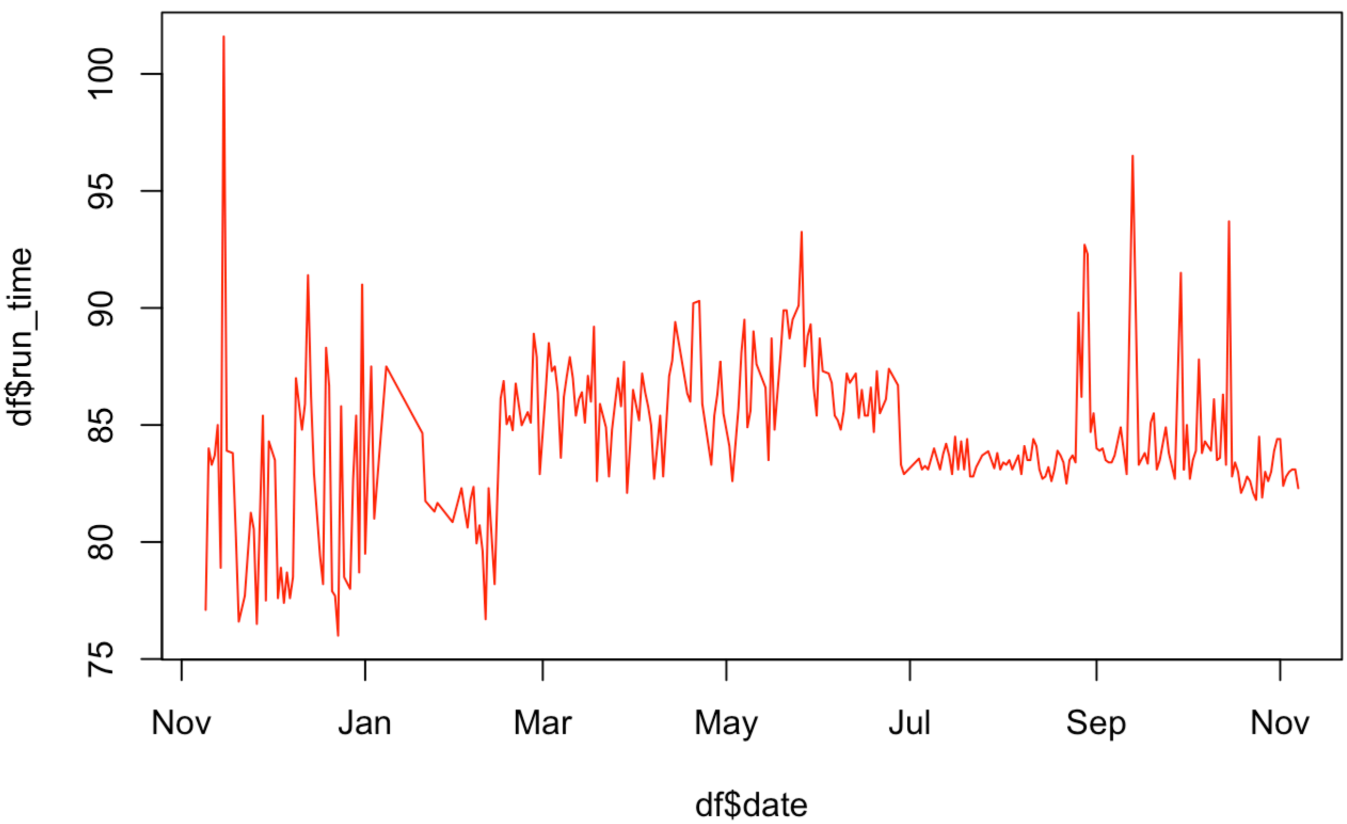
```
df <- readr::read_csv("UBHPC_8cores_NWChem_Wall_Clock_Time.csv",show_col_types = FALS
E)
df <- df %>% mutate(date = as.Date(date, format="%m/%d/%Y %H:%M")) %>%
  as_tsibble(index = date)
head(df)
```

```
## # A tsibble: 6 x 2 [1D]
##   date       run_time
##   <date>        <dbl>
## 1 2017-11-09     77.1
## 2 2017-11-10     84
## 3 2017-11-11     83.3
## 4 2017-11-12     83.7
## 5 2017-11-13     85
## 6 2017-11-14     78.9
```

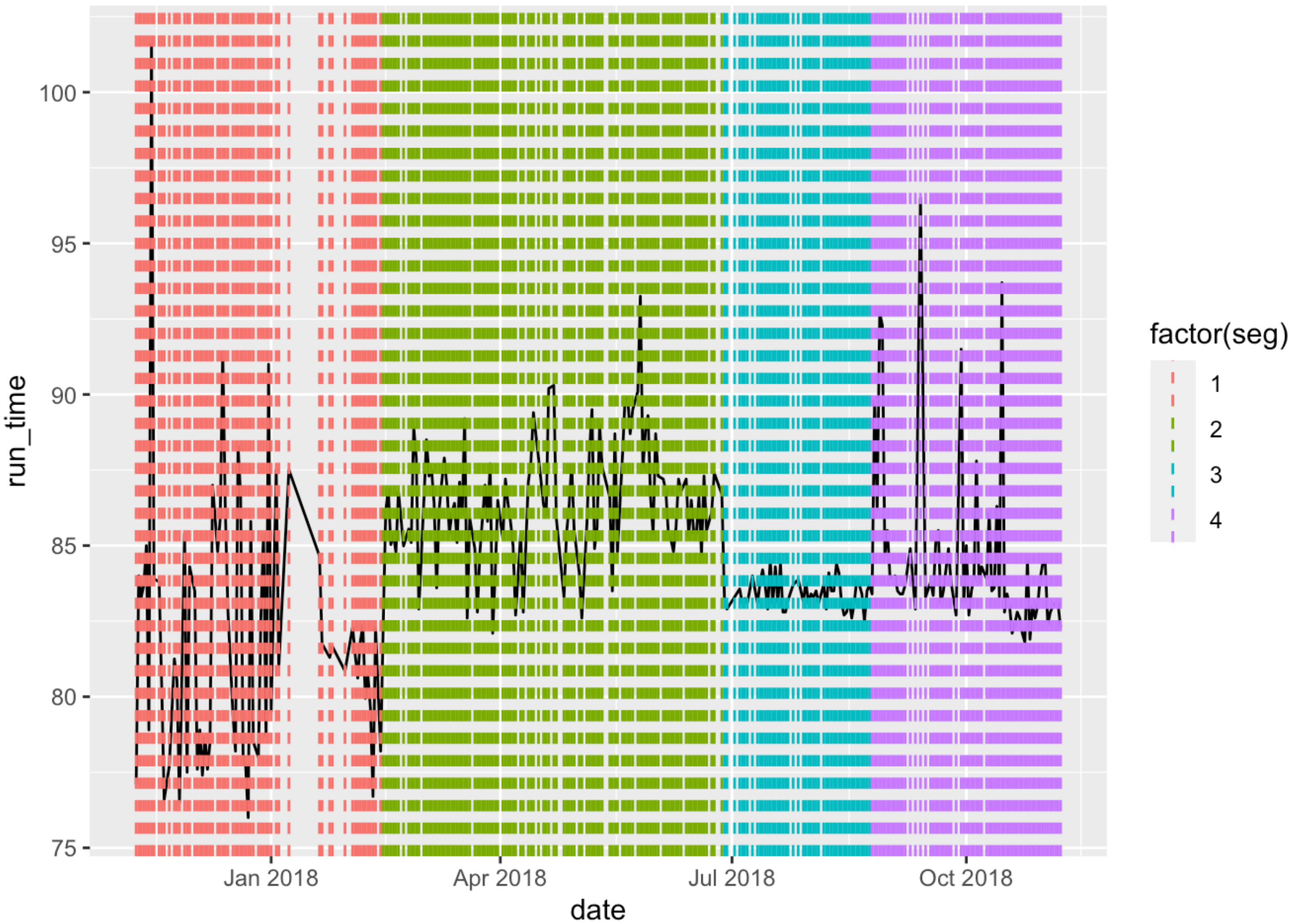
```
plot(df$date, df$run_time, type = "l", col = "red")
```



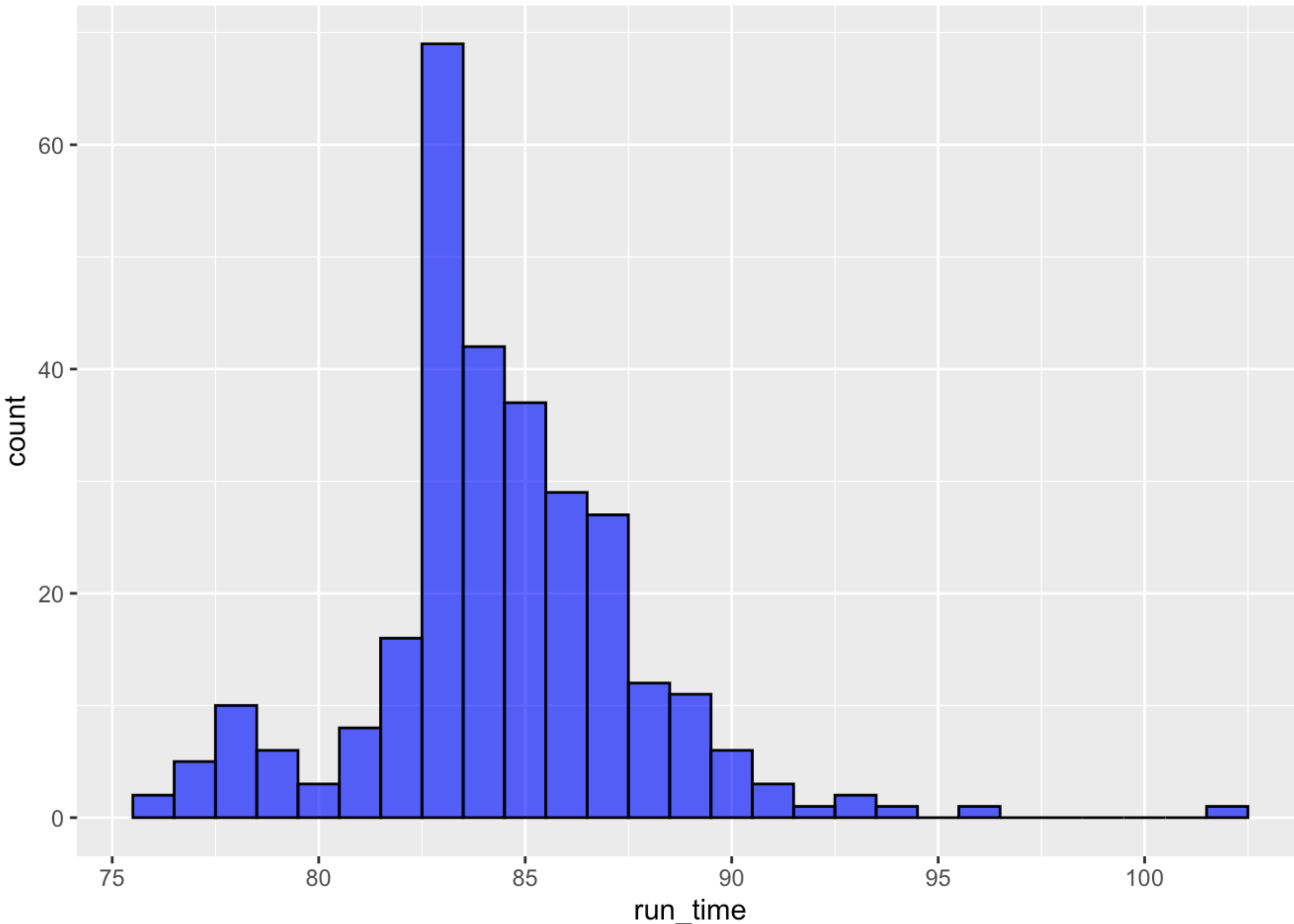
Due to sudden change in magnitude and trends pattern I can make out 4 segments in the plot. Below are the change points segmenting those which I can find: >“2017-11-09” to “2018-02-14” is the first segment. >“2018-02-14” to “2018-06-28” is the second segment. >“2018-06-28” to “2018-08-25” is the third segment. >“2018-08-25” to “2018-11-07” is the fourth segment.

```
df$seg = 0
change_point = c(ymd("2017-11-09"),ymd("2018-02-14"),ymd("2018-06-28"),ymd("2018-08-2
5"),ymd("2018-11-07"))

for (i in seq_along(change_point)) {
  df$seg[df$date >= change_point[i] & df$date < change_point[i + 1]] <- i
}

df <- df %>%
  mutate(seg = ifelse(seg == 0, 4, seg))
```

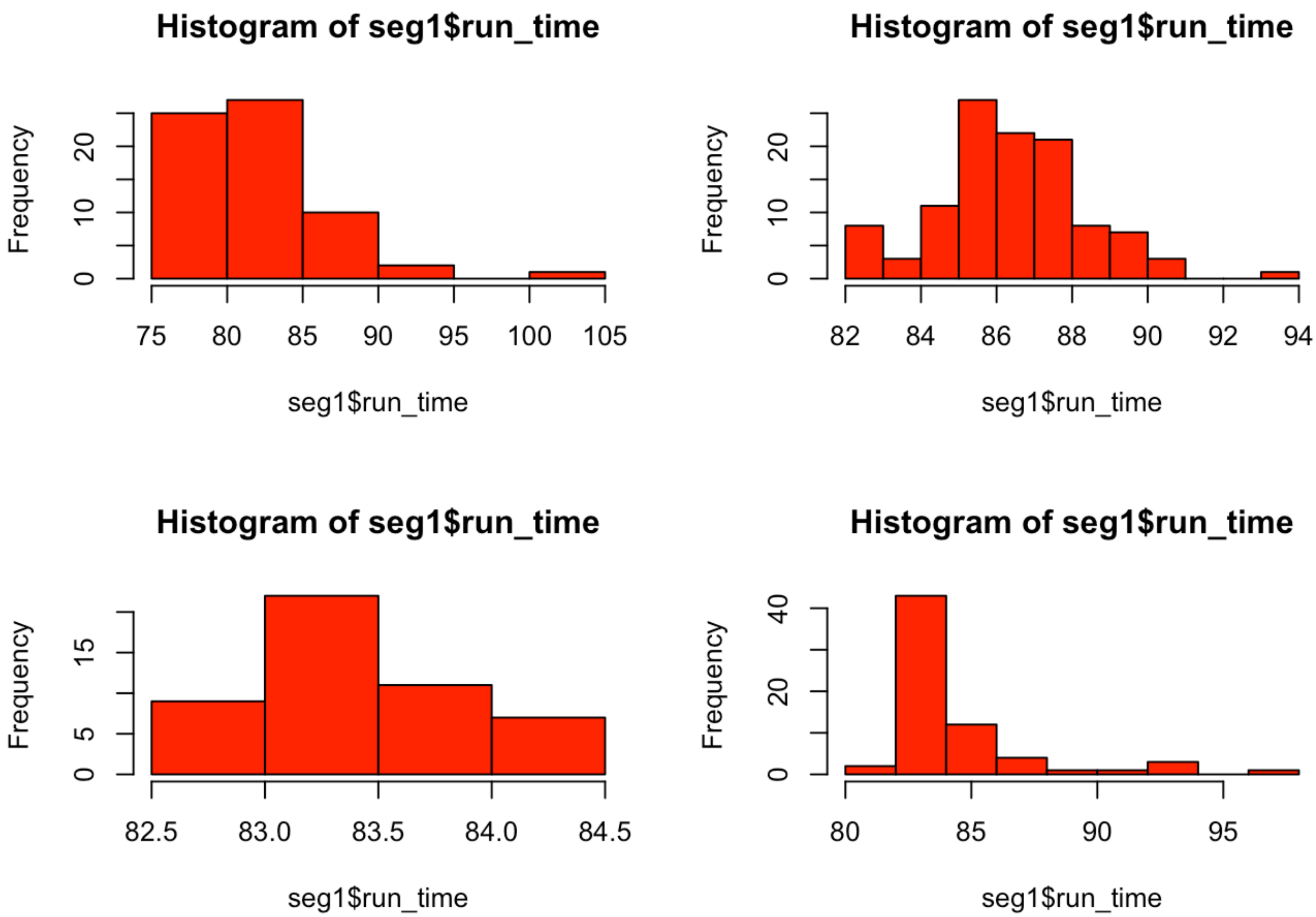
```
df %>%
  ggplot(aes(x = date, y = run_time)) +
  geom_line() +
  geom_vline(data = df, aes(xintercept = date, color = factor(seg)), linetype = "dash
ed")
```



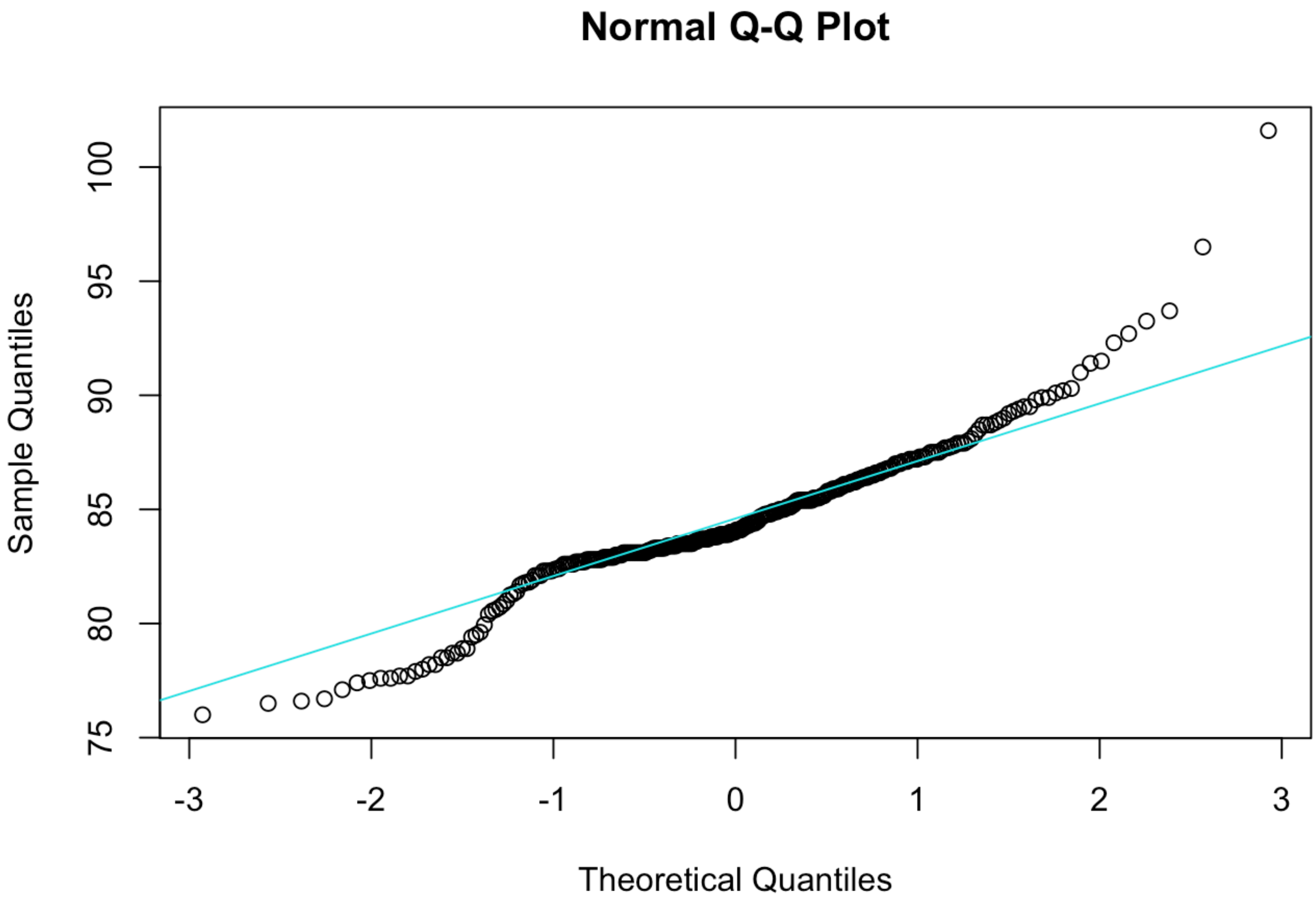

```
ggplot(df, aes(x = run_time)) +
  geom_histogram(binwidth = 1, fill = "blue", color = "black", alpha = 0.7)
```



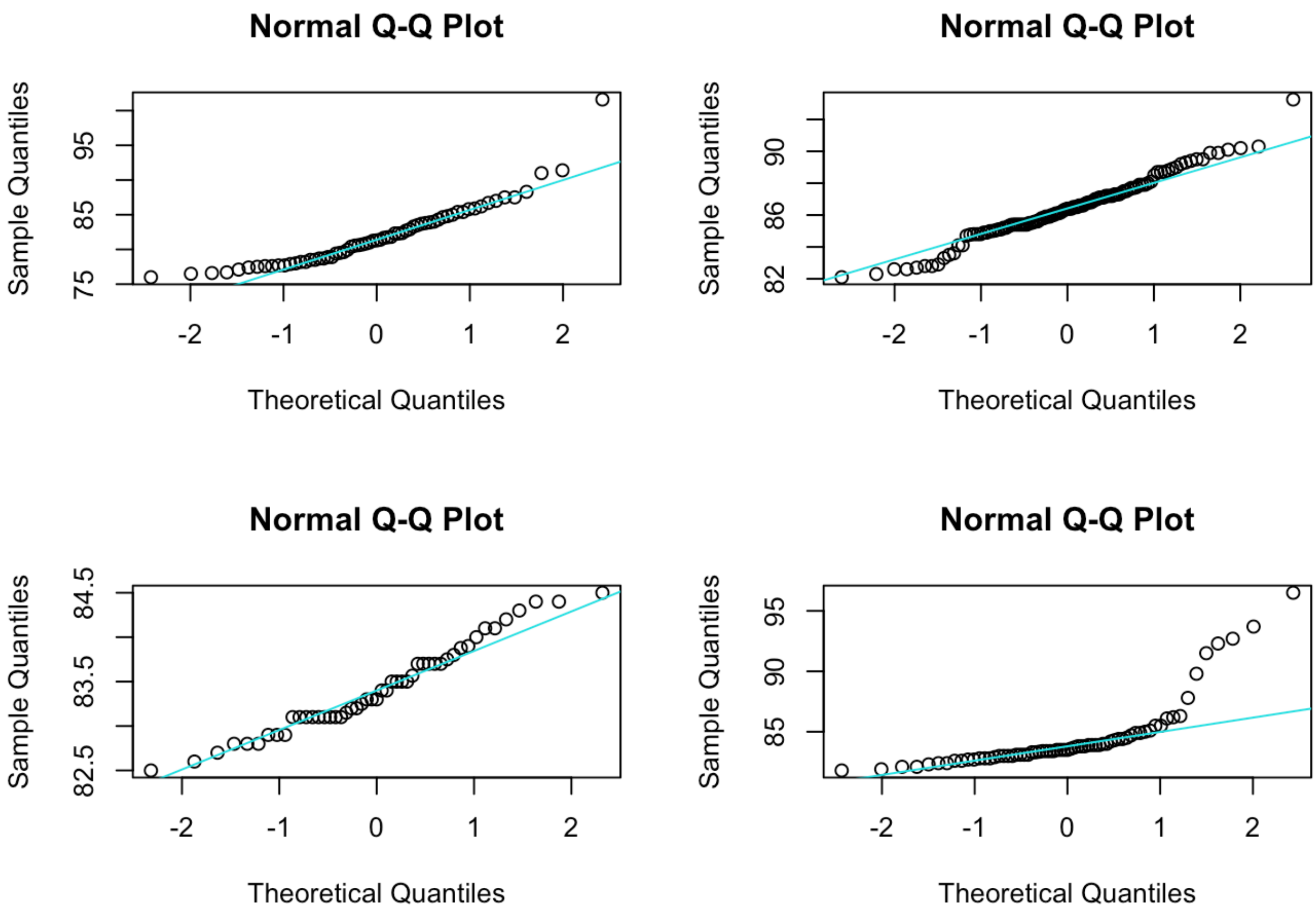

```r
par(mfrow = c(2, 2))
for (i in 1:max(df$seg)) {
  seg1 <- df[df$seg == i, ]
  hist(seg1$run_time,col = "red")
}
```




```
qqnorm(df$run_time)
qqline(df$run_time, col = 5)
```




```
par(mfrow = c(2, 2))
for (i in 1:max(df$seg)) {
  seg1 <- df[df$seg == i, ]
  qqnorm(seg1$run_time)
  qqline(seg1$run_time, col = 5)
}
```



Based on a visual review of the QQ plots and histograms for each segment, none of the distributions appear to be sufficiently normal. The presence of outliers in the QQ plots indicates that the data in these segments is not regularly distributed. When examining or modelling these segments, it is critical to consider other techniques or distributions because normality assumptions may not hold true.

```
cpt_result <- cpt.meanvar(df$run_time, method = "PELT", test.stat = "Normal")
cpts(cpt_result)
```

```
##  [1]  66 143 145 167 169 176 226 236 238 269 284 286
```

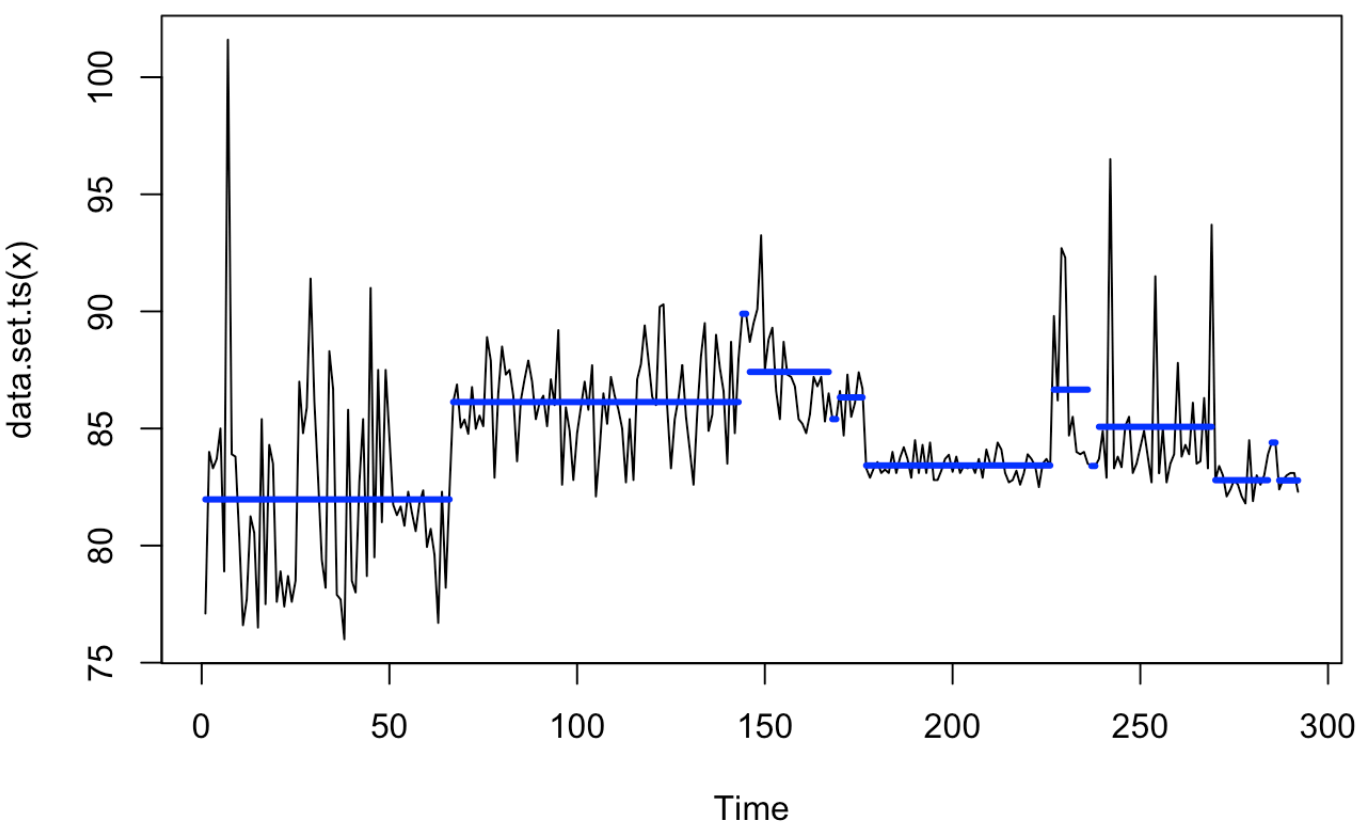
```
param.est(cpt_result)
```

```
## $mean
##  [1] 81.97241 86.12996 89.90000 87.41591 85.40000 86.32857 83.42189 86.66000
##  [9] 83.40000 85.07258 82.79333 84.40000 82.78333
##
## $variance
##  [1] 18.8877751  3.4362700  0.0000000  3.8375878  0.0000000  0.8134694
##  [7]  0.2405919 11.5304000  0.0000000 10.0009417  0.5072889  0.0000000
## [13]  0.1047222
```

```
print(pen.value(cpt_result))
```

```
## [1] 22.70702
```

```
plot(cpt_result,cpt.width=3,cpt.col='blue')
```



```
v1.crops=cpt.var(df$run_time,method="PELT",penalty="CROPS",pen.value=c(22.27/8, 22.2
7*4))
```

```
## [1] "Maximum number of runs of algorithm = 28"
## [1] "Completed runs = 2"
## [1] "Completed runs = 3"
## [1] "Completed runs = 5"
## [1] "Completed runs = 9"
## [1] "Completed runs = 16"
## [1] "Completed runs = 20"
## [1] "Completed runs = 23"
```

```
cpts.full(v1.crops)
```

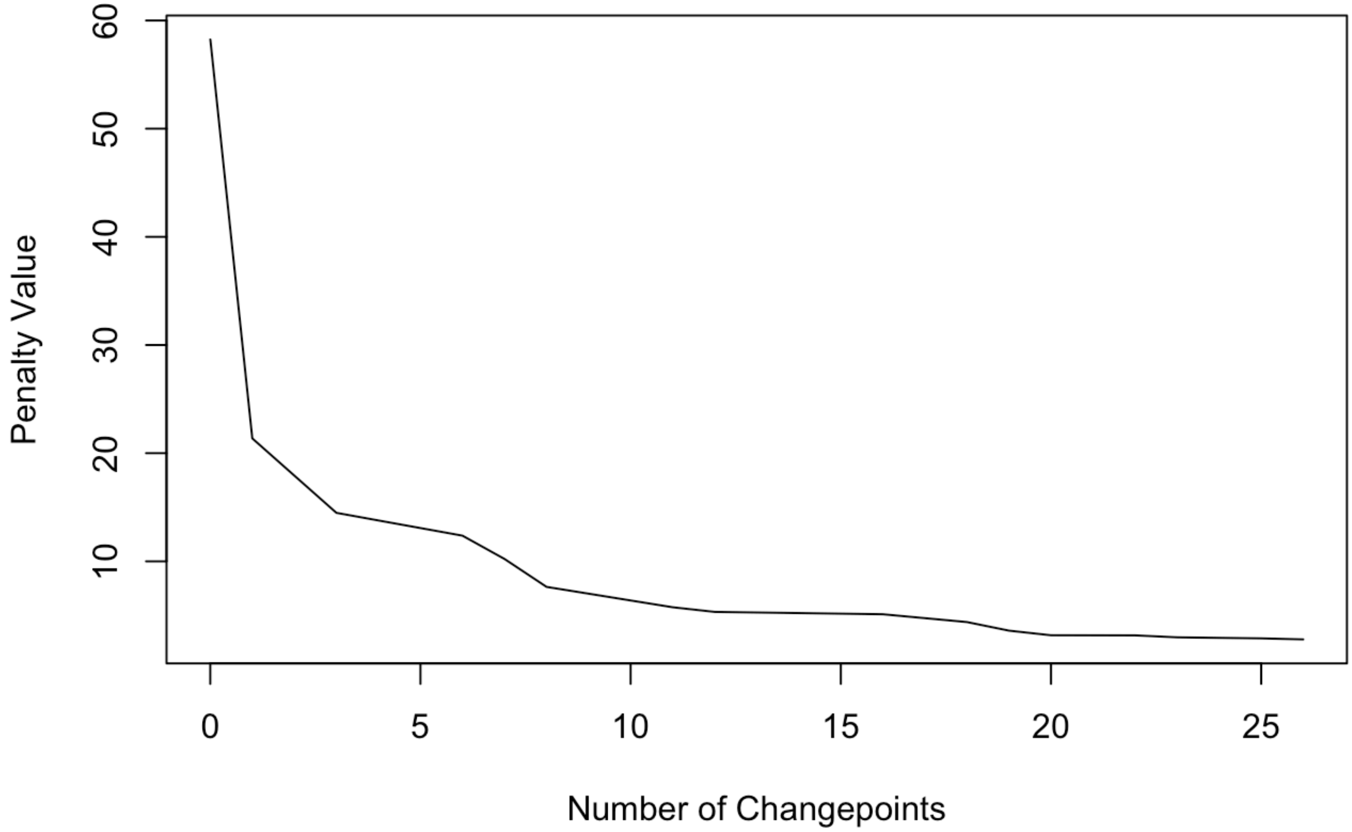
```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
##  [1,]    2    5    7    9   17   19   46   68   75    95   115   123   132
##  [2,]    2    5    7    9   17   19   65   75   95   115   123   132   158
##  [3,]    2    5    7    9   17   19   65  116  123   132   158   162   176
##  [4,]    2    5    7    9   17   19   65  116  123   132   157   176   226
##  [5,]    2    5    7    9   65  116  123  132  157   176   226   230   240
##  [6,]    2    5    7    9   65  115  143  152  176   226   230   240   242
##  [7,]    2    5    7    9   65  116  157  176  226   230   240   242   252
##  [8,]    2    5    7    9   65  116  157  176  226   230   240   242   252
##  [9,]   65  116  157  176  226  230  240  242  252   254   268   270    NA
## [10,]   65  116  158  226  230  240  242  252  254   268   270    NA    NA
## [11,]   65  116  158  226  230  240  242  252   NA    NA    NA    NA    NA
## [12,]   65  116  158  226  230  240  242   NA   NA    NA    NA    NA    NA
## [13,]   46  158  226  230  240  242   NA   NA   NA    NA    NA    NA    NA
## [14,]   46  158  226   NA   NA   NA   NA   NA   NA    NA    NA    NA    NA
## [15,]   65   NA   NA   NA   NA   NA   NA   NA   NA    NA    NA    NA    NA
## [16,]   NA   NA   NA   NA   NA   NA   NA   NA   NA    NA    NA    NA    NA
##       [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25]
##  [1,]   158   162   176   226   230   240   242   252   254   268   270   284
##  [2,]   162   176   226   230   240   242   252   254   268   270   284   286
##  [3,]   226   230   240   242   252   254   268   270   284   286    NA    NA
##  [4,]   230   240   242   252   254   268   270   284   286    NA    NA    NA
##  [5,]   242   252   254   268   270   284   286    NA    NA    NA    NA    NA
##  [6,]   252   254   268   270   284   286    NA    NA    NA    NA    NA    NA
##  [7,]   254   268   270   284   286    NA    NA    NA    NA    NA    NA    NA
##  [8,]   254   268   270    NA    NA    NA    NA    NA    NA    NA    NA    NA
##  [9,]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## [10,]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## [11,]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## [12,]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
```

```
## [13,]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## [14,]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## [15,]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
## [16,]    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
##       [,26]
##  [1,]   286
##  [2,]    NA
##  [3,]    NA
##  [4,]    NA
##  [5,]    NA
##  [6,]    NA
##  [7,]    NA
##  [8,]    NA
##  [9,]    NA
## [10,]    NA
## [11,]    NA
## [12,]    NA
## [13,]    NA
## [14,]    NA
## [15,]    NA
## [16,]    NA
```

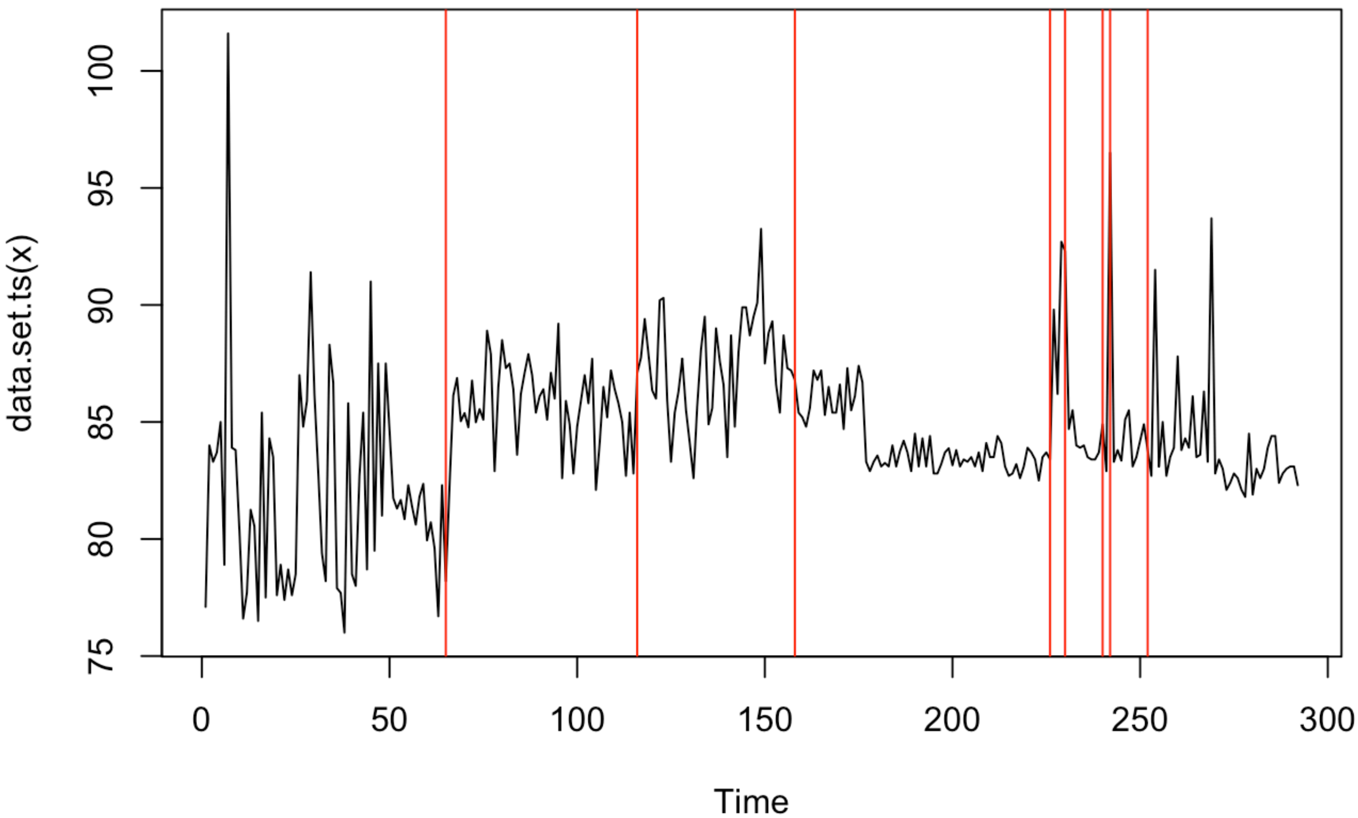
```
pen.value.full(v1.crops)
```

```
##  [1]  2.783750  2.879143  2.974764  3.156478  3.166249  3.593606  4.386950
##  [8]  5.108777  5.325320  5.755618  7.638868 10.218877 12.373445 14.486327
## [15] 21.367756 58.242264
```

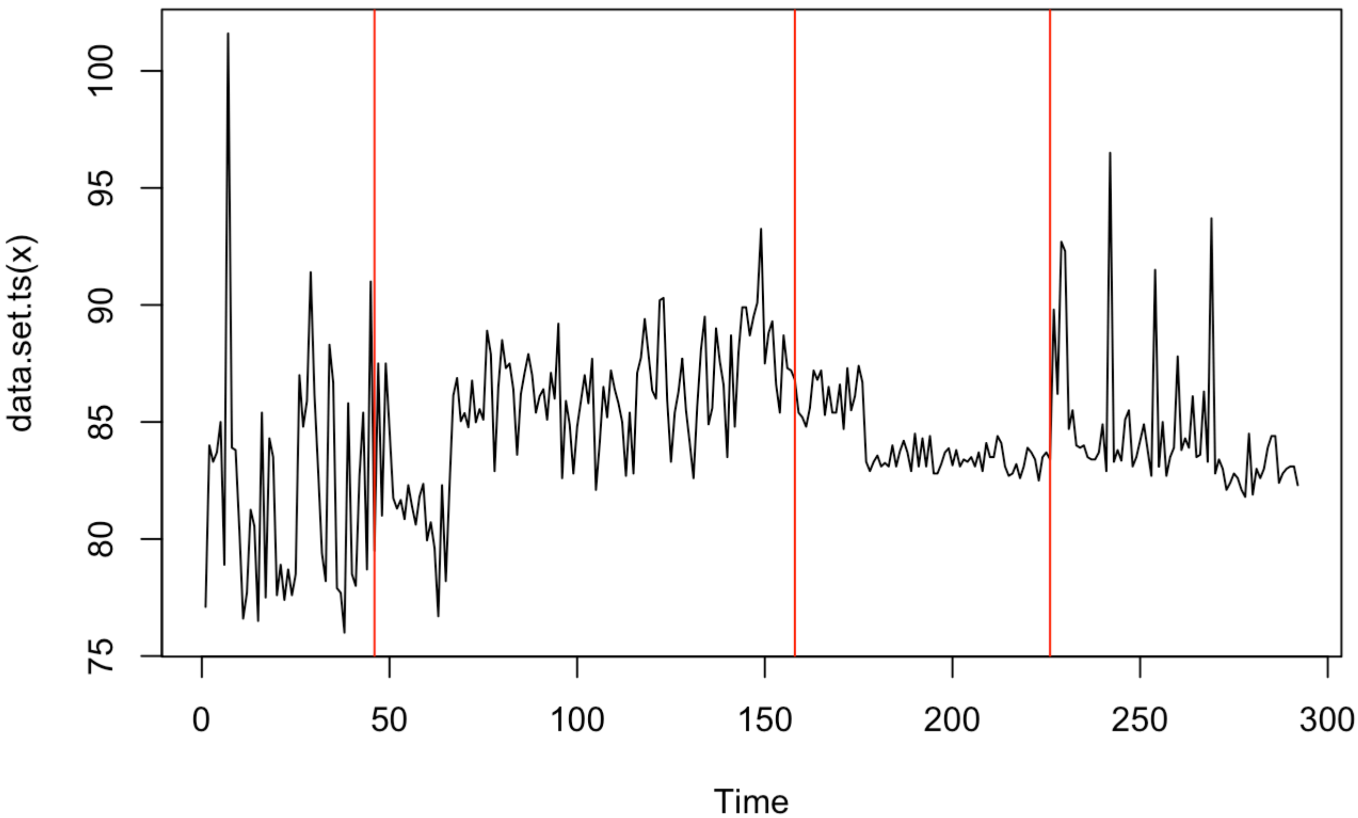
```
plot(v1.crops,diagnostic=TRUE)
```




```
plot(v1.crops,ncpts=8)
```




```
plot(v1.crops,ncpts=3)
```



#18,16,12,11,8,7,6,3,1,0 possible change points The ideal choice will be 3 and 8 segments according to the elbow plot.

My initial estimate of using four segments did not match the optimum results given by the CROPS platform. CROPS advised three and eight parts instead, indicating that my earlier option may not have been the best. The recommended numbers, three and eight, were more logical and fit better with the optimization criteria and goals after a comprehensive study of the CROPS data. This emphasizes the need of making informed decisions based on data-driven optimization, as well as the importance of incorporating data insights into decision-making processes.

When runtime data does not follow a normal distribution, the usefulness of changepoint detection algorithms based on normality assumptions is in doubt. It is recommended to look into alternate approaches that are more in line with the actual distribution of the data. Given the variety of data sources, specific statistical procedures may be more successful. For example, given the non-normal distribution of the runtime data, approaches built for distributions such as gamma or exponential might result in more meaningful and accurate changepoint identification. As a result, it is critical to select a technique that is well-suited to the unique properties of the runtime data, ensuring more reliable changepoint recognition.