Muscle4TS: MUltivariate Sparse CLustering for Extremes in Times Series

In this notebook we provide an heuristic adaptation of the MUSCLE algorithm [3] to heavy tailed time series. The aim here is to detect components that take large values together and one after the other during extreme events that can cover multiple timeframes. The new method is then applied on three financial datasets.

It is part of a research project done by Benoît Reber under the joint supervision of Prof. Wintenberger (LPSM), Assoc. Prof. Meyer (IMAG) and Assoc. Prof. Buritica (AgroParisTech)

What it is about: Heavy tailed time series and MUSCLE algorithm

Extreme value theory is interested in providing simple models for the tails of heavy tailed random variables. In the one dimensional non-negative case, the idea boils down to approximate the distribution above a high threshold by a Pareto distribution whose parameters will be estimated using the few data available above the above-mentioned threshold. In extreme value theory we therefore face a bias-variance trade-offrelated to the choice of the threshold. If we lower the threshold, more data is used implying a reduction of the variance, but since the assumption that the distribution follows a Pareto distribution might not be reasonable anymore with the lower threshold, the new data points might induce an increase in the bias of the estimators.

Dealing with time series, we say that an extreme event happens in a block of successive timestamps of a time series if the l^p norm of the block is “large”. As a consequence an extreme event can be caused by either a single large value at some timestamps or multiple smaller values at multiple timeframes. We provide here minimum details and refer to the recent monograph by Mikosch & Wintenberger [1].

We consider stationary $\mathbb{R}^d$ valued time series $X_n, n \in \mathbb{N}$ that are heavy tailed in the sense that they are regularly varying with index α (RV_α). Writing $X_{[m,m+h]}$ for $\{X_m, X_{m+1}, \cdots, X_{m+h}\}$, $||\cdot||_p$ for the usual l^p norm, this means that for any $m \geq 0$ and $h \geq 0$ we have, as $t \to \infty$, the weak convergence

$$\mathbb{P}\left(t^{-1}X_{[m,m+h]} \in \cdot \mid ||X_{[m,m+h]}||_p > t\right) \xrightarrow{w} \mathbb{P}\left(Y\, Q^{(p)}(h) \in \cdot\right)$$

where Y and $Q^{(p)}(h)$ are independent, Y is $Pareto(\alpha)$, $||Q^{(p)}(h)||_p = 1$. We call $Q^{(p)}(h)$ the spectral component of $X_{[0,h]}$. Under appropriate mixing and anti-clustering conditions, there exists a process, called spectral tail process $Q_n^{(p)}, n \in \mathbb{N}$ such that $Q^{(p)}(h) = (Q_n^{(p)})_{n \leq h}$

The MUSCLE algorithm has been design to deal with the iid setting for which the definition of Regular Variation is the same as above, with the difference that one must take $h = 0$ and $p = 1$. The algorithm aims at learning a set that contains the support of the spectral component. Therefore it identifies coordinates of the time series (X_t) that can take large values simultaneously while taking into account time (some extreme event might be defined by large value taken by some coordinate at time t and large value taken by another coordinate at time $t + 1$).

For β a subset of $\{1, 2, \ldots, d\}$, let $C_\beta = \left\{x \in \mathbb{R}^d \text{ s.t. } x_i > 0 \text{ for } i \in \beta\ \right\}$ a maybe extremal direction. MUSCLE aims at finding all the sets C_β that are contained in no larger $C_{\tilde{\beta}}, \tilde{\beta} \subset \beta$ —they are said to be maximal—such that

$$\mathbb{P}\left(Q^{(p)}(h) \in C_\beta\right) > 0 \quad \text{and} \quad \mathbb{P}\left(Q^{(p)}(h) \in C_{\tilde{\beta}}\right) = 0 \ \text{ for all } \tilde{\beta} \supsetneq \beta.$$

As a consequence, the algorithm provides a sparse representation of the support of $Q^{(p)}(h)$ made up of vectors whose coordinates only take the values 0 and 1. Estimating the probability of $Q^{(p)}(h)$ to belong to each set C_β found by the algorithm, we get a simple sparse generative model that could be used to sample new extreme points. However this method has not been assessed yet. We refer to [2] for details about sparse regular variation used as theoretical foundation for MSUCLE and to [3] for the definition of the algorithm and the empirical evaluation of its performance in the iid case (no timeseries).

As a byproduct, MUSCLE also provides the extremal directions that are not maximal and for each extremal direction, the probability for an extremal event to happen in this direction is estimated.

It is important to note that the penalization used in the MUSCLE algorithm by its authors does not correspond to the one they deduced from the model and its assumptions in [3]. Doing so, it works better that the latter but the reasons why are not yet clearly understood.

Our method

Under usual mixing and anti-clustering assumptions on the $\mathbb{R}^d$ valued timeseries $X_n, n \in \mathbb{N}$, we expect some asymptotic independence between the blocks when the length of the blocks goes to infinity at the same time as the number of blocks goes to infinity. In this case we would have at the limit that the blocks behave like iid random vectors.

Let b denote the length of the blocks we are considering. We choose to deal with sliding block rather than consecutive blocks, expecting to get the same result at the limit. The idea is simply that if b is "large enough" relative to the duration l of an extremal event, then the extreme event will be detected approximatively The extreme event will be detected $b - 1 + 1$ times while we could get false detections at most $2 * (l - 1)$ times when the extreme event is not entirely in the block. We write "at most" here because as an extreme event is detected using the l^p norm of the block, if not all timestamps with large values are covered by the block, it is possible that no extreme event is detected. Under appropriate assumptions, e.g. to keep things simple we can assume b goes to infinity and make the single big jump assumption, it is expected that everything works as desired when the size of the dataset goes to infinity.

To keep things short, the idea is simply to apply the original MUSCLE algorithm to the vectors that represent the blocks. These vectors are simply built by concatenating the b vectors $X_n, X_{n+1}, \dots, X_{n+b}$ for each n such that it is well defined. In the original MUSCLE algorithm, the parameter that is chosen using an AIC method is the number k of vectors considered to be extreme (as a byproduct, MUSCLE provides a way to choose such k, which is a difficult problem in extreme value theory). Here we will consider *prop* the share of extreme data instead of k. This is motivated by the fact that we use a non-constant block length b, whence the number of vectors is not always the same and k is not relevant anymore.

The choice of the block length b will depend on the share of data considered to be extreme in the same way as it is done in the numerical experiments parts of [3], i.e. setting $b = (\mathit{prop})^{-1/2}$ where *prop* denotes the proportion of extreme data.

Doing so, for an heavy-tailed time series, our new method Muscle4TS provides:

(1) an automatic choice of the block length b that will be used in a second notebook in the estimation of the form

$$f_p^Q = \mathbb{E}(f_p(YQ^{(p)}))$$

for suitable l^p-continuity functions $f_p : l^p \to \mathbb{R}$ invariant to shift operator (we refer to [4,5] for details about such statistics).

(2) an automatic choice of the share of blocks to be considered extremes

(3) a sparse representation of the spectral tail process and its support

Experiments on synthetic data

We try the new method on simple auto-regressive AR and moving average MA models for which the temporal patterns of extremal events are known.

```r
library(tsExtremes)
library(VGAM)
library(rlang)
library(latex2exp) # to insert latex symbols in the plots
source("muscle_TS.R")
```

Heavy-tailed AR(1)

First we consider a regularly varying AR(1) model $X_t, t \in \mathbb{R}$, defined using heavy tailed innovations, as the solution to the equation

$$X_{t+1} = \phi X_t + Z_t \,, \qquad t \in \mathbb{R}$$

where (Z_t) are iid and regularly varying with tail index $\alpha > 0$ (we choose $\alpha = 1$) and $\phi \in [0, 1)$.

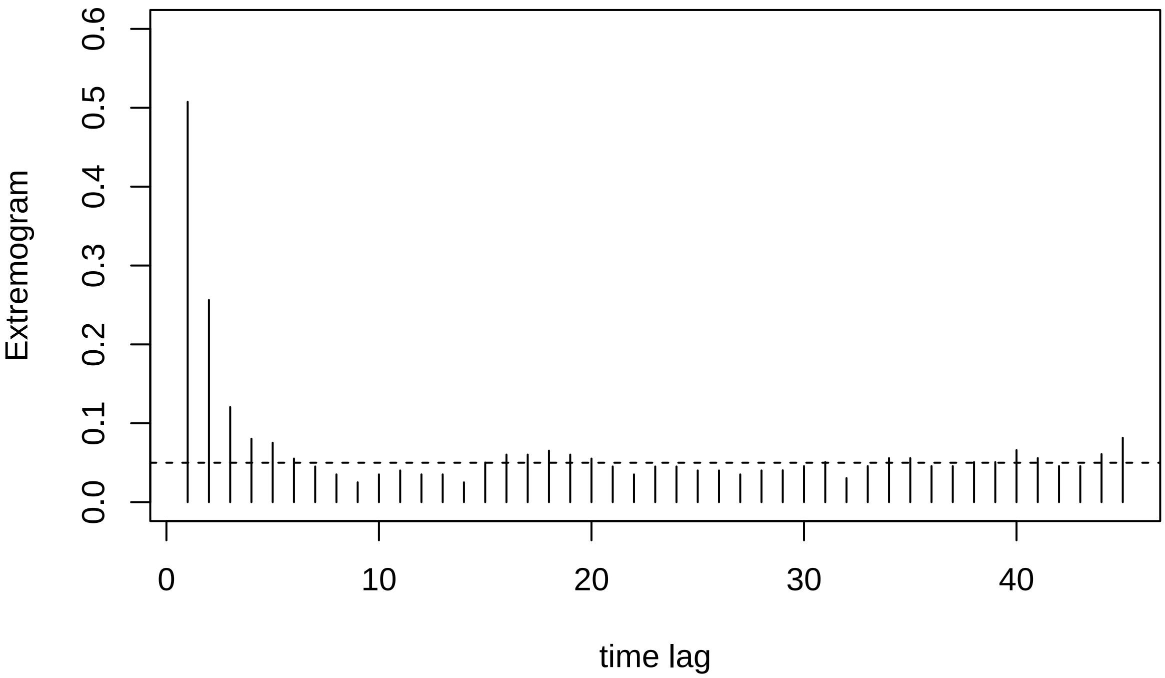
```r
set.seed(2025)
n       <- 4000
d       <- 1
phi     <- 0.5
alpha   <- 1
sample  <- ARm(n,phi, Z.gen = function(n) VGAM::rpareto(n, shape = 1/alpha) )
```

The extremogram have been introduced by Davis and Mikosch (2009, [6]) as the process χ_t defined, for every $t > 0$,by

$$\chi_t = \lim_{x \to \infty} \mathbb{P}(|X_t| > x \mid |X_0| > x).$$

It is a variant of the usual correlogram designed to deal with temporal dependance in extreme values.

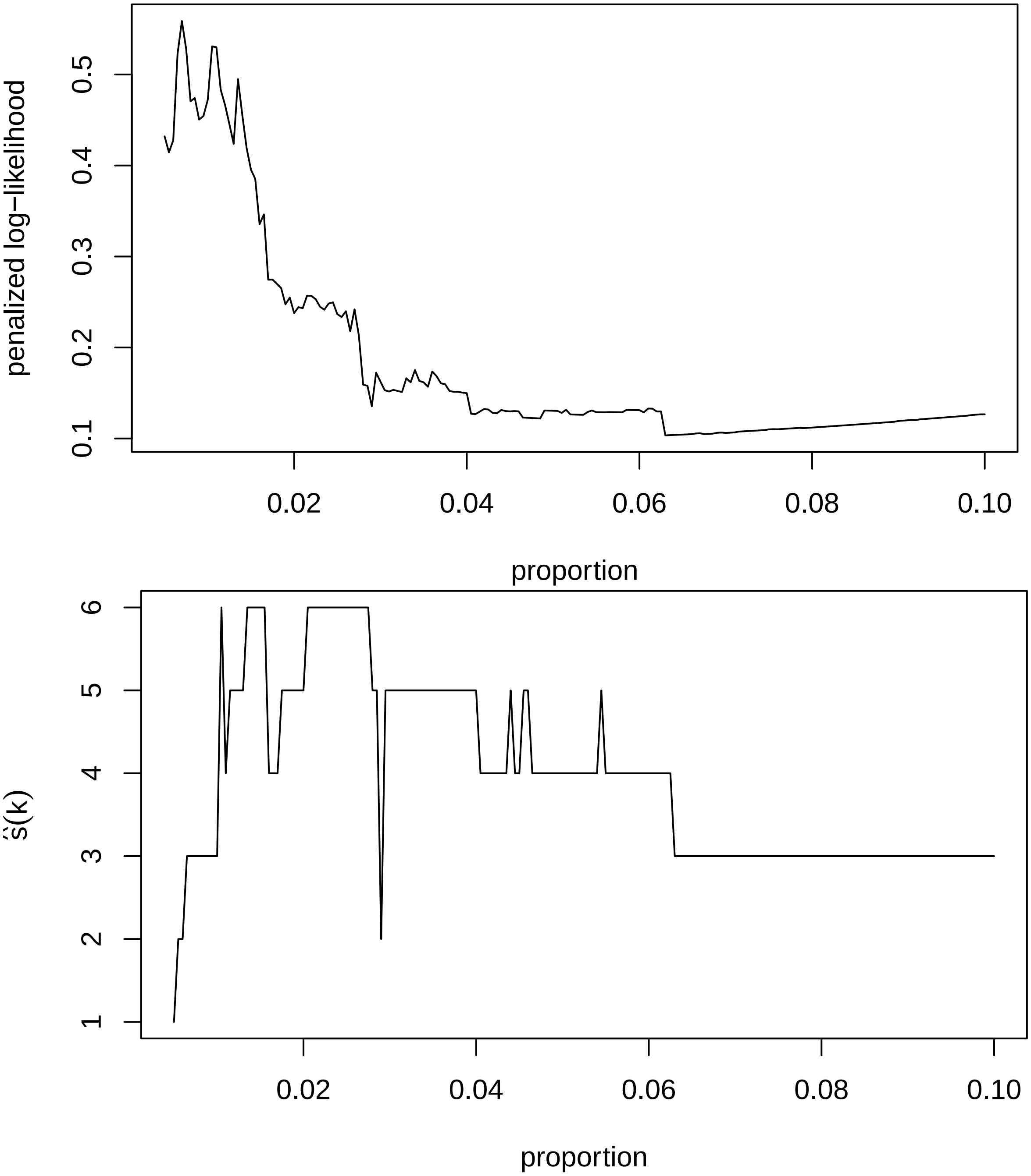
```r
extremogram(sample)
```



```
## function (sample, maxlag = 10, q)
## {
##     sapply(1:maxlag, function(k) mean(sample[(which(sample >
##         q) + k)] > q, na.rm = T))
## }
## <bytecode: 0x13bc837a0>
## <environment: namespace:tsExtremes>
```

When X_t takes an extreme value for some t, there is a significant probability that X_s for s in $t + \{1, 2, 3\}$ to be extreme too. As a consequence we expect to detect extreme events in the timeseries that consist in 1 to 4 large values. We now apply our method Muscle4TS.

```
prop <- seq(0.005, 0.10, by = 0.0005)
results <- muscle_clusters_TS(sample, prop,phi=2)
```



```
s_hat <- results[[4]]
b <- results[[3]]
extr_dir <- results[[1]][1:(d*b), ]
```

```
results[-1]
```

```
## [[1]]
##  prop
## 0.063
##
## [[2]]
## b
## 3
##
## [[3]]
## s
## 3
##
## [[4]]
## [1] 0.4056225 0.3092369 0.2851406
```

```
extr_dir
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    1
## [2,]    1    0    1
## [3,]    1    0    0
```

Our method only detected extrem events that are made up of 1 to 3 large values and estimate that among extreme events, a succession of 3 large values is slighty more likely to happen than a succession of 2 large values or a single large values.

Heavy-tailed MA(2)

We build a simple $\mathbb{R}^3$ valued timeseries $X_t, t \in \mathbb{R}$ whose coordinates are moving average MA(2) models depending on the same iid noise $Z_t, t \in \mathbb{R}$ which is $Pareto(\alpha)$ distributed (we choose again $\alpha = 1$). $X_t, t \in \mathbb{R}$ is defined as a solution to the equation

$$X_t(1) = Z_t \qquad X_t(2) = Z_t + \varphi_{2,1} Z_{t-1} \qquad X_t(3) = \ Z_t + \varphi_{3,1} Z_{t-1} + \varphi_{3,3} Z_{t-2}$$

where $\varphi_{i,j} \in \mathbb{R}$. We choose $\varphi_{i,j} = 1$.

```
m_dependence_shifted_3d <- function(n,m=2,phi1=rep(1,m-1),phi2=rep(1,m),alpha=1){
  Z <- matrix(round((1/runif(n+m))**(1/alpha), 2))
  X1 <- matrix(Z[(1+m):length(Z)])
  X2 <- matrix(Z[(1+m):length(Z)])
  X3 <- matrix(Z[(1+m):length(Z)])
  for (i in 1:(m-1)){
    X2 <- X2 + phi1[i] * Z[(1+m-i):(m-i+n)] #* Z[(i+1):(i+1+n-1)]
  }
  for (i in 1:(m)){
    X3 <- X3 + phi2[i] * Z[(1+m-i):(m-i+n)]
  }
  dim(X1) <- c(1,n)
  dim(X2) <- c(1,n)
  dim(X3) <- c(1,n)
  X <- rbind(X1,X2)
  X <- rbind(X,X3)
  return(X)
}
```

Relying on this we build a $\mathbb{R}^6$ valued timeseries $X_t, t \in \mathbb{R}$ whose first 3 and last 3 coordinates are independent MA(2) processes defined as above. For each $\mathbb{R}^3$ process we expect to detect mainly the pattern

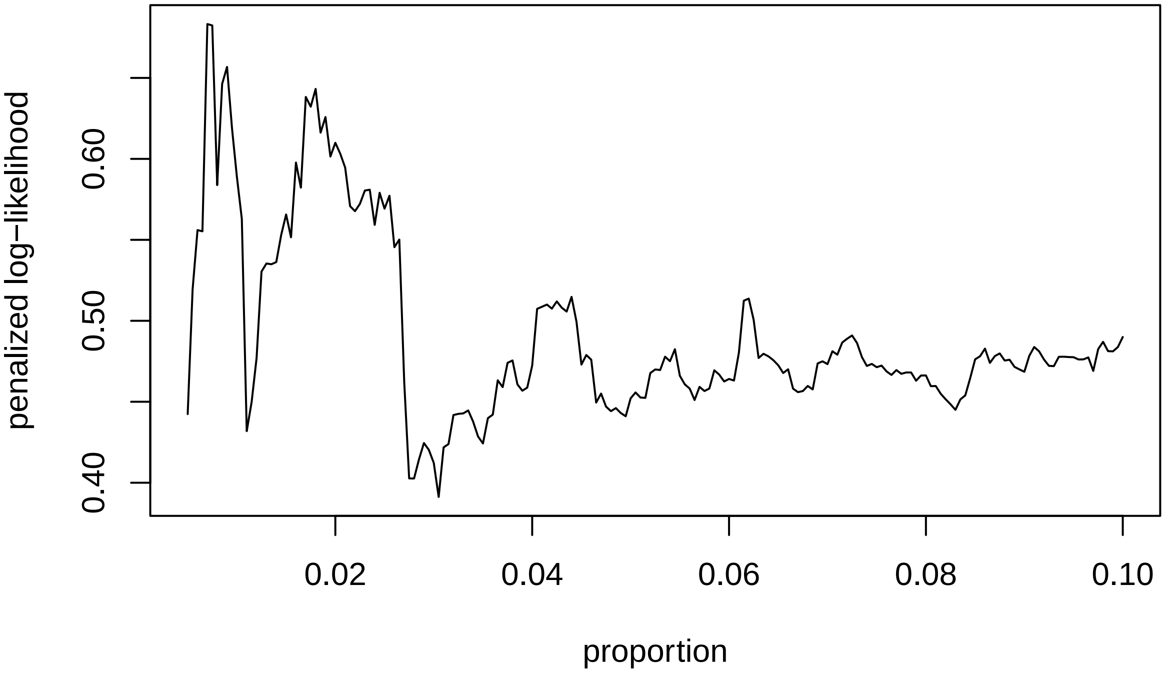
$$(1,1,1),(0,1,1),(0,0,1)$$

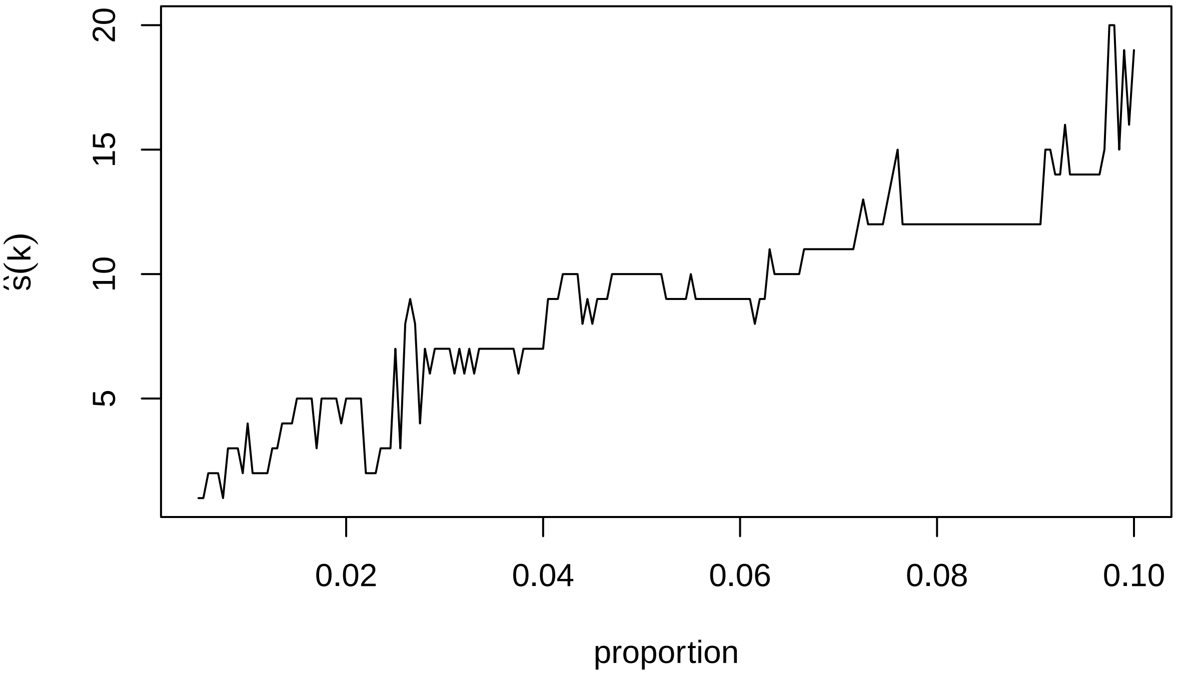
with possibly the last one or the two last ones missing. As the two $\mathbb{R}^3$-valued time series are independant we do not expect to see extreme events made up of extreme events on both processes at the same time.

```
set.seed(2025)
n       <- 4000
d <- 6
alpha   <- 1
sample  <- rbind( m_dependence_shifted_3d(n),  m_dependence_shifted_3d(n))
```

We now apply our Muscle4TS method.

```
prop <- seq(0.005, 0.10, by = 0.0005)
results <- muscle_clusters_TS(sample, prop,phi=2)
```





```
s_hat <- results[[4]]
b <- results[[3]]
extr_dir <- results[[1]][1:(d*b), ]
```

```
results[-1]
```

```
## [[1]]
##   prop
## 0.0305
##
## [[2]]
## b
## 5
##
## [[3]]
## s
## 7
##
## [[4]]
## [1] 0.34736842 0.30526316 0.09473684 0.09473684 0.08421053 0.04210526 0.03157895
```

```
extr_dir
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7]
##  [1,]    1    0    1    1    0    0    0
##  [2,]    1    0    1    1    0    0    0
##  [3,]    1    0    1    1    0    0    0
##  [4,]    0    1    0    1    1    0    1
##  [5,]    0    1    0    1    1    1    1
##  [6,]    0    1    0    1    1    1    1
##  [7,]    0    0    0    1    0    0    0
##  [8,]    1    0    1    1    0    0    0
##  [9,]    1    0    1    1    0    0    0
## [10,]    0    0    0    1    0    0    0
## [11,]    0    1    0    1    1    0    0
## [12,]    0    1    0    1    1    1    0
## [13,]    0    0    0    1    0    0    0
## [14,]    0    0    0    1    0    0    0
## [15,]    1    0    0    1    0    0    0
## [16,]    0    0    0    1    0    0    0
## [17,]    0    0    0    1    0    0    0
## [18,]    0    1    0    1    0    0    0
## [19,]    0    0    0    1    0    0    0
## [20,]    0    0    0    1    0    0    0
## [21,]    0    0    0    1    0    0    0
## [22,]    0    0    0    1    0    0    0
## [23,]    0    0    0    1    0    0    0
## [24,]    0    0    0    1    0    0    0
## [25,]    0    0    0    1    0    0    0
## [26,]    0    0    0    1    0    0    0
## [27,]    0    0    0    1    0    0    0
## [28,]    0    0    0    1    0    0    0
## [29,]    0    0    0    1    0    0    0
## [30,]    0    0    0    1    0    0    0
```

Although we detect unexpected extremal directions, the expected ones are detected. Moreover, among extremal direction, the weight of the unexpected ones ; for example the extremal event consisting of large values everywhere is associated to a weight of only 10%.

Conclusion

The new Muscle4TS method effectively captures the pattern of extremal events in simple heavy-tailed AR and MA models where the innovation noise is heavy-tailed and follows a Pareto(α) distribution. Moreover, in both cases we not that the minimum of the penalized log-likelihood is attained for a value $prop^*$ of *prop* such that $\hat{s}$ seen as a function of *prop* is "almost" constant, hence continuous, in some neighbourhood of $prop^*$. This is reassuring as it implies that the optimal value s^* does not depend, to some extent, on the sequence of values we test for *prop*.

This is a significant finding, as it addresses a key challenge in modeling time series with extreme value behavior. However, this preliminary work lacks a comparative analysis with other established methods and requires further numerical validation. The theoretical properties of the Muscle4TS approach also need to be rigorously established to ensure its robustness and reliability. Future research will focus on benchmarking Muscle4TS against existing state-of-the-art models for heavy-tailed time series and developing a comprehensive theoretical framework for the method.

Experiments on real datasets

Daily returns by industry sectors in the USA

We apply the new Muscle4TS method to analyze daily returns from 49 US industry portfolios. The goal is to detect groups of industries that experience large, consecutive returns during extreme market events. The dataset, sourced from the Ken French data library, will be restricted to covers only the period from 1971 to 2021 in order to avoid dealing with sectors that did not exist prior to 1971

The portfolios are constructed based on four-digit SIC codes, with each NYSE, AMEX, and NASDAQ stock assigned to an industry at the end of June each year. Daily returns are then calculated from July of that year to June of the following year. It is important to note that the data has undergone several revisions over time to improve accuracy and consistency. These revisions include updates to the market return calculation, changes in how stocks are handled after delisting, and revisions to the computation of operating profitability.

The 49 industry portfolios daily returns dataset was taken from https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html. Details on the industries can be found on the file 'details_industries.txt'.

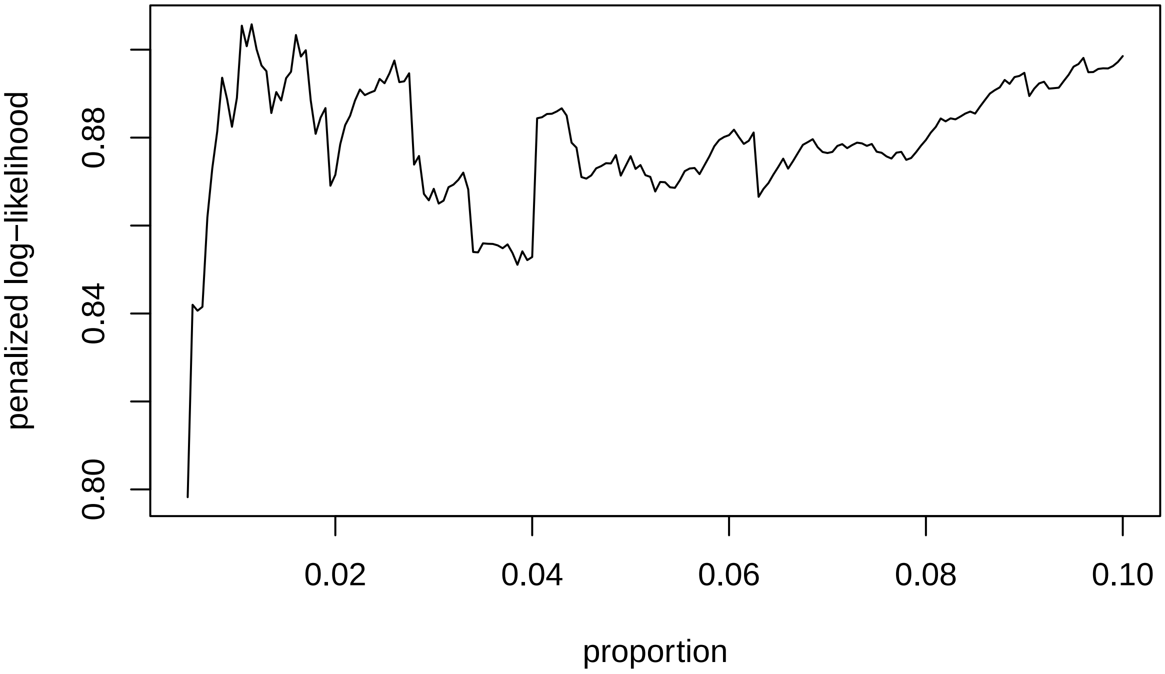
```
# We load the dataset
portfolios <-read.delim("49_Industry_Portfolios_Daily.txt", sep="", dec=".", header = TRUE)
industries <- colnames(portfolios)
# We restrict the time period to 1970-2019
date_min <- which(rownames(portfolios)==19700102)
date_max <- which(rownames(portfolios)==20191231)
mydata <- t(portfolios[date_min:date_max, ])
```

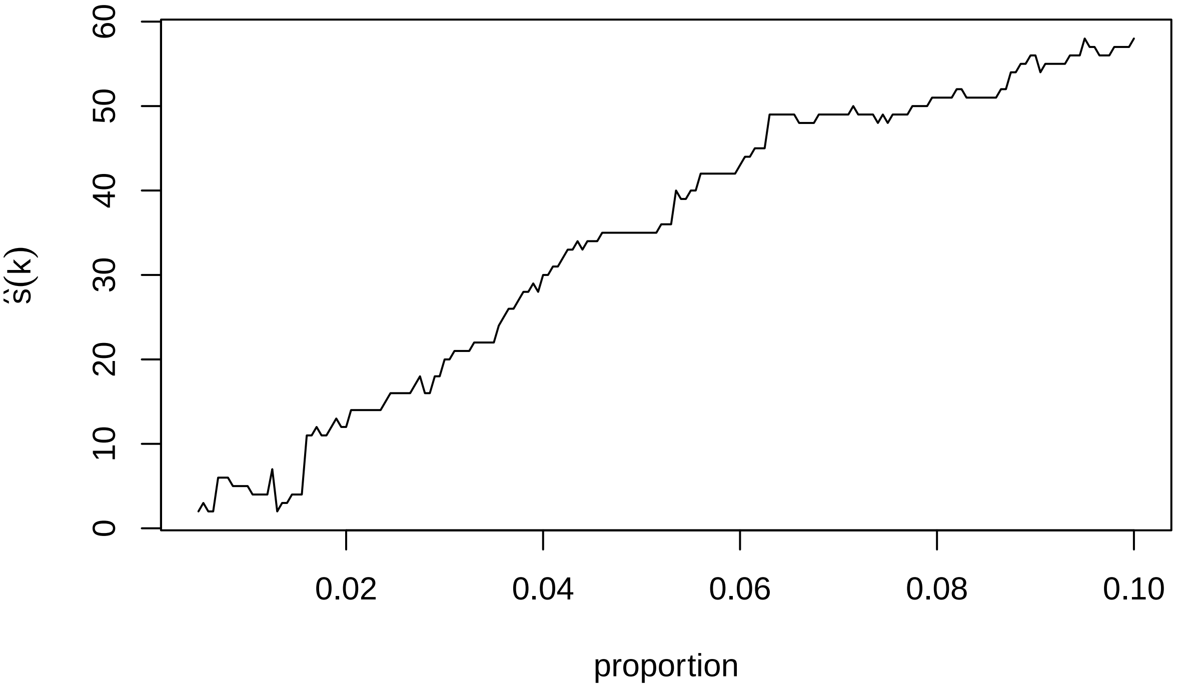
We transform the data to highlight the extremes using the tail index estimated using Hill estimator.

```
norm_mydata<-apply(abs(mydata),2,sum)
alpha_mydata <- 1/(alphaestimator(norm_mydata,
                                  plot=F,
                                  k1 = floor(length(norm_mydata)^0.8))$xi)
X <- sign(mydata) * abs(mydata)^(alpha_mydata)
d <- nrow(X)
n <- ncol(X)
```

We apply the Muscle4TS method to the dataset.

```
prop <- seq(0.005, 0.10, by = 0.0005)
result <- muscle_TS(X, prop,phi=2)
```





We ignore the first minimum which is likely caused by the lack of data (remember the bias-variance in Extreme Value Theory mentioned earlier).

```
# We choose the minimal value for KL_penalized and the associated values for M, s_hat, prop_hat, etc.
min_above_threshold <- min(which((result[[1]][,1] > 0.01)))
minimum <- min_above_threshold + which.min(result[[1]][min_above_threshold:nrow(result[[1]]),5])
s_hat <- result[[1]][minimum,4]
prop_hat <- result[[1]][minimum,1]
M <- result[[2]][[minimum]]
M <- as.matrix(M[, 1:s_hat]) # we take only the s_hat first columns
b <- result[[1]][minimum,2] # the 'optimal' bloc length
weights <- M[(d*b+1), ]/ sum(M[(d*b+1), ])

directions <- list(M, prop_hat, b, s_hat, weights)

s_hat <- directions[[4]]
b <- directions[[3]]
extr_dir <- directions[[1]][1:(d*b), ] # the faces

if (is.matrix(extr_dir)==FALSE){
  extr_dir <- as.matrix(extr_dir)
}
```

Finally we translate the extremal directions found in terms of industry sectors as follows; Fun -1" "Steel -1" "Gold 2" mean that we detect a pattern with a large negative value for "Fun" and "Steel" at time 1 and a large positive value for "Gold" at time 2. See the file 'details_industries.txt' for details on the industries sector.

```
extr_portfolios <- list()

for (j in 1:s_hat){
  current_dir <- c()
  for (block in 0:(b-1)){
    tmp <- extr_dir[(block * d + 1):((block+1) * d),]
    current_dir <- c( current_dir, paste(industries[ which(tmp[ ,j]!=0) ], (block+1) * tmp[ which(tmp[
  extr_portfolios[[j]] <- current_dir
  }
}

extr_portfolios
```

```
## [[1]]
## [1] "Softw -1"
##
## [[2]]
## [1] "Gold 1"
##
## [[3]]
## [1] "Softw 1"
##
## [[4]]
## [1] "Softw 1" "Softw 2"
##
## [[5]]
## [1] "RlEst 1"
##
## [[6]]
## [1] "Hardw 1"
##
## [[7]]
## [1] "Fun -1"   "Aero -1"  "Guns 1"   "Trans -1"
##
## [[8]]
## [1] "Txtls 1"
##
## [[9]]
## [1] "Txtls 1" "Autos 1" "Banks 1" "RlEst 1"
##
## [[10]]
## [1] "Books 1"
##
## [[11]]
## [1] "Soda -1"
##
## [[12]]
## [1] "Agric 1"
##
## [[13]]
## [1] "Hshld -1" "Gold 1"   "Oil 1"
##
## [[14]]
## [1] "Txtls -1"
##
## [[15]]
## [1] "Fun -1"   "Steel -1" "Gold -2"
##
## [[16]]
## [1] "Coal -1"  "Banks -1" "RlEst -1" "Fin -1"
##
## [[17]]
## [1] "Coal -1" "Steel 2" "ElcEq 2" "Mines 2" "Coal 2"  "RlEst 2"
##
## [[18]]
## [1] "Banks -1" "Fin -1"   "Coal 2"   "Banks 2"  "Fin 2"
##
## [[19]]
## [1] "Softw -1" "Softw 2"
##
## [[20]]
##  [1] "Food -1"  "Soda -1"  "Beer -1"  "Smoke -1" "Toys -1"  "Fun -1"
##  [7] "Books -1" "Hshld -1" "Clths -1" "MedEq -1" "Drugs -1" "Chems -1"
## [13] "Rubbr -1" "Txtls -1" "BldMt -1" "Cnstr -1" "Steel -1" "Mach -1"
## [19] "ElcEq -1" "Autos -1" "Aero -1"  "Guns -1"  "Mines -1" "Oil -1"
## [25] "PerSv -1" "BusSv -1" "Hardw -1" "Softw -1" "Chips -1" "LabEq -1"
## [31] "Paper -1" "Boxes -1" "Whlsl -1" "Rtail -1" "Meals -1" "Insur -1"
## [37] "Other -1"
##
## [[21]]
## [1] "Steel 1" "Mines 1" "Coal 1"  "Oil 1"   "Coal -3"
##
## [[22]]
##  [1] "Coal -1"  "Fun -3"   "Books -3" "Hlth -3"  "Cnstr -3" "Steel -3"
##  [7] "FabPr -3" "Mach -3"  "Autos -3" "Ships -3" "Coal -3"  "PerSv -3"
## [13] "Banks -3" "Insur -3" "RlEst -3" "Fin -3"
##
## [[23]]
## [1] "Steel -1" "Mines -1" "Coal -1"  "Banks -1" "Fin -1"
##
## [[24]]
## [1] "Cnstr 1" "Steel 1" "Banks 1" "Fin 1"   "Other 1" "Txtls 3"
##
## [[25]]
## [1] "Banks 1" "RlEst 1" "Fin 1"
##
## [[26]]
##  [1] "Softw -1" "Fun 3"    "Hlth 3"   "MedEq 3"  "PerSv 3"  "Softw 3"
##  [7] "LabEq 3"  "Whlsl 3"  "Meals 3"  "RlEst 3"  "Other 3"
##
## [[27]]
## [1] "Coal -1" "Gold -3"
##
## [[28]]
## [1] "Mines 1" "Coal 1"
##
## [[29]]
## [1] "Softw 1" "Softw 2" "Softw 3"
```

On the positive side, we detect extremal events that one may have expected like “Txtls 1” “Autos 1” “Banks 1” “RlEst 1” that may happen following good news regarding the purchasing power of the population. On the other hand, some others detected directions seem difficult to interpret ; e.g., for the extremal event “Steel 1” “Mines 1” “Coal 1” “Oil 1” “Coal -3” it is not clear what could cause such pattern, hence this may be a false detection. We lack a well theoreticaly founded alternative method to compare with.

Commodity futures prices

We also try to apply the new Muscle4TS method to a daily time-series of commodity futures prices. The objective is to identify groups of commodities that experience large, simultaneous price changes during periods of market stress. The dataset, sourced from a Kaggle project by Debashish, contains daily futures prices for over 20 different commodities, quoted in Euros. The analysis is restricted to the period from January 4, 2000, to December 31, 2021, to ensure data consistency and avoid missing values.

The dataset includes commodities from six main categories: Energy, Industrial Metals, Precious Metals, Grains, Livestock, and Softs. For this analysis, the "Gasoline" commodity is excluded due to missing values. This curated dataset allows for the study of how different commodity groups respond to extreme market events, providing insights into their interconnectedness.

The dataset was taken from https://www.kaggle.com/datasets/debashish311601/commodity-prices/data.

```r
commo_prices <- read.csv("commodity_futures.csv", header = TRUE)
commo_prices <- commo_prices[, !(names(commo_prices) %in% c("Date", "GASOLINE") ) ][2:5679,]
commodities <- colnames(commo_prices)
n <- nrow(commo_prices)
returns <- 100 *t( log(commo_prices[2:n,]) - log(commo_prices[1:(n-1),]) )
returns[ is.na(returns) ] <- 0
```

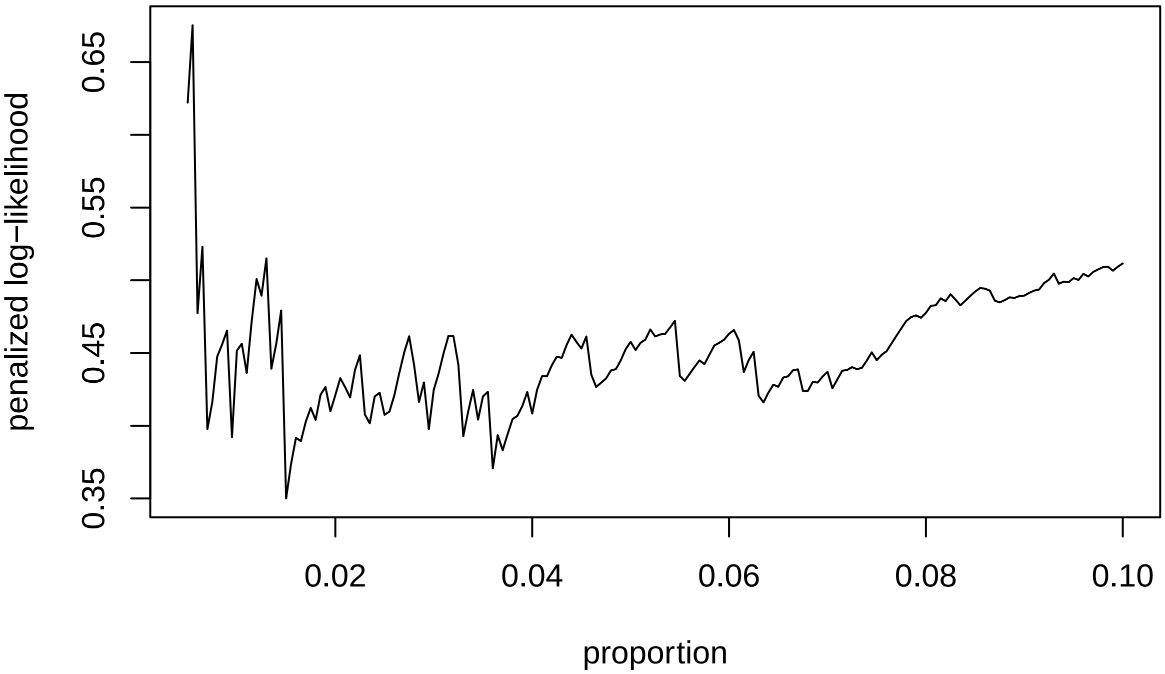
We transform the data to highlight the extremes using the tail index estimated using Hill estimator.

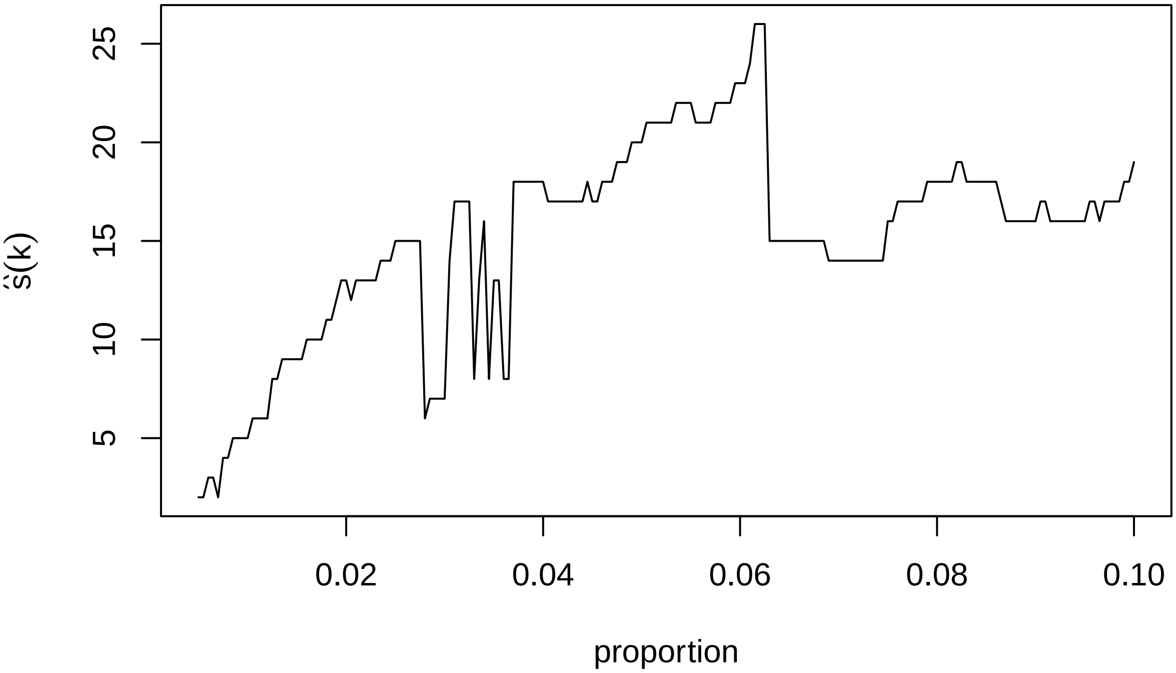
```r
norm_mydata<-apply(abs(returns),2,sum)
alpha_mydata <- 1/(alphaestimator(norm_mydata, plot=F, k1 = floor(length(norm_mydata)^0.8))$xi)

X <- sign(returns) * abs(returns)^(alpha_mydata)
d <- nrow(X)
n <- ncol(X)
```

We apply the Muscle4TS method to the dataset.

```r
prop <- seq(0.005, 0.10, by = 0.0005)
result <- muscle_TS(X, prop,phi=2)
```





```
# We choose the minimal value for KL_penalized and the associated values for M, s_hat, prop_hat, etc.
min_above_threshold <- min(which((result[[1]][,1] > 0.005)))
minimum <- min_above_threshold + which.min(result[[1]][min_above_threshold:nrow(result[[1]]),5])
s_hat <- result[[1]][minimum,4]
prop_hat <- result[[1]][minimum,1]
M <- result[[2]][[minimum]]
M <- as.matrix(M[, 1:s_hat]) # we take only the s_hat first columns
b <- result[[1]][minimum,2] # the 'optimal' bloc length
weights <- M[(d*b+1), ]/ sum(M[(d*b+1), ])

directions <- list(M, prop_hat, b, s_hat, weights)
```

```
s_hat <- directions[[4]]
b <- directions[[3]]
extr_dir <- directions[[1]][1:(d*b), ] # the faces

if (is.matrix(extr_dir)==FALSE){
  extr_dir <- as.matrix(extr_dir)
}
```

Finally we translate the extremal directions found in terms of commodities as follows ; “WTI.CRUDE -1” “BRENT.CRUDE 2” means that we detect a pattern with a large negative value for “WTI.CRUDE” at time 1 and a large positive value for “BRENT.CRUDE” at time 2.

```
extr_commo <- list()

for (j in 1:s_hat){
  current_dir <- c()
  for (block in 0:(b-1)){
    tmp <- extr_dir[(block * d + 1):((block+1) * d),]
    current_dir <- c( current_dir, paste(commodities[ which(tmp[ ,j]!=0) ], (block+1) * tmp[ which(tmp[
  extr_commo[[j]] <- current_dir
  }
}
extr_commo
```

```
## [[1]]
## [1] "NATURAL.GAS 1"
##
## [[2]]
## [1] "WTI.CRUDE -1"
##
## [[3]]
## [1] "WTI.CRUDE 1"
##
## [[4]]
## [1] "ZINC -1"
##
## [[5]]
## [1] "NICKEL -1"
##
## [[6]]
## [1] "CORN -1"
##
## [[7]]
## [1] "LEAN.HOGS -1"
##
## [[8]]
## [1] "WTI.CRUDE -1"   "BRENT.CRUDE -1"
##
## [[9]]
## [1] "ZINC -1"   "NICKEL -1" "ZINC 2"    "NICKEL 2"
```

We mainly detect extreme variations of a single price on a single timstamp. That’s quite surprising since one may have expected energy prices to be somehow linked, possibly with some lag. We only detect the joint variation of the “WTI.CRUDE” and “BRENT.CRUDE” on a single timeframe.

It is possible that we are facing some independence in the extreme regions. This can be achieved between two variables A and B even if (A, B) have a positive probability to take very large values as soon as the tails of at least A or B is heavier than the one of $||(A, B)||\nvdash_{A>s,B>s}$ for arbitrarily large $s > 0$.

1min

Finally we test our method on $1m$ OHLC data downloaded from TradingView. Taking smaller timeframes, more data is available and we hope to detect patterns covering multiple timeframes and involving multiple assets. Indeed, for a small timaframe we can reasonably expect that the extreme variation in an asset B caused by an extreme variation in the asset A can occur at different sampling time.

The dataset contains OHLC for: (1) GC futures gold (2) E-mini S&P 500 (3) ETH/USDT trading pair on Binance (4) BTC/USDT trading pair on Binance

```r
gold <- read.csv("COMEX_DL_GC.csv", header = TRUE, sep = ";")
sp500 <- read.csv("CME_MINI_DL_ES.csv", header = TRUE, sep = ";")
eth <- read.csv("BINANCE_ETHUSDT.csv", header = TRUE, sep = ";")
btc <- read.csv("BINANCE_BTCUSDT.csv", header = TRUE, sep = ";")

dates <- Reduce(intersect, list(gold$time, sp500$time, eth$time, btc$time))

gold <- gold[ gold$time %in% dates, names(gold) != "time"]
sp500 <- sp500[ sp500$time %in% dates, names(sp500) != "time"]
eth <- eth[ eth$time %in% dates, names(eth) != "time"]
btc <- btc[ btc$time %in% dates, names(btc) != "time"]

colnames(gold) <- paste(colnames(gold), " gold")
colnames(sp500) <- paste(colnames(sp500), " sp500")
colnames(eth) <- paste(colnames(eth), " eth")
colnames(btc) <- paste(colnames(btc), " btc")

prices <- Reduce(cbind, list(gold, sp500, eth, btc))
#prices$time <- dates

n <- nrow(prices)
returns <- 100 *t( log(prices[2:n,]) - log(prices[1:(n-1),]) )
returns[ is.na(returns) ] <- 0
assets <- rownames(returns)
```

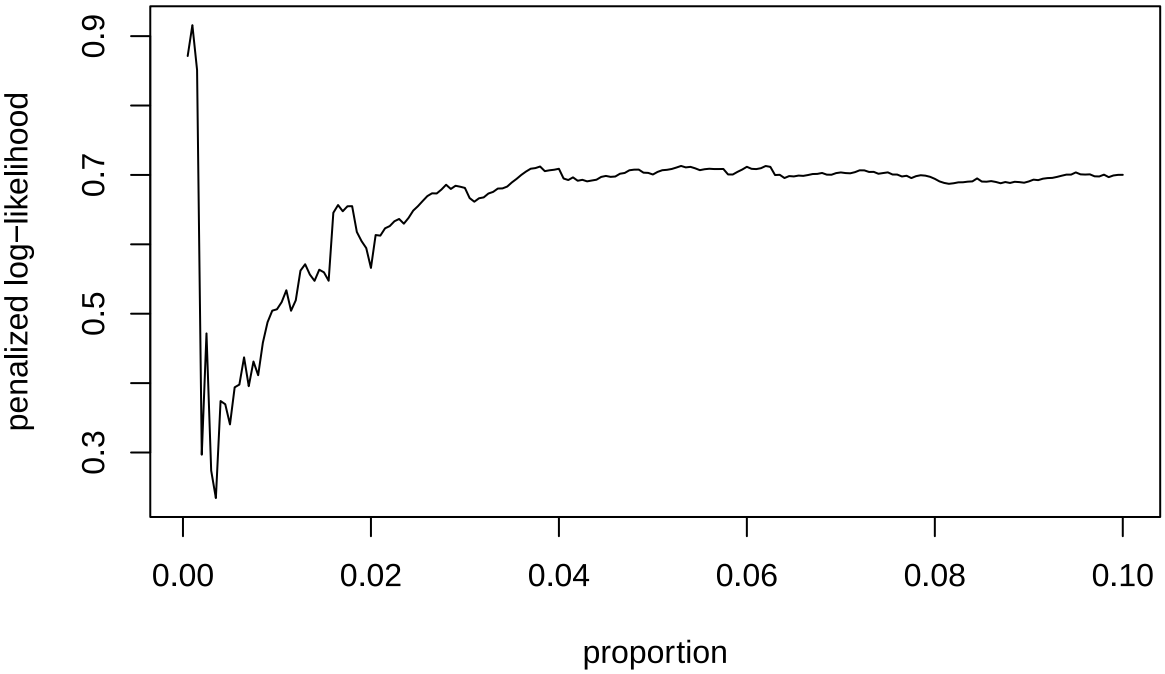
We transform the data to highlight the extremes using the tail index estimated using Hill estimator.

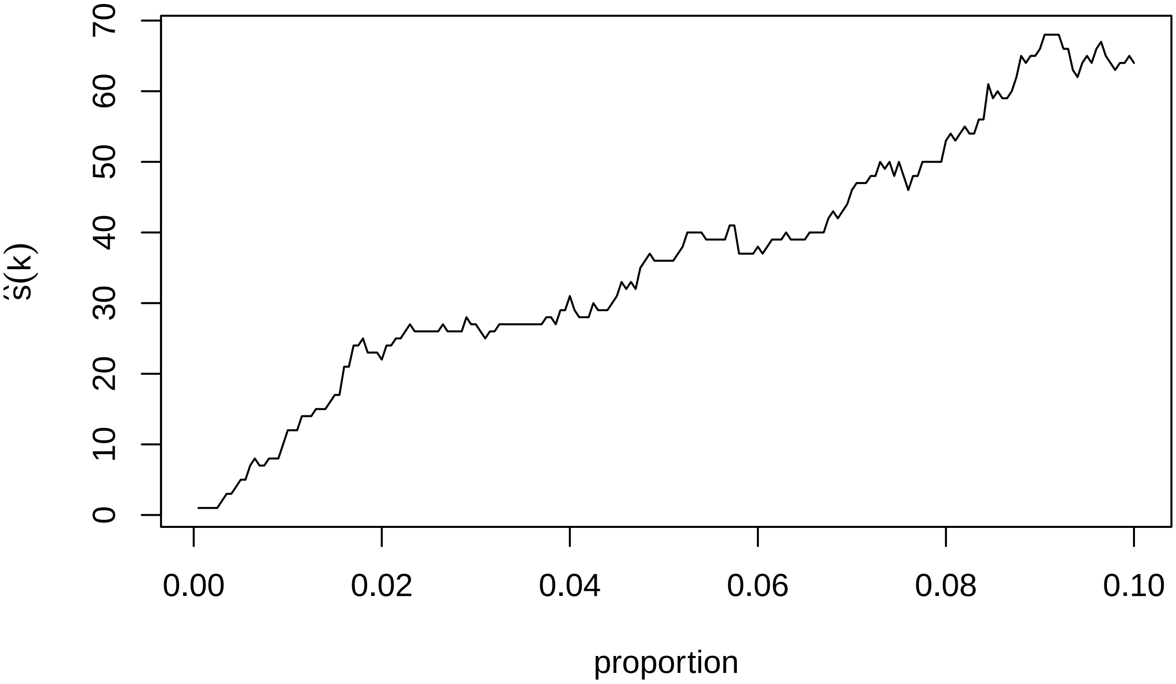
```r
norm_mydata<-apply(abs(returns),2,sum)
alpha_mydata <- 1/(alphaestimator(norm_mydata, plot=F, k1 = floor(length(norm_mydata)^0.8))$xi)

X <- sign(returns) * abs(returns)^(alpha_mydata)
d <- nrow(X)
n <- ncol(X)
```

We apply the Muscle4TS method to the dataset.

```r
prop <- seq(0.0005, 0.10, by = 0.0005)
result <- muscle_TS(X, prop,phi=2)
```





```
# We choose the minimal value for KL_penalized and the associated values for M, s_hat, prop_hat, etc.
min_above_threshold <- min(which((result[[1]][,1] > 0)))
minimum <- min_above_threshold + which.min(result[[1]][min_above_threshold:nrow(result[[1]]),5])
s_hat <- result[[1]][minimum,4]
prop_hat <- result[[1]][minimum,1]
M <- result[[2]][[minimum]]
M <- as.matrix(M[, 1:s_hat]) # we take only the s_hat first columns
b <- result[[1]][minimum,2] # the 'optimal' bloc length
weights <- M[(d*b+1), ]/ sum(M[(d*b+1), ])

directions <- list(M, prop_hat, b, s_hat, weights)
```

```
s_hat <- directions[[4]]
b <- directions[[3]]
extr_dir <- directions[[1]][1:(d*b), ] # the faces

if (is.matrix(extr_dir)==FALSE){
  extr_dir <- as.matrix(extr_dir)
}
```

```
extr_returns <- list()

for (j in 1:s_hat){
  current_dir <- c()
  for (block in 0:(b-1)){
    tmp <- extr_dir[(block * d + 1):((block+1) * d),]
    current_dir <- c( current_dir, paste(assets[ which(tmp[ ,j]!=0) ], (block+1) * tmp[ which(tmp[ ,j]!
  extr_returns[[j]] <- current_dir
  }
}
extr_returns
```

```
## [[1]]
## [1] "open  btc -1"  "high  btc -1"  "low  btc -1"   "close  btc -1"
##
## [[2]]
## [1] "open  eth 1"  "high  eth 1"  "low  eth 1"   "close  eth 1"
##
## [[3]]
## [1] "high  eth 1"  "close  eth 1" "open  eth 2"  "low  eth 2"
```

The method manage to capture dependence and temporal dependence between some of the time series related to the same asset but unfortunately not between times series related to different assets. Moreover, while we cannot question the relevance of the extremal directions found, there is clearly not enough of them considering the strong dependence between different prices of the same asset at the same timestamp.

On the segment $[0.2, 0.4]$, $\hat{s}$ seen as a function of *prop* is almost constant. This prompt us to search for a minimizer with the additionnal constraint of being in this set.

```
# We choose the minimal value for KL_penalized and the associated values for M, s_hat, prop_hat, etc.
min_above_threshold <- min(which((result[[1]][,1] > 0.02)))
minimum <- min_above_threshold + which.min(result[[1]][min_above_threshold:nrow(result[[1]]),5])
s_hat <- result[[1]][minimum,4]
prop_hat <- result[[1]][minimum,1]
M <- result[[2]][[minimum]]
M <- as.matrix(M[, 1:s_hat]) # we take only the s_hat first columns
b <- result[[1]][minimum,2] # the 'optimal' bloc length
weights <- M[(d*b+1), ]/ sum(M[(d*b+1), ])

directions <- list(M, prop_hat, b, s_hat, weights)


s_hat <- directions[[4]]
b <- directions[[3]]
extr_dir <- directions[[1]][1:(d*b), ] # the faces
```

```r
if (is.matrix(extr_dir)==FALSE){
  extr_dir <- as.matrix(extr_dir)
}
```

```r
extr_returns <- list()

for (j in 1:s_hat){
  current_dir <- c()
  for (block in 0:(b-1)){
    tmp <- extr_dir[(block * d + 1):((block+1) * d),]
    current_dir <- c( current_dir, paste(assets[ which(tmp[ ,j]!=0) ], (block+1) * tmp[ which(tmp[ ,j]!
  extr_returns[[j]] <- current_dir
  }
}
extr_returns
```

```
## [[1]]
## [1] "low  eth -1"   "close  eth -1" "open  eth -2"
##
## [[2]]
## [1] "high  eth 1"  "close  eth 1" "open  eth 2"
##
## [[3]]
## [1] "close  eth -1" "open  eth -2"
##
## [[4]]
## [1] "low  eth -1"
##
## [[5]]
## [1] "open  eth 1"
##
## [[6]]
## [1] "low  eth -1"   "close  eth -1"
##
## [[7]]
## [1] "low  eth -1"   "close  eth -1" "open  eth -2"  "high  eth -2"
##
## [[8]]
## [1] "high  eth 1"  "close  eth 1" "open  eth 2"  "low  eth 2"
##
## [[9]]
## [1] "open  btc -1" "low  btc -1"
##
## [[10]]
## [1] "high  eth 1"  "close  eth 1"
##
## [[11]]
## [1] "high  eth 1"
##
## [[12]]
## [1] "low  eth 1"
##
## [[13]]
## [1] "open  eth 1"  "high  eth 1"  "low  eth 1"   "close  eth 1"
```

```
##
## [[14]]
## [1] "high  eth 1"  "low  eth 1"   "close  eth 1"
##
## [[15]]
## [1] "close  eth 1" "open  eth 2"
##
## [[16]]
## [1] "high  eth 1"  "close  eth 1" "open  eth 2"  "high  eth 2"  "low  eth 2"
##
## [[17]]
## [1] "close  eth 1" "open  eth 2"  "low  eth 2"
##
## [[18]]
## [1] "low  eth -1"   "close  eth -1" "open  eth -2"  "close  eth 2"
## [5] "open  eth 3"   "low  eth 3"
##
## [[19]]
## [1] "low  eth -1"   "close  eth -1" "low  btc -1"   "open  eth -2"
## [5] "close  eth 2"  "close  btc 2"  "open  eth 3"   "low  eth 3"
## [9] "open  btc 3"
##
## [[20]]
## [1] "high  eth 1"  "high  eth -2"
##
## [[21]]
## [1] "high  eth 1"   "close  eth 1"  "high  btc 1"   "close  btc 1"
## [5] "open  eth 2"   "close  eth -2" "open  btc 2"   "open  eth -3"
## [9] "high  eth -3"
##
## [[22]]
## [1] "high  eth 1"  "close  eth 1" "open  eth 2"  "low  eth 2"   "close  eth 2"
## [6] "open  eth 3"
##
## [[23]]
## [1] "high  eth 1"  "close  eth 1" "open  eth 2"  "close  eth 3" "open  eth 4"
##
## [[24]]
## [1] "low  gold -1"  "low  eth -1"   "close  eth -1" "open  eth -2"
## [5] "high  eth -2"  "close  eth 3"  "open  eth 4"   "low  eth 4"
##
## [[25]]
##  [1] "low  eth -1"   "close  eth -1" "open  eth -2"  "high  eth -2"
##  [5] "low  eth -2"   "close  eth -2" "low  btc -2"   "close  btc -2"
##  [9] "open  eth -3"  "high  eth -3"  "open  btc -3"  "high  btc -3"
```

On the positive side, we detect way more extremal directions that all seem intuitive. As a consequence, further investigations are required on the choice of the penalization used which is inspired by an heuristic choice done by Meyer and Wintenberger for the original MUSCLE algorithm.

On the negative side, again we fail to detect much dependance structure between extremal events related to different assets. It is quite surprising given that it is known that ETH and BTC's behavior are to some extent linked. Maybe the tails of joint extremes are not heavy enough to be detected (a possible explanation already given in the preceding example). It is also possible that the reaction of one asset to the brutal change in the price of another one may spread accros some timestamps and thus may not be as brutal but the explanation does not seem very intuitive.

Conclusion

Muscle4TS shows significant promise as a new tool for identifying complex extreme event patterns in heavy-tailed time series, particularly those with temporal dependencies. The method's ability to automatically determine key parameters like block length and the proportion of extreme data is a major step forward, as this is a notoriously difficult problem in extreme value theory.

However, the current work is still preliminary. The experiments on real-world data highlight several limitations, including the detection of difficult-to-interpret patterns and a failure to fully capture known dependencies in some cases. The method's theoretical properties and robustness are not yet fully established, and it lacks a direct comparison with other state-of-the-art models for heavy-tailed time series.

Future research should focus on:

(1) Benchmarking: Comparing Muscle4TS against existing methods to understand its strengths and weaknesses more clearly.

(2) Theoretical Framework: Developing a rigorous theoretical foundation for the method to ensure its reliability and to guide the choice of its parameters, especially the penalization term.

(3) Refining the Algorithm: Investigating alternative penalization strategies to improve the detection of subtle or less-prominent extreme event patterns, particularly in datasets with complex interdependencies.

Overall, Muscle4TS provides a valuable new approach, but further development is needed before it can be widely applied with confidence.

References

[1] Mikosch, T., Wintenberger, O. (2024) Extreme Value Theory for Time Series: Models with PowerLaw tails Springer Nature, Switzerland.

[2] Nicolas Meyer and Olivier Wintenberger. Sparse regular variation. Advances in Applied Probability, 53(4):1115–1148, 2021.

[3] Meyer, N. and Wintenberger, O. (2024). Multivariate sparse clustering for extremes., Journal of the American Statistical Association, 119(547):1911–1922.

[4] Buritica, G., Mikosch, T. and Wintenberger, O. (2022) Large deviations of l^p-blocks of regularly varying time series and applications to cluster inference Preprint in arXiv:2106.12822.

[5] Buritica, G. and Wintenberger, O. (2025) On the asymptotics of extremal l^p-blocks cluster inference, Extremes, 1-73, Springer US

[6] Davis, R., and Mikosch, T. (2009) The extremogram: A correlogram for extreme events. Bernoulli, 15, 977–1009