Robust Fpop : A package to detect changepoints in the Presence of Outliers using the Biweight, L1 and Huber loss.

Summary

Here we illustrate how use the robseg package implementing the the approach described in the following arXiv paper[1] available at : https://arxiv.org/abs/1609.07363.

Install the package from github

You should first download the source code available at https://github.com/guillemr/robust-fpop. In R you can do this using the devtools package:

```
library(devtools)
```

```
##      usethis
```

```
install_github("guillemr/robust-fpop")
```

```
## Skipping install of 'robseg' from a github remote, the SHA1 (ce49c26a) has not changed since last ins
##   Use `force = TRUE` to force installation
```

Load the package

You can then load the package as follow and set some parameters for Rmd.

```
require(robseg)
```

```
##      robseg
```

```
knitr::opts_chunk$set(fig.width=11, fig.height=7)
```

Simulated data

In this Rmarkdown file we will illustrate the robseg function for the biweight, L1, Huber and L2 loss. As an example we will consider the simulation made in [2] using a student noise rather than a Gaussian noise.

```
source("Simulation.R")
i <- 1     ## there are 6 scenarios we take the first one
dfree <- 6 ## degree of freedom of the Student noise

## we recover the info of the first scenario
Ktrue   <- Simu[[i]]$Ktrue
bkptrue <- as.integer( Simu[[i]]$bkpPage29[-c(1, Ktrue+1)] )
signaltrue <- Simu[[i]]$signal
sigmatrue  <- Simu[[i]]$sigma

## we simulate one profile
set.seed(1)
x.data <- signaltrue + rt(n=length(signaltrue), df=dfree)*sigmatrue
```

We estimate the variance using successive differences and mad as follow:

```
est.sd <- mad(diff(x.data)/sqrt(2))
```

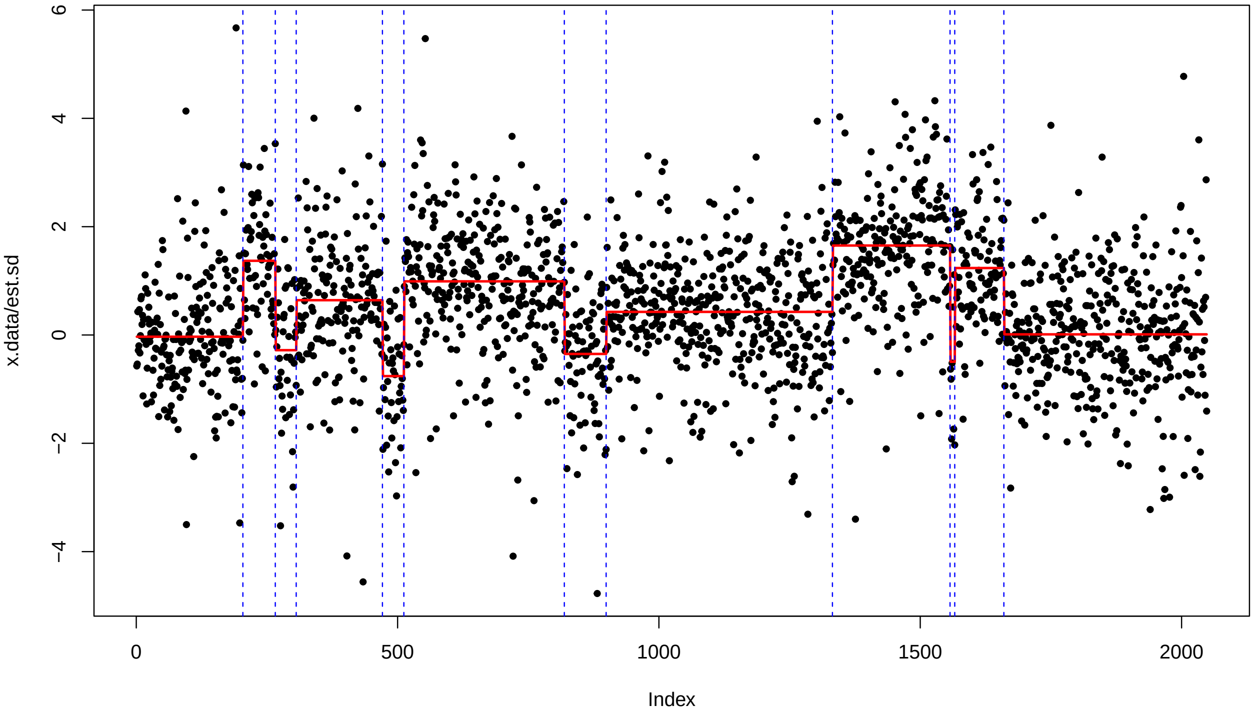
In the following we illustrate how to run Robust Fpop for the Biweight, Huber, L1 and L2 losses.

Robust Fpop with the Biweight loss

Here we ran Robust Fpop with the biweight loss. We set the penalty to $\beta = 2\log(n)$ and the threshold parameter to $K = 3$.

```
## run dynamic programming
res.ou <- Rob_seg.std(x = x.data/est.sd,
                      loss = "Outlier",
                      lambda = 2*log(length(x.data)),
                      lthreshold=3)
## estimated changepoints
cpt <- res.ou$t.est[-length(res.ou$t.est)]

## simple ploting of changes and smoothed profile
plot(x.data/est.sd, pch=20, col="black")
lines(res.ou$smt, col="red", lwd=2)
abline(v=cpt, lty=2, col="blue")
```



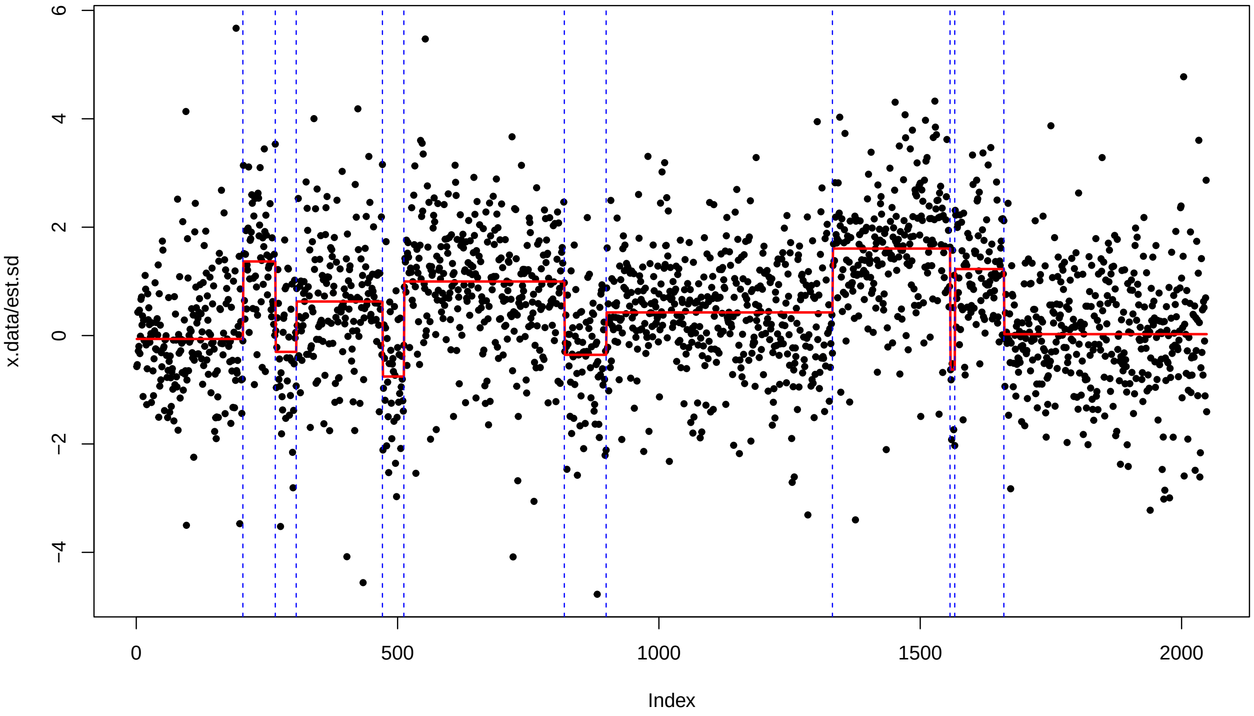
Robust Fpop with the Huber loss

We now run Robust Fpop with the Huber loss fixing the penalty to $\beta = 1.4\log(n)$ and the threshold parameter to 1.345.

```
## run dynamic programming
res.hu <- Rob_seg.std(x = x.data/est.sd,
                      loss = "Huber",
                      lambda = 1.4*log(length(x.data)),
                      lthreshold = 1.345)
```

```
## estimated changepoints
cpt <- res.hu$t.est[-length(res.hu$t.est)]

## simple ploting of changes and smoothed profile
plot(x.data/est.sd, pch=20, col="black")
lines(res.hu$smt, col="red", lwd=2)
abline(v=cpt, lty=2, col="blue")
```



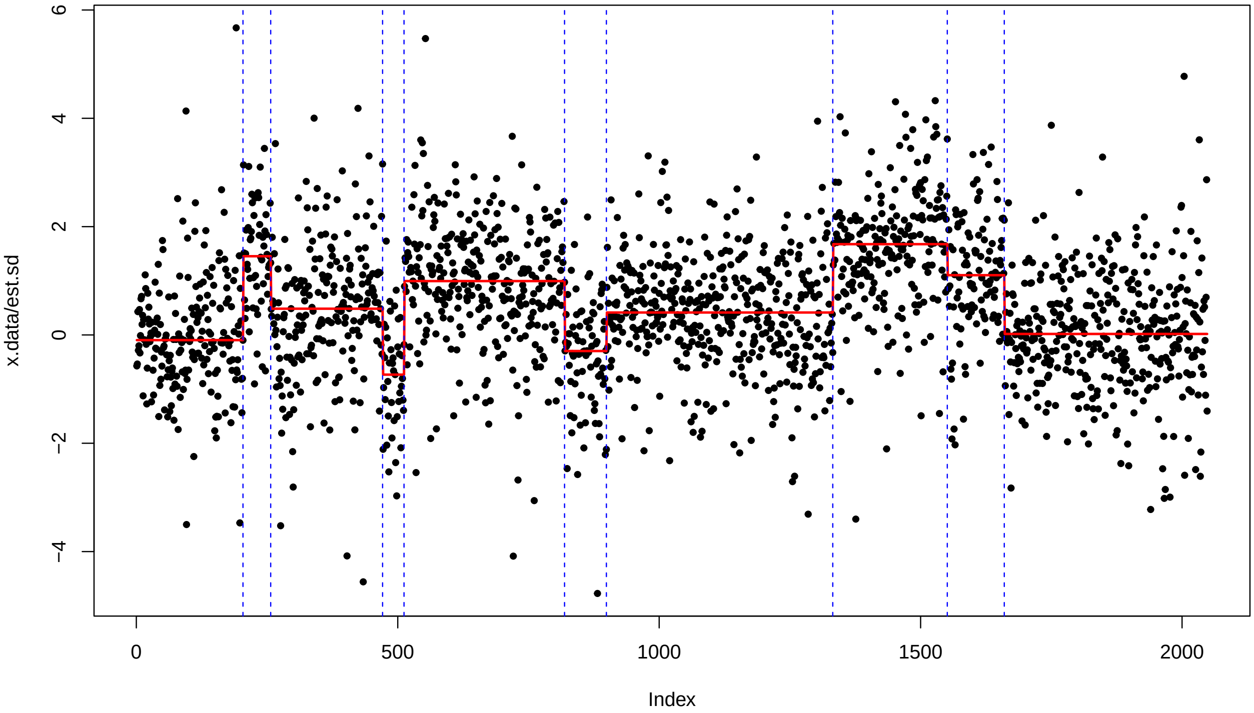
Robust Fpop with L1 loss

We now run Robust Fpop with L1 loss fixing the penalty to $\beta = \log(n)$. In this example on segment is not detected : $[1556 - 1597]$.

```
## run dynamic programming
res.l1 <- Rob_seg.std(x = x.data/est.sd,
                      loss = "L1",
                      lambda = log(length(x.data)))

## estimated changepoints
cpt <- res.l1$t.est[-length(res.l1$t.est)]

## simple ploting of changes and smoothed profile
plot(x.data/est.sd, pch=20, col="black")
lines(res.l1$smt, col="red", lwd=2)
abline(v=cpt, lty=2, col="blue")
```



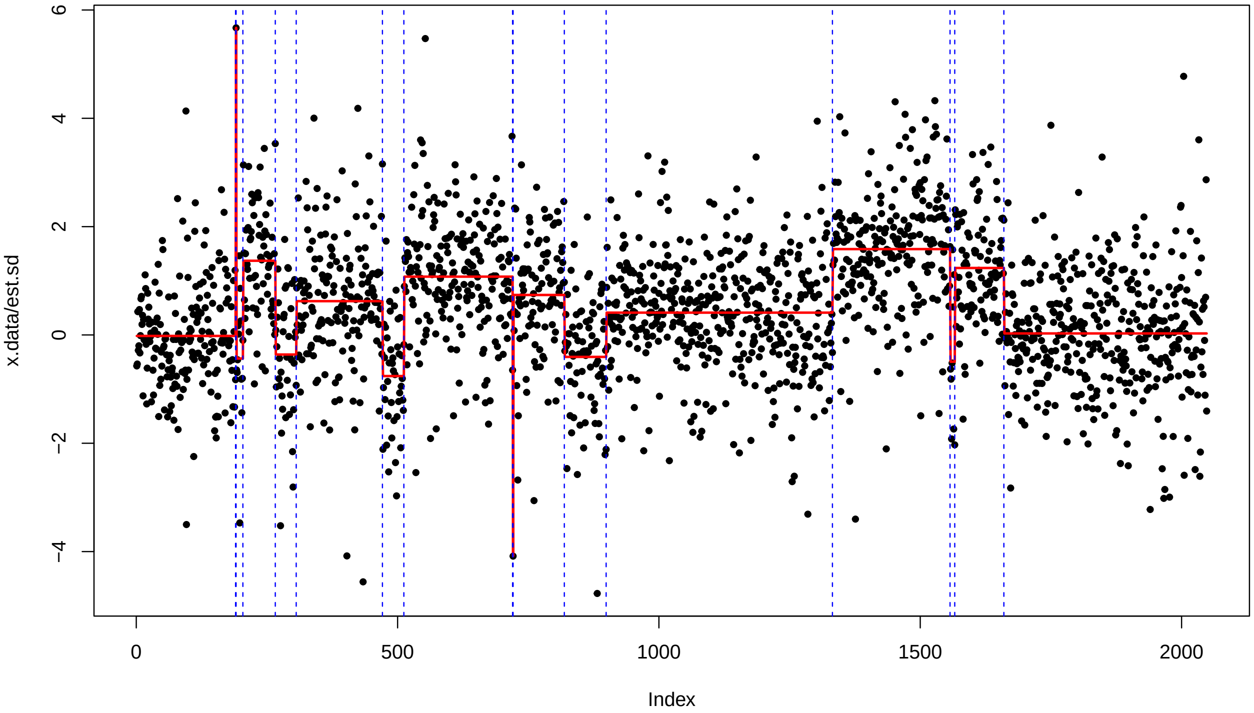
Fpop with the L2 loss

We now ran Fpop with the L2 loss [1] fixing the penalty $\beta = 2\log(n)$. In this example, some outlier data points are detected as segments.

```
## run dynamic programming
res.l2 <- Rob_seg.std(x = x.data/est.sd,
                      loss = "L2",
                      lambda=2*log(length(x.data)))

## estimated changepoints
cpt <- res.l2$t.est[-length(res.l2$t.est)]

## simple ploting of changes and smoothed profile
plot(x.data/est.sd, pch=20, col="black")
lines(res.l2$smt, col="red", lwd=2)
abline(v=cpt, lty=2, col="blue")
```



Some references

[1] Fearnhead, Paul and Rigaill, Guillem. "Changepoint Detection in the Presence of Outliers" arXiv:1609.07363

[2] Maidstone, Robert, et al. "On optimal multiple changepoint algorithms for large data." Statistics and Computing (2014): 1-15.)