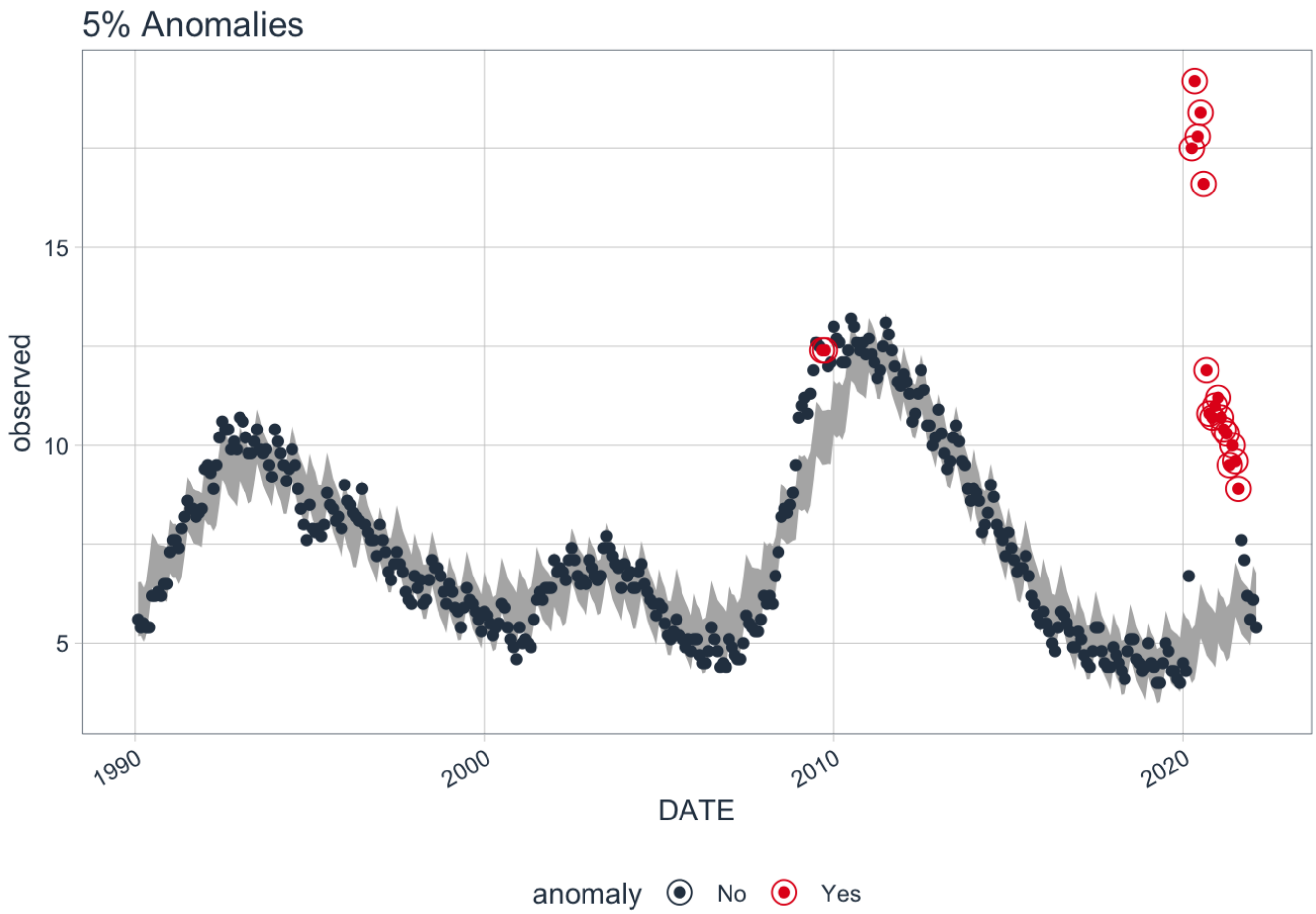
```
## frequency = 12 months
```

```
## trend = 60 months
```

```
#> frequency = 7 days
#> trend = 91 days
p6
```



```
p7
```



Finally, we can extract the anomalous data points.

```
df %>%
  time_decompose(unemploy_rate_la) %>%
  anomalize(remainder) %>%
  time_recompose() %>%
  filter(anomaly == 'Yes')
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

```
## frequency = 12 months
```

```
## trend = 60 months
```

```
## # A time tibble: 17 × 10
## # Index:         DATE
##    DATE       observed  season trend remainder remaind…¹ remai…² anomaly recom…³
##    <date>        <dbl>   <dbl> <dbl>     <dbl>     <dbl>   <dbl> <chr>     <dbl>
##  1 2020-04-01     17.5 -0.378   4.98     12.9      -2.31    2.47 Yes        2.28
##  2 2020-05-01     19.2 -0.328   5.03     14.5      -2.31    2.47 Yes        2.38
##  3 2020-06-01     17.8  0.0992  5.08     12.6      -2.31    2.47 Yes        2.86
##  4 2020-07-01     18.4  0.612   5.12     12.7      -2.31    2.47 Yes        3.42
##  5 2020-08-01     16.6  0.366   5.17     11.1      -2.31    2.47 Yes        3.22
##  6 2020-09-01     11.9  0.0819  5.22      6.60     -2.31    2.47 Yes        2.98
##  7 2020-10-01     10.8 -0.0616  5.26      5.60     -2.31    2.47 Yes        2.88
##  8 2020-11-01     10.7 -0.192   5.31      5.59     -2.31    2.47 Yes        2.80
##  9 2020-12-01     11   -0.351   5.35      6.00     -2.31    2.47 Yes        2.68
## 10 2021-01-01     11.2  0.243   5.39      5.56     -2.31    2.47 Yes        3.32
## 11 2021-02-01     10.7  0.0108  5.44      5.25     -2.31    2.47 Yes        3.13
## 12 2021-03-01     10.4 -0.102   5.48      5.02     -2.31    2.47 Yes        3.07
## 13 2021-04-01     10.3 -0.378   5.53      5.15     -2.31    2.47 Yes        2.84
## 14 2021-05-01      9.5 -0.328   5.57      4.25     -2.31    2.47 Yes        2.93
## 15 2021-06-01     10    0.0992  5.62      4.28     -2.31    2.47 Yes        3.40
## 16 2021-07-01      9.6  0.612   5.66      3.32     -2.31    2.47 Yes        3.96
## 17 2021-08-01      8.9  0.366   5.71      2.82     -2.31    2.47 Yes        3.76
## # … with 1 more variable: recomposed_l2 <dbl>, and abbreviated variable names
## #   ¹remainder_l1, ²remainder_l2, ³recomposed_l1
```

Methods and Techniques used in “anomalize”

Anomaly detection is performed on remainders from a time series analysis that have had removed both:

- Seasonal Components: cyclic pattern usually occurring on a daily cycle for minute or hour data. Here, the cyclic pattern can be interpreted as yearly cycles for monthly data
- Trend Components: Longer term growth that happens over many observations

Therefore, the main goal of step 1 is to generate remainders from a time series. The seasonal decomposition outperforms ARIMA and other machine learning models

We can observe two techniques for seasonal decomposition in the “anomalize” package.

STL: Seasonal Decomposition of Time Series by Loess

Twitter:

```
# STL Decomposition Method
p1 <- df %>%
    time_decompose(unemploy_rate_la,
                   method    = "stl") %>%
    anomalize(remainder) %>%
    plot_anomaly_decomposition() +
    ggtitle("STL Decomposition")
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

```
## frequency = 12 months
```

```
## trend = 60 months
```

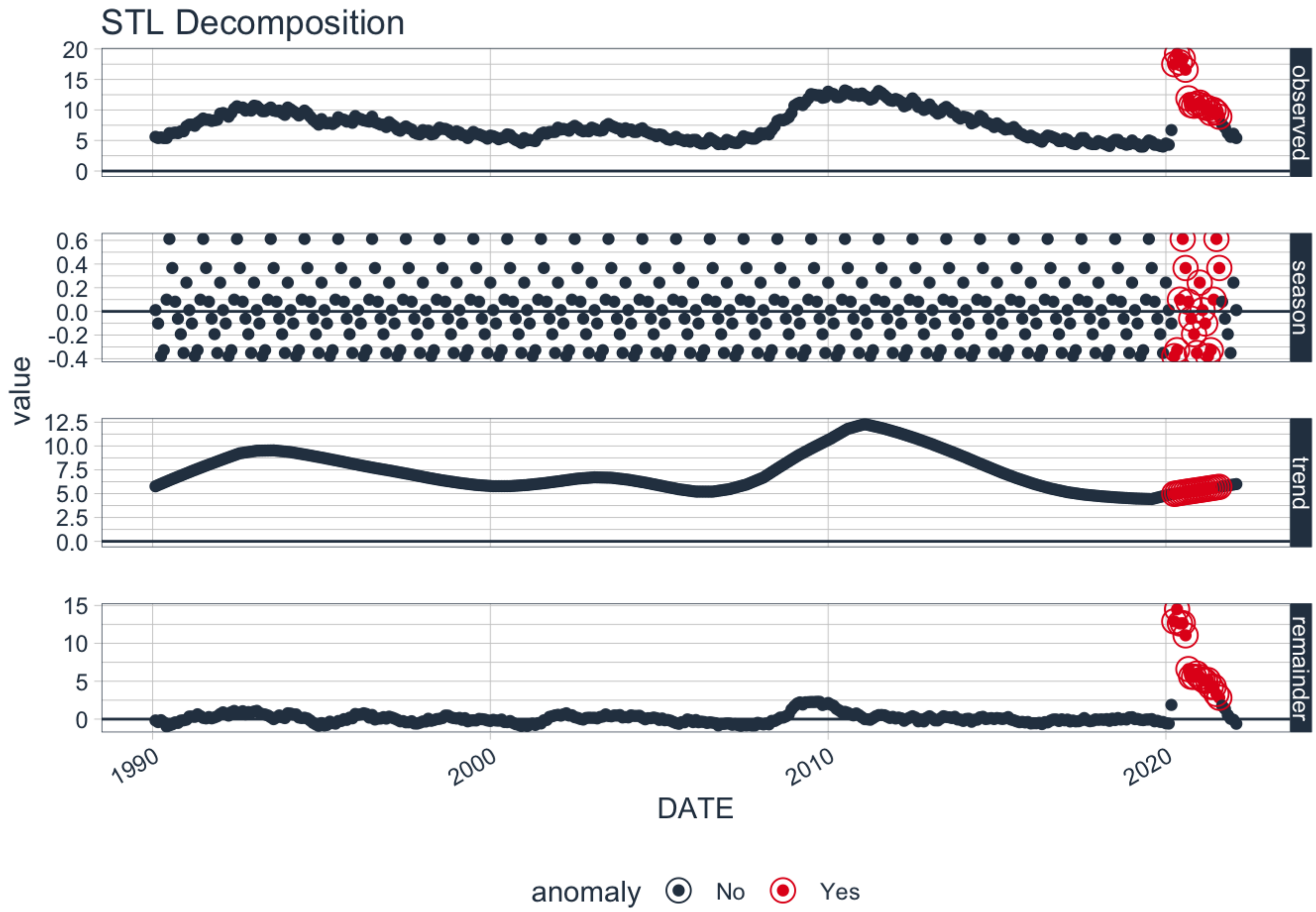
```
#> frequency = 7 days
#> trend = 91 days
#> Registered S3 method overwritten by 'quantmod':
#>   method            from
#>   as.zoo.data.frame zoo

# Twitter Decomposition Method
p2 <- df %>%
    time_decompose(unemploy_rate_la,
                   method    = "twitter") %>%
    anomalize(remainder) %>%
    plot_anomaly_decomposition() +
    ggtitle("Twitter Decomposition")
```

```
## Converting from tbl_df to tbl_time.
## Auto-index message: index = DATE
```

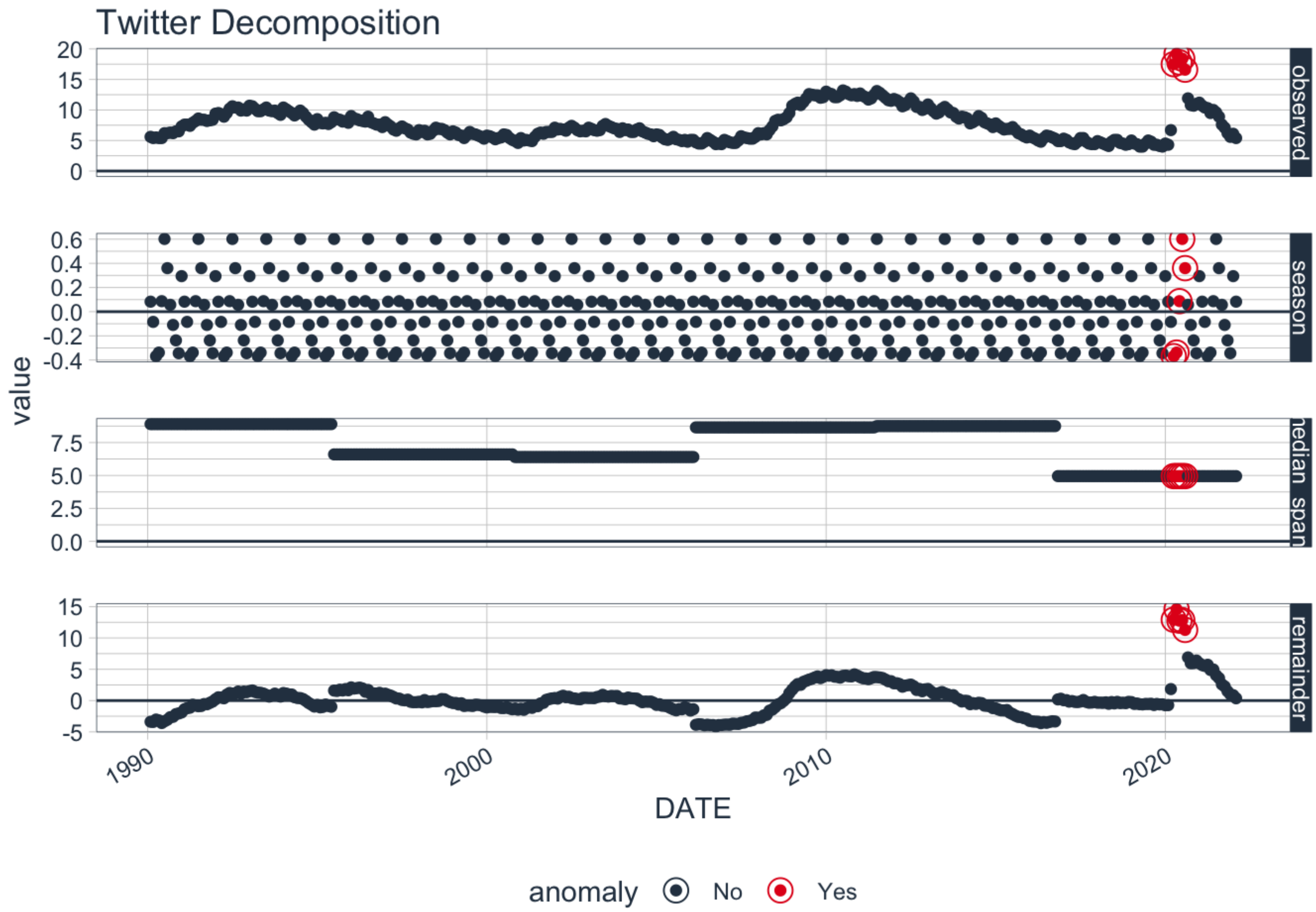
```
## frequency = 12 months
```

```
## median_span = 64 months
```

```
#> frequency = 7 days
#> median_span = 85 days

# Show plots
p1
```

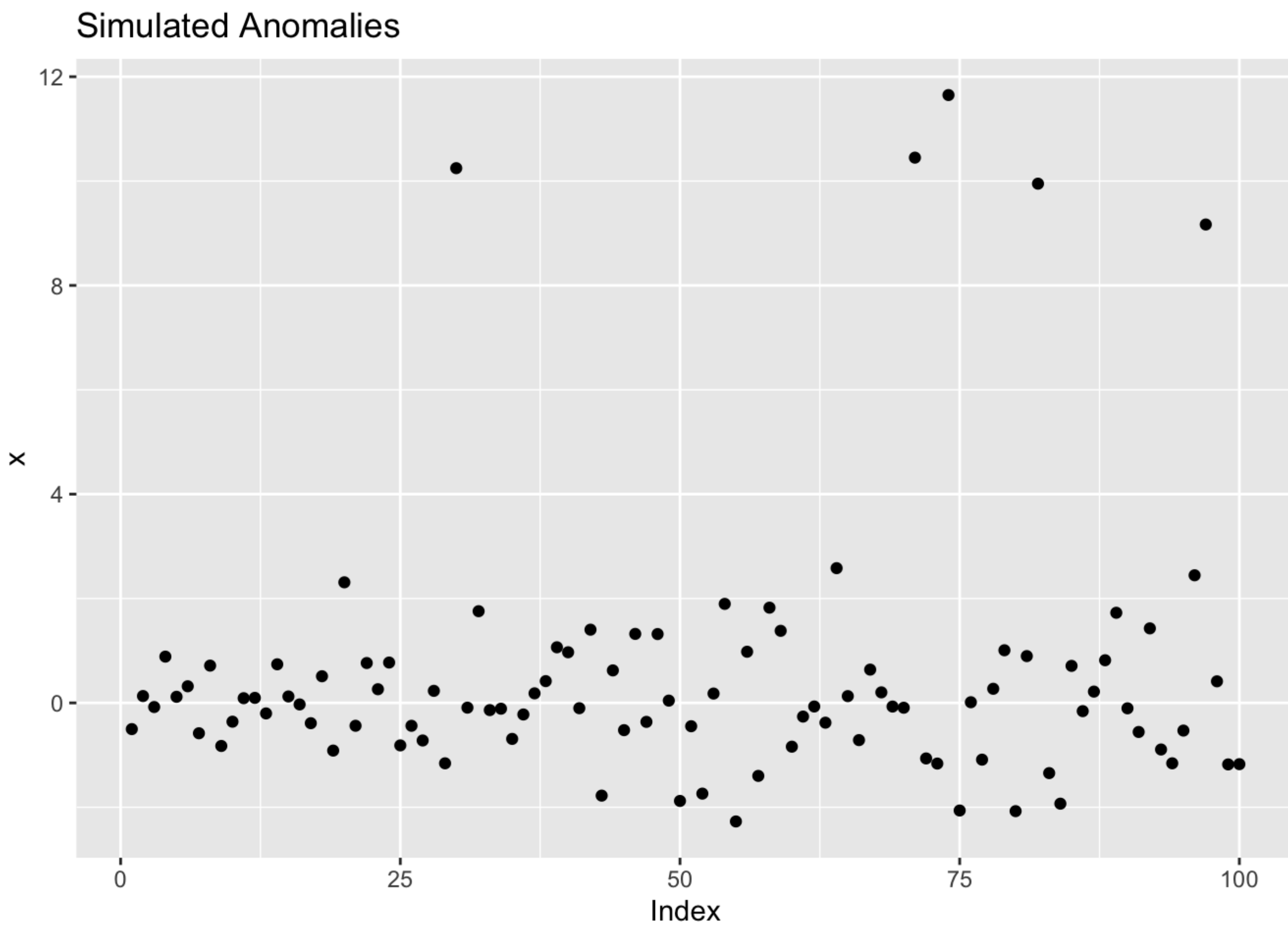

STL Decomposition
observed
season
trend
remainder
value
DATE
1990
2000
2010
2020
anomaly
No
Yes

```
p2
```

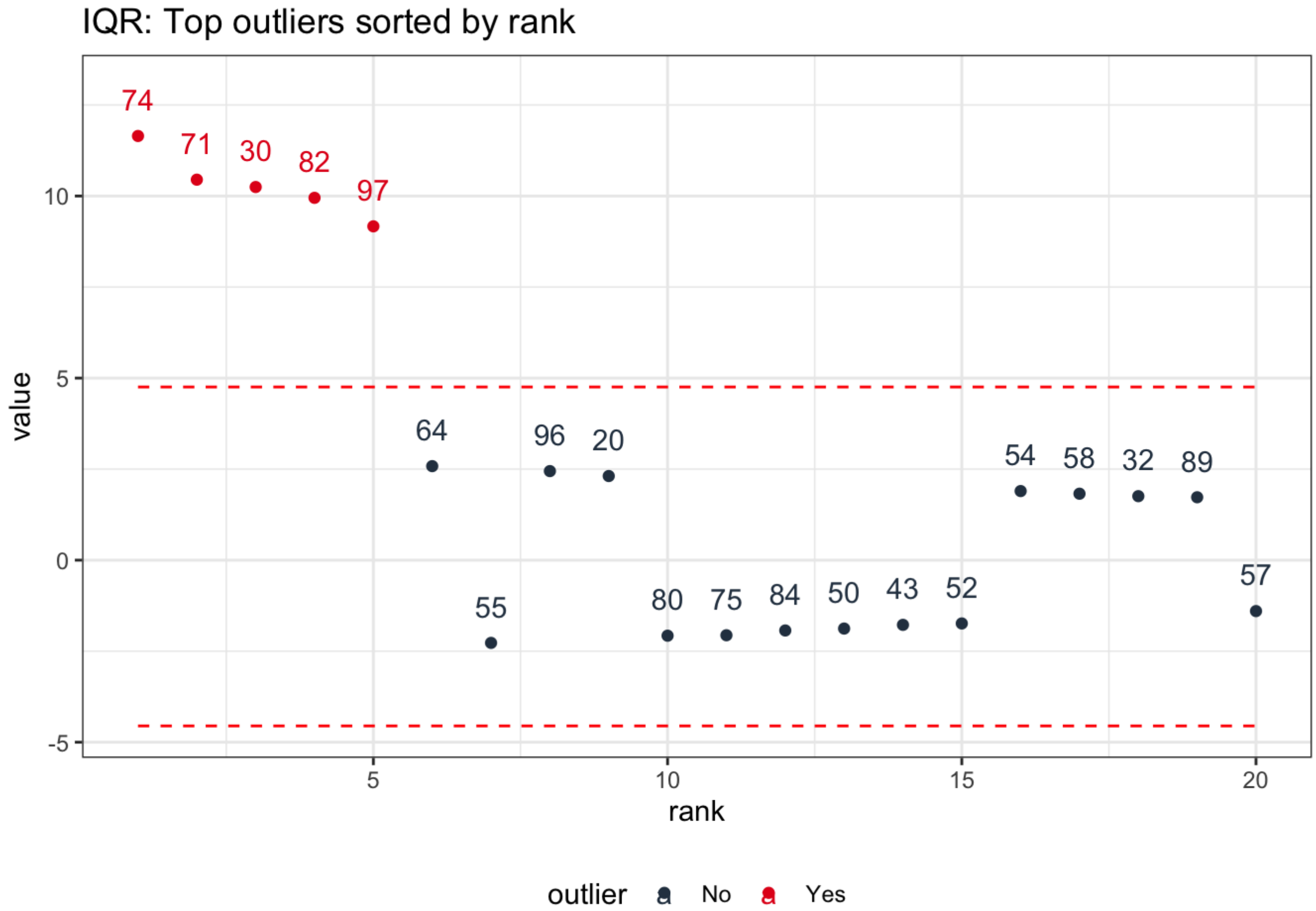


Comparison of IQR and GESD Methods

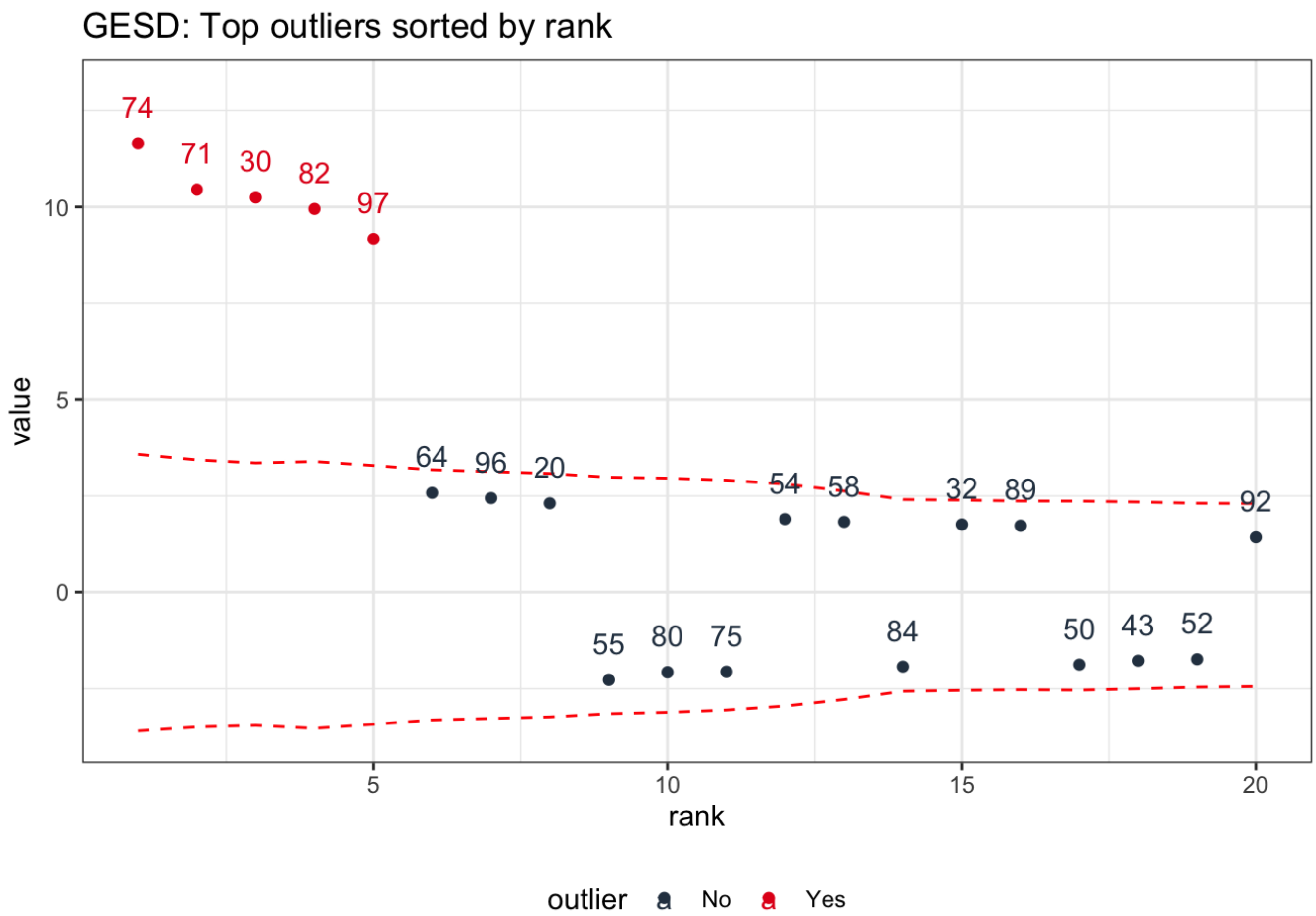
```
# Generate anomalies
set.seed(100)
x <- rnorm(100)
idx_outliers    <- sample(100, size = 5)
x[idx_outliers] <- x[idx_outliers] + 10

# Visualize simulated anomalies
qplot(1:length(x), x,
      main = "Simulated Anomalies",
      xlab = "Index")
```


Simulated Anomalies
12
8
4
0
x
0
25
50
75
100
Index

```
# Analyze outliers: Outlier Report is available with verbose = TRUE
iqr_outliers <- iqr(x, alpha = 0.05, max_anoms = 0.2, verbose = TRUE)$outlier_report

gesd_outliers <- gesd(x, alpha = 0.05, max_anoms = 0.2, verbose = TRUE)$outlier_report

# ploting function for anomaly plots
ggsetup <- function(data) {
    data %>%
        ggplot(aes(rank, value, color = outlier)) +
        geom_point() +
        geom_line(aes(y = limit_upper), color = "red", linetype = 2) +
        geom_line(aes(y = limit_lower), color = "red", linetype = 2) +
        geom_text(aes(label = index), vjust = -1.25) +
        theme_bw() +
        scale_color_manual(values = c("No" = "#2c3e50", "Yes" = "#e31a1c")) +
        expand_limits(y = 13) +
        theme(legend.position = "bottom")
}


# Visualize
p3 <- iqr_outliers %>%
    ggsetup() +
    ggtitle("IQR: Top outliers sorted by rank")

p4 <- gesd_outliers %>%
    ggsetup() +
    ggtitle("GESD: Top outliers sorted by rank")

# Show plots
p3
```

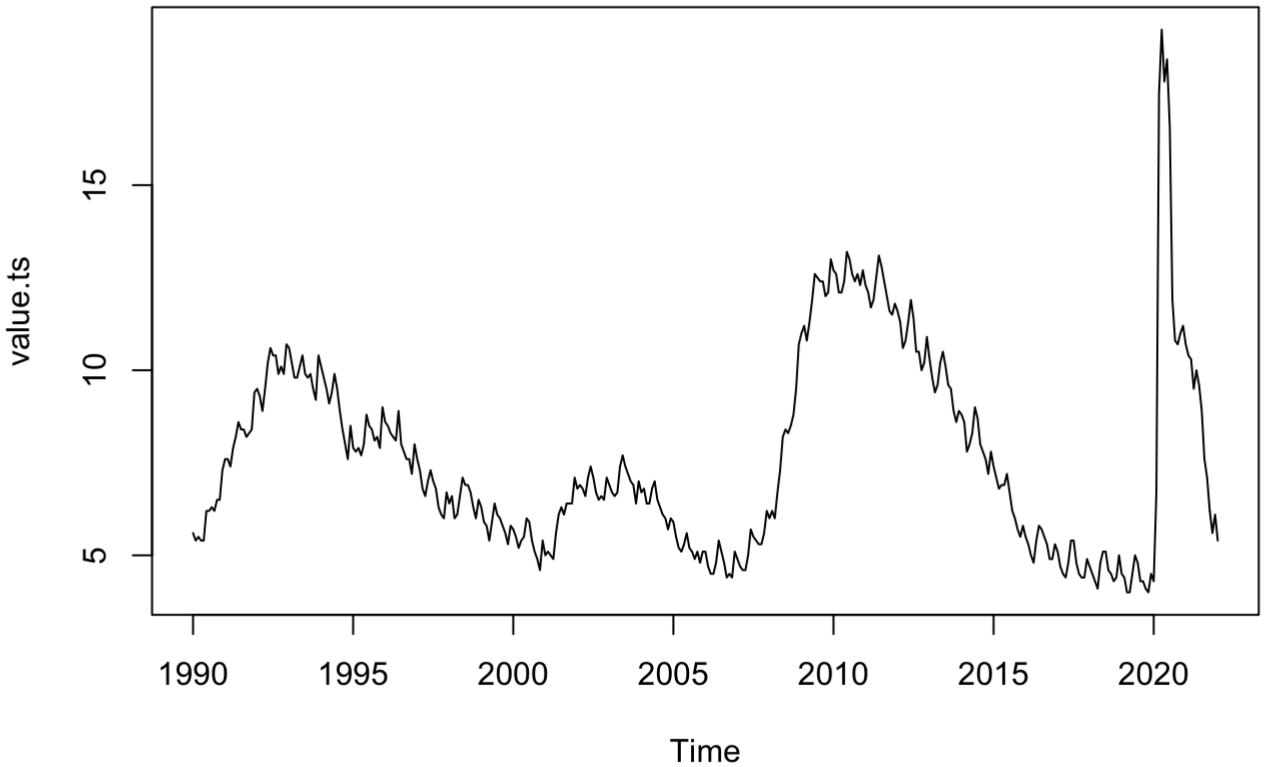

IQR: Top outliers sorted by rank
value
rank
10
5
0
-5
5
10
15
20
74
71
30
82
97
64
96
20
55
80
75
84
50
43
52
54
58
32
89
57
outlier
No
Yes

```
p4
```



Change-point Detection

```
processed_data$DATE <- as.Date(processed_data$DATE, "%Y/%m/%d") # change date from chara
cters to date element
data <- tibble(processed_data$DATE,processed_data$unemploy_rate_la) %>%
  rename("unemploy_rate_la" = "processed_data$unemploy_rate_la", "date" = "processed_dat
a$DATE")
value.ts = ts(data$unemploy_rate_la,start = c(1990,1),end = c(2022,1), frequency = 12)
plot(value.ts)
```

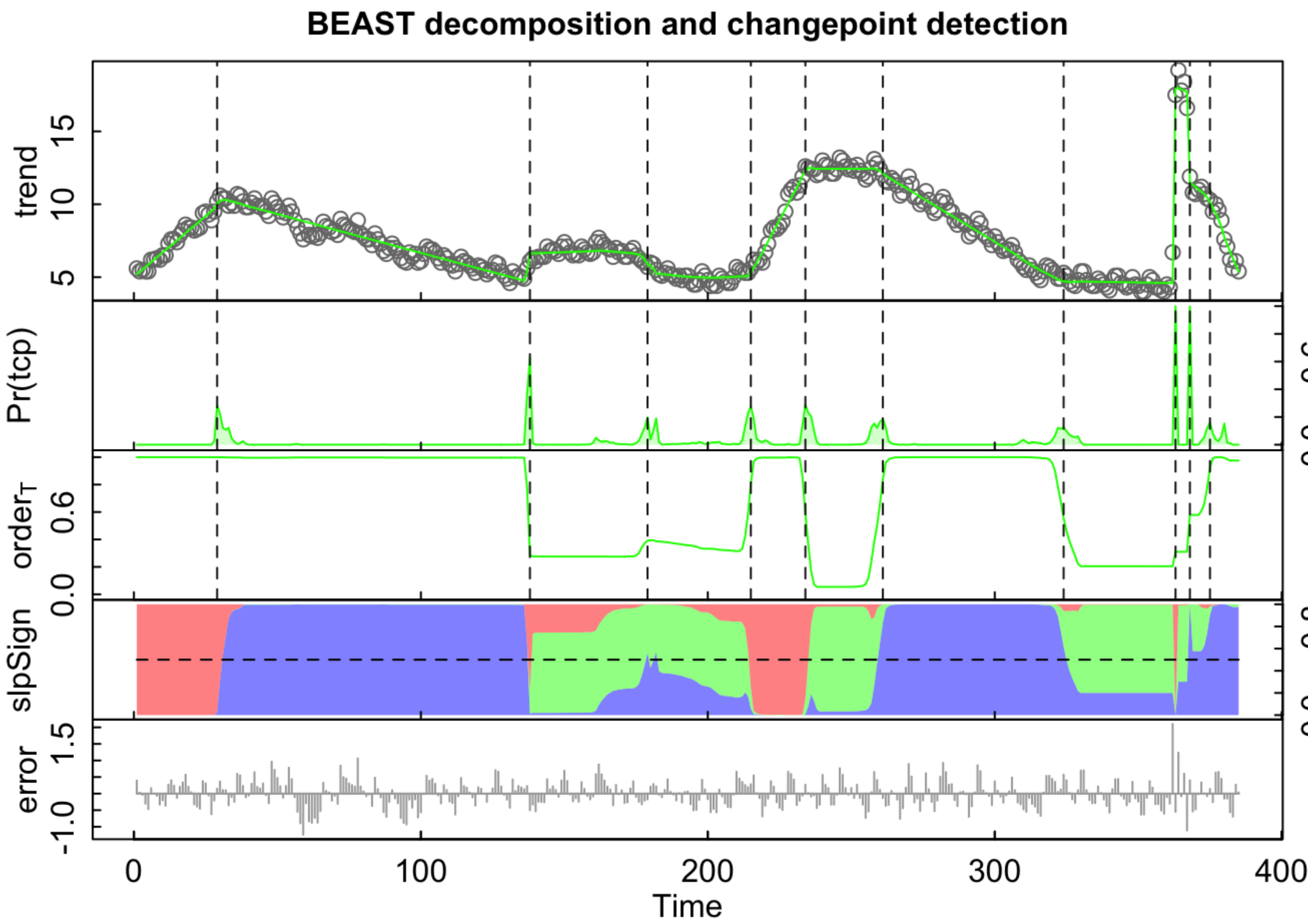



```
## Using Rbeast
y <- data$unemploy_rate_la
out=beast(y, season='none')
```

```
##
## #--------------------------------------------------#
## #      OPTIONS used in the MCMC inference          #
## #--------------------------------------------------#
## #  Set extra$printOptions=0 to suppress printing   #
## #--------------------------------------------------#
##
## #......Start of displaying 'MetaData' ......
##   metadata = list()
##   metadata$isRegularOrdered = TRUE
##   metadata$season           = 'none'
##   metadata$startTime        = 1.00000
##   metadata$deltaTime        = 1.00000
##   metadata$whichDimIsTime   = 1
##   metadata$missingValue     = NaN
##   metadata$maxMissingRate   = 0.7500
##   metadata$detrend          = FALSE
## #........End of displaying MetaData ........
##
## #......Start of displaying 'prior' ......
##   prior = list()
##   prior$modelPriorType     = 1
##   prior$trendMinOrder      = 0
##   prior$trendMaxOrder      = 1
##   prior$trendMinKnotNum  = 0
##   prior$trendMaxKnotNum  = 10
##   prior$trendMinSepDist  = 3
##   prior$K_MAX            = 22
##   prior$precValue        = 1.500000
##   prior$precPriorType    = 'uniform'
## #......End of displaying pripr ......
##
## #......Start of displaying 'mcmc' ......
##   mcmc = list()
##   mcmc$seed                      = 0
##   mcmc$samples                   = 8000
##   mcmc$thinningFactor            = 5
##   mcmc$burnin                    = 200
##   mcmc$chainNumber               = 3
##   mcmc$maxMoveStepSize           = 4
##   mcmc$trendResamplingOrderProb  = 0.1000
##   mcmc$credIntervalAlphaLevel    = 0.950
## #......End of displaying mcmc ......
##
## #......Start of displaying 'extra' ......
##   extra = list()
##   extra$dumpInputData        = TRUE
##   extra$whichOutputDimIsTime = 1
##   extra$computeCredible      = FALSE
##   extra$fastCIComputation    = TRUE
##   extra$computeTrendOrder    = TRUE
##   extra$computeTrendChngpt   = TRUE
```

```
##   extra$computeTrendSlope    = TRUE
##   extra$tallyPosNegTrendJump = TRUE
##   extra$tallyIncDecTrendJump = TRUE
##   extra$printProgressBar     = TRUE
##   extra$printOptions         = TRUE
##   extra$consoleWidth         = 80
##   extra$numThreadsPerCPU     = 2
##   extra$numParThreads        = 0
## #......End of displaying extra ......
##
##
-Progress:  0.0% done[>********************************************************]
\Progress:  4.2% done[==>******************************************************]
|Progress:  8.3% done[=====>***************************************************]
/Progress: 12.5% done[=======>*************************************************]
-Progress: 16.7% done[==========>**********************************************]
\Progress: 20.8% done[============>********************************************]
|Progress: 25.0% done[==============>******************************************]
/Progress: 29.2% done[=================>***************************************]
-Progress: 33.3% done[===================>*************************************]
\Progress: 37.5% done[=====================>***********************************]
|Progress: 41.7% done[========================>********************************]
/Progress: 45.8% done[==========================>******************************]
-Progress: 50.0% done[=============================>***************************]
\Progress: 54.2% done[===============================>*************************]
|Progress: 58.3% done[=================================>***********************]
/Progress: 62.5% done[====================================>********************]
-Progress: 66.7% done[======================================>******************]
\Progress: 70.8% done[========================================>****************]
|Progress: 75.0% done[===========================================>*************]
/Progress: 79.2% done[=============================================>***********]
-Progress: 83.3% done[================================================>********]
\Progress: 87.5% done[==================================================>******]
|Progress: 91.7% done[====================================================>****]
/Progress: 95.8% done[=======================================================>*]
-Progress:100.0% done[=========================================================]
```

```
plot(out)
```


BEAST decomposition and changepoint detection
trend
5
10
15
Pr(tcp)
0.0
0.6
orderT
0.0
0.6
slpSign
0.0
0.8
error
-1.0
1.5
0
100
200
300
400
Time

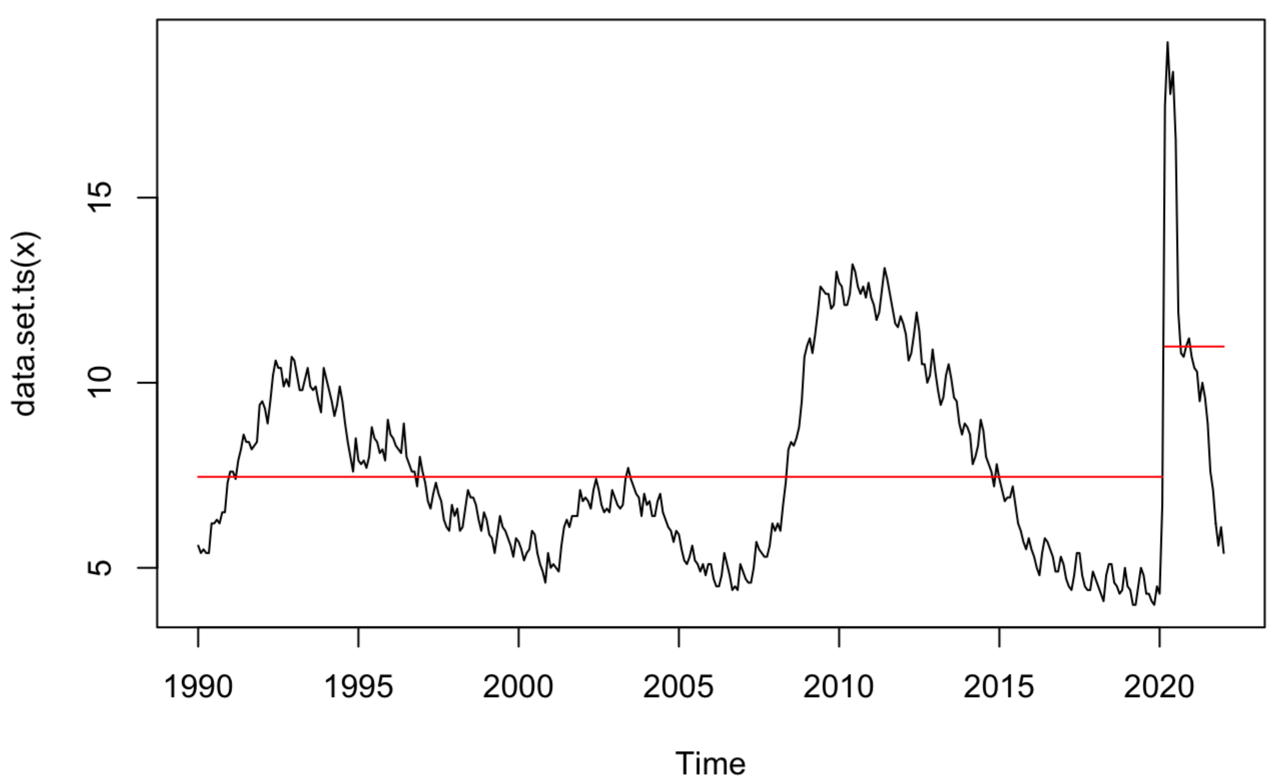
```
print(out)
```

```
##  [1;31m####################################################################
## #                        Seasonal  Changepoints                        #
## ######################################################################
##  [0m No seasonal/periodic component present (i.e., season='none')
##
##
##  [1;31m####################################################################
## #                        Trend  Changepoints                           #
## ######################################################################
##  [0m.-------------------------------------------------------------------.
## | Ascii plot of probability distribution for number of chgpts (ncp) |
## .-------------------------------------------------------------------.
## |Pr(ncp = 0 )=0.000|*                                               |
## |Pr(ncp = 1 )=0.000|*                                               |
## |Pr(ncp = 2 )=0.000|*                                               |
## |Pr(ncp = 3 )=0.000|*                                               |
## |Pr(ncp = 4 )=0.000|*                                               |
## |Pr(ncp = 5 )=0.000|*                                               |
## |Pr(ncp = 6 )=0.000|*                                               |
## |Pr(ncp = 7 )=0.000|*                                               |
## |Pr(ncp = 8 )=0.000|*                                               |
## |Pr(ncp = 9 )=0.008|*                                               |
## |Pr(ncp = 10)=0.991|*********************************************** |
## .-------------------------------------------------------------------.
## |    Summary for number of Trend ChangePoints (tcp)                 |
## .-------------------------------------------------------------------.
## |ncp_max    = 10   | MaxTrendKnotNum: A parameter you set           |
## |ncp_mode   = 10   | Pr(ncp=10)=0.99: There is a 99.1% probability  |
## |                  | that the trend component has 10 changepoint(s).|
## |ncp_mean   = 9.99 | Sum{ncp*Pr(ncp)} for ncp = 0,...,10            |
## |ncp_pct10  = 10.00 | 10% percentile for number of changepoints      |
## |ncp_median = 10.00 | 50% percentile: Median number of changepoints  |
## |ncp_pct90  = 10.00 | 90% percentile for number of changepoints      |
## .-------------------------------------------------------------------.
## | List of probable trend changepoints ranked by probability of      |
## | occurrence: Please combine the ncp reported above to determine    |
## | which changepoints below are  practically meaningful              |
## '-------------------------------------------------------------------'
## |tcp#              |time (cp)                  |prob(cpPr)          |
## |------------------|---------------------------|--------------------|
## |1                 |363.000000                 |1.00000             |
## |2                 |368.000000                 |0.99833             |
## |3                 |138.000000                 |0.99288             |
## |4                 |234.000000                 |0.72175             |
## |5                 |215.000000                 |0.69033             |
## |6                 |29.000000                  |0.62588             |
## |7                 |261.000000                 |0.47642             |
## |8                 |179.000000                 |0.39742             |
## |9                 |324.000000                 |0.36246             |
## |10                |375.000000                 |0.35917             |
## .-------------------------------------------------------------------.
##
```

```
##
##
## NOTE: the beast output object 'o' is a LIST. Type 'str(o)' to see all
## the elements in it. Or use 'plot(o)' or 'plot(o,interactive=TRUE)' to
## plot the model output.
```

```
# cpt.mean - mean only changes
# cpt.var - variance only changes
# cpt.meanvar - mean and variance changes
```

```
# How do we know if a change-point found is significant or not?
# We calculate cost of the whole data with no change
# If the difference is large enough then we say there is no change
# default change-point metric in changepoint package to test if there is a change point
or
# not is MBIC - a Modified Bayesian Information Criterion

## Using changepoint
m1.amoc = cpt.mean(value.ts, penalty = "MBIC")
#cpts(m1.amoc) # checks for at most one change-point value
plot(m1.amoc)
```



```
# we can see that we definitely need way more change points that just one
#LA.default = cpt.mean(value.ts)
#cpts(LA.default)
#param.est(LA.default)

# it can be seen that the variance might not be equal to 1, so we must appropriately sca
le it
#LA.scale = cpt.mean(as.vector(scale(data$unemploy_rate_la)))
#cpts(LA.scale)
```

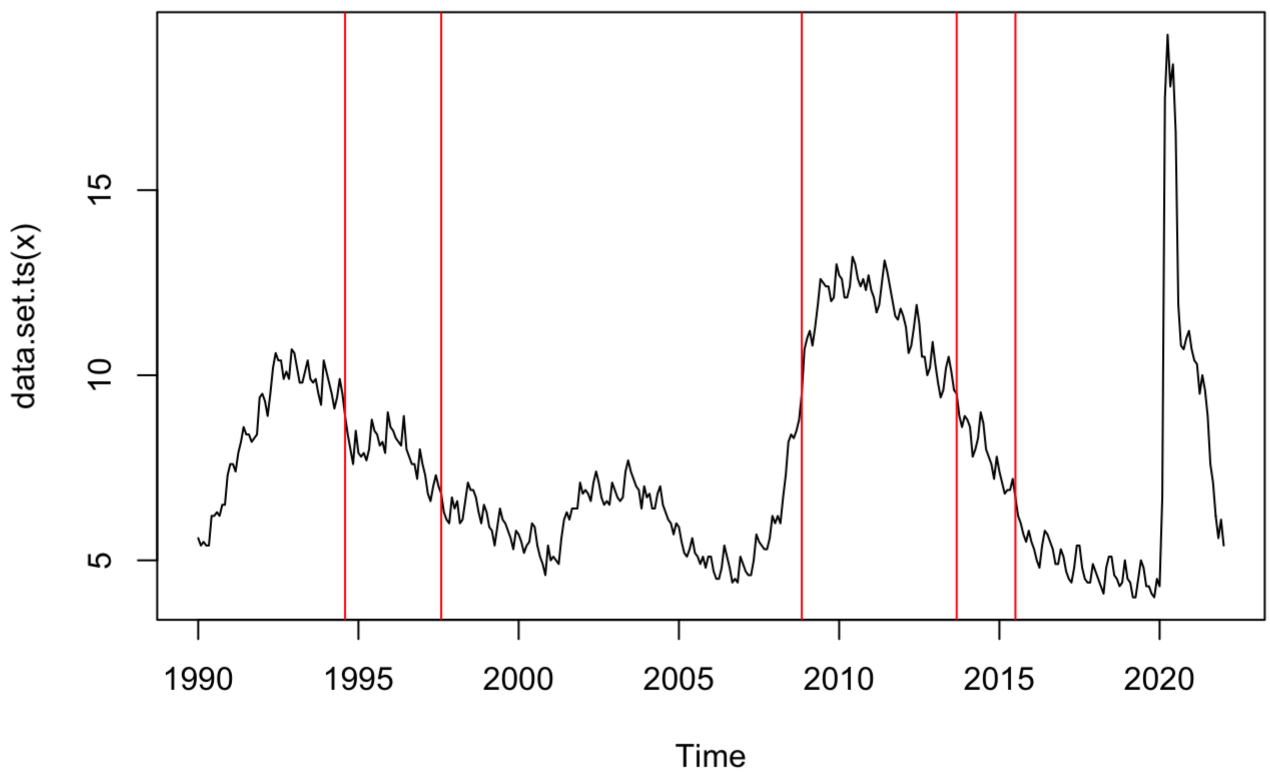
```
# since we still get the same changepoint after scaling the variance, that shows that th
is is actually a changepoint
# and not just because we forced it to show a change point
m2.man = cpt.var(value.ts,method = "PELT")
cpts(m2.man)
```

```
## [1]  56  92 227 285 307
```

```
param.est(m2.man)
```

```
## $variance
## [1]  2.5815784  0.3584182  0.9341180  1.0841766  0.5650413 12.9404471
##
## $mean
## [1] 7.668831
```

```
plot(m2.man)
```



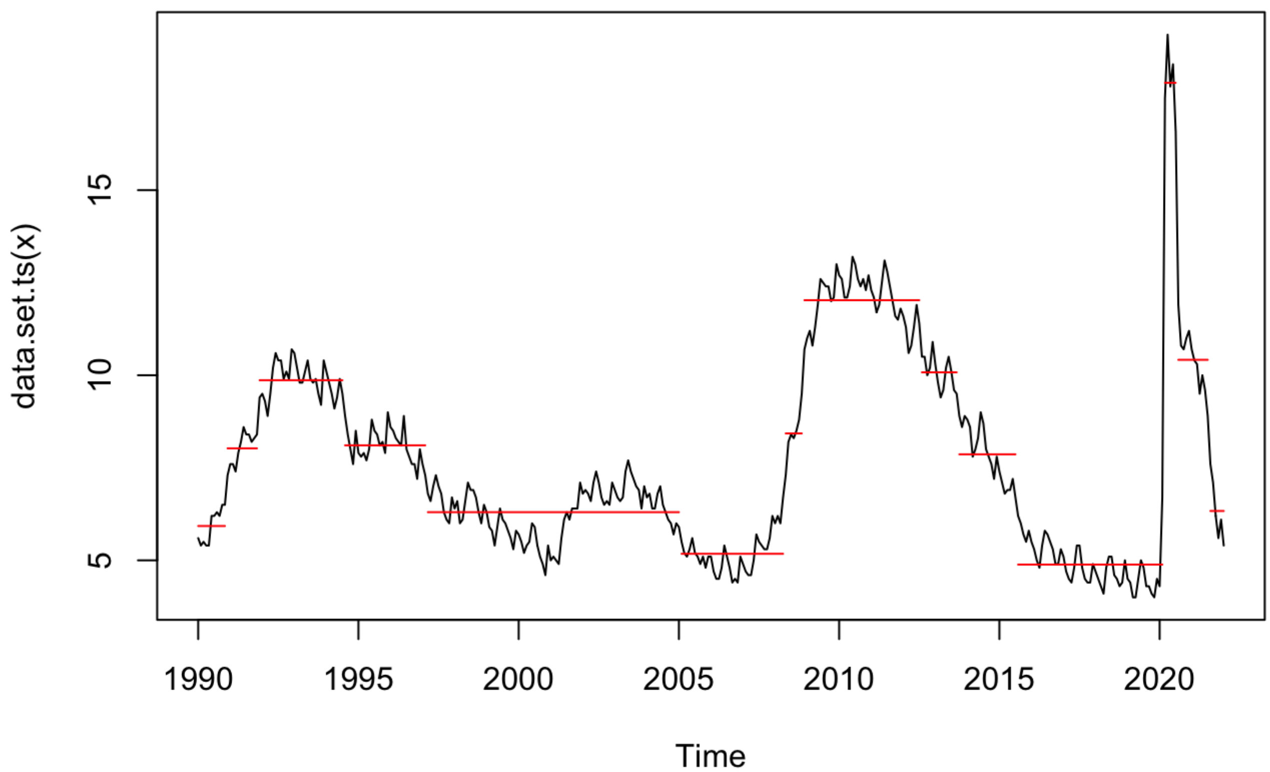

```
## using mean
m3.man = cpt.mean(value.ts,method = "PELT")
cpts(m3.man)
```

```
##  [1]  11  23  55  86 181 220 227 271 285 307 362 367 379
```

```
param.est(m3.man)
```

```
## $mean
##  [1]  5.927273  8.025000  9.865625  8.106452  6.301053  5.176923  8.428571
##  [8] 12.027273 10.078571  7.863636  4.885455 17.900000 10.416667  6.333333
```

```
plot(m3.man)
```




```
## finding changepoint with respect to variance and mean
mv1.pelt <- cpt.meanvar(value.ts, method = "PELT")
mv2.pelt <- cpt.meanvar(value.ts, method = "BinSeg")
```

```
## Warning in BINSEG(sumstat, pen = pen.value, cost_func = costfunc, minseglen
## = minseglen, : The number of changepoints identified is Q, it is advised to
## increase Q to make sure changepoints have not been missed.
```

```
length(cpts(mv1.pelt))
```

```
## [1] 52
```

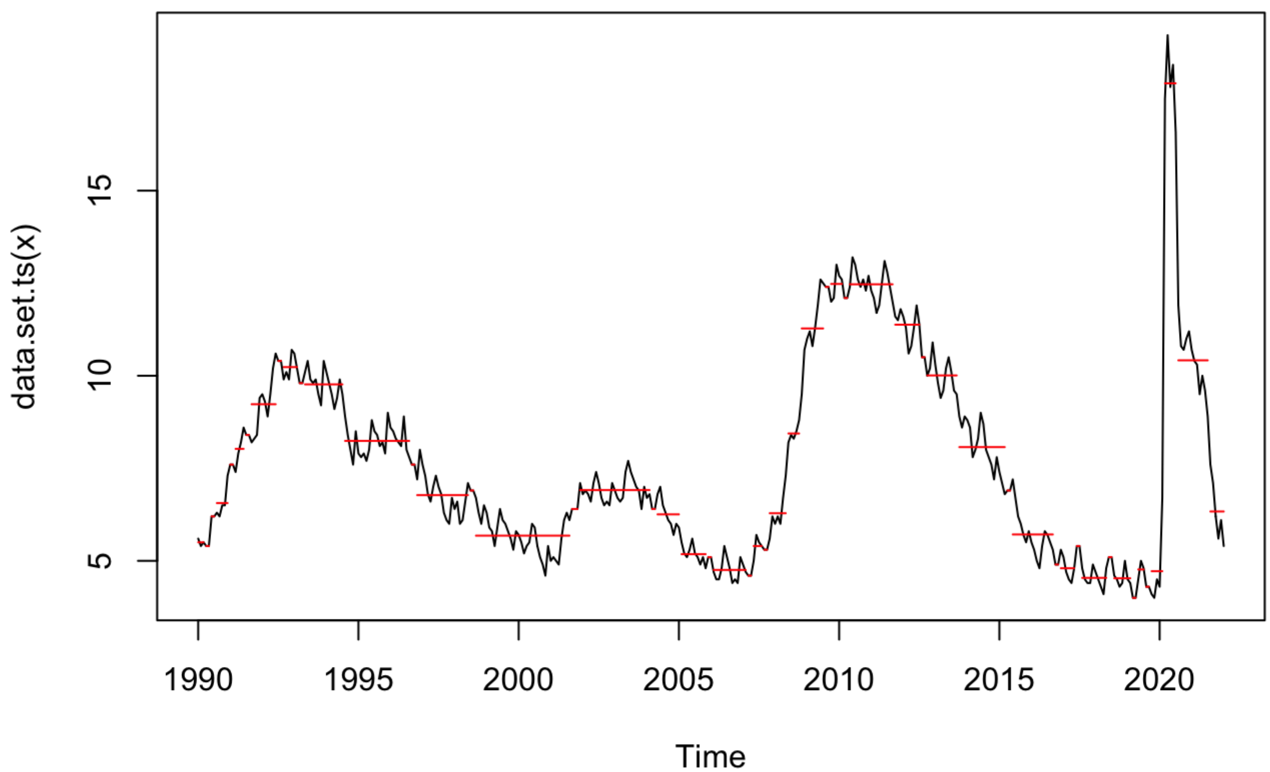
```
length(cpts(mv2.pelt))
```

```
## [1] 5
```

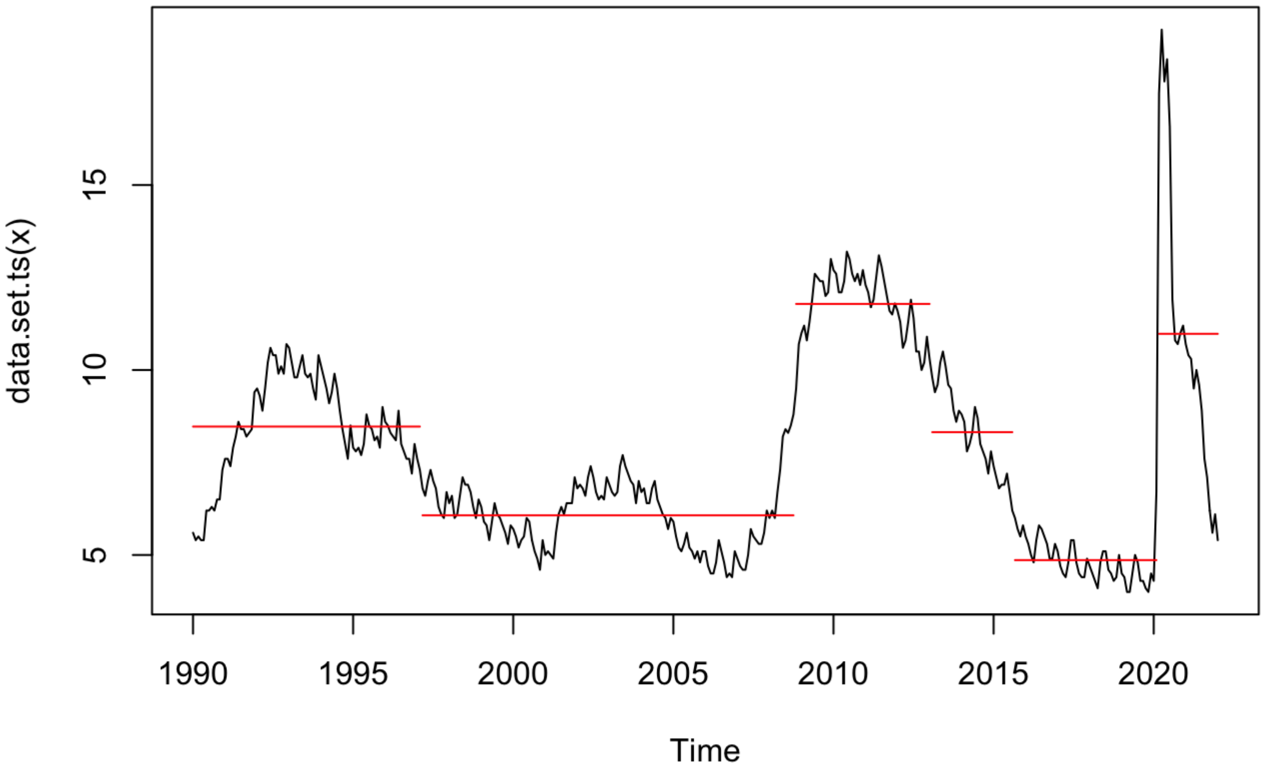
```
param.est(mv2.pelt)
```

```
## $mean
## [1]  8.470930  6.071429 11.786275  8.319355  4.861111 10.978261
##
## $variance
## [1]  1.8308991  0.8496122  0.7788312  1.3222060  0.3301543 16.8573535
```

```
plot(mv1.pelt)
```




```
plot(mv2.pelt)
```




```
# notice that PELT produces way too many points so overall it's not that useful to our a
nalysis, therefore
# we should change our method, thus I opted for BinSeg. And we already can see that it p
roduces much more useful
# data than PELT does since it shows meaning points of interest.
```