R Script

```
############################################################
#                                                          #
#                         Memoire                          #
#                                                          #
#    Marches, News environnementales & performances        #
#                                                          #
#                                                          #
#         M1 Economie et Finance - Paris Dauphine          #
#                  Prof. Yannick Le Pen                    #
#                                                          #
#                Montagu / Veron-Tarabeux                  #
#                                                          #
############################################################

######### Chargement des librairies #########
#*******************************************
#
# Installation
if (!require("readxl")) install.packages("readxl")
if (!require("xlsx")) install.packages("xlsx")
if (!require("ggplot2")) install.packages("ggplot2")

# Chargement
library("readxl")
library("xlsx")
library("ggplot2")
library("scales") # pour date_breaks
library("zoo")
library('strucchange')
library('stats')
library("naniar")
source('functions.r')

######### Recuperation des donnees #########
#******************************************
#

#------------ WSJ Climate Change News Index : CCN.xlsx ----------

CCN = as.data.frame(read_excel("CCN.xlsx"))
# Conversion de CCN en time serie
CCN = ts(CCN$wsj, frequency=12,start=c(1984,1))
# Decoupage CCN pour recuperer autant de donnees que ENV2
CCN = window(CCN,start=c(2010,1),end=c(2017,6))

# Y a t-il de la saisonnalite ?
dec= decompose(CCN)
png('dec_CCN.png')
plot(dec)
dev.off()

# Stationnairte de CCN
png('ACF_PACF_CCN.png')
par(mfrow=c(2,1))
acf(CCN)
pacf(CCN)
dev.off()
# => autocorrelogrammes representatifs d'une serie stationnaire


#------------ Environmental Score News Index : ENV2.xlsx ----------
```

```
ENV2 = as.data.frame(read_excel("ENV2_monthly.xlsx"))

#Representation graphique de l'ENV2
x11(width=20,height=10);
ggplot(ENV2, aes(x=seq(as.Date("2010-01-01"),
                       length.out = length(ENV2$green_score),by="1 month"))) +
  geom_line(aes(y=ENV2$green_score), color = "steelblue") +
  xlab("Time") + ylab("ENV2 value") +
  ggtitle("Environmental Score News Index") +
  scale_x_date(date_labels="%Y",date_breaks = '1 year')
ggsave('ENV2.png')

# Stationnairte de l'ENV2
png('ACF_PACF_ENV2.png')
par(mfrow=c(2,1))
acf(ENV2$green_score)
pacf(ENV2$green_score)
dev.off()
# => autocorrelogrammes representatifs d'une serie stationnaire

decENV2 = decompose(ts(ENV2$green_score,start=c(2010,1),frequency = 12))
png('dec_ENV2.png')
plot(decENV2)
dev.off()

#—————— Comparaison des deux indices ——————

ENV2 = ts(read_excel("ENV2_monthly.xlsx"), start=c(2010,1), frequency=12)[,2]
ENV2 = window(ENV2,end=c(2017,6))

toPlot = data.frame('month' = seq(as.Date("2010-01-01"),
                                  length.out = length(ENV2),by="1 month"),
                    'CCN' = CCN,
                    'ENV2' = ENV2)

# Representation graphique lineaire
x11(width=20,height=10);ggplot(toPlot, aes(x=month)) +
  geom_line(aes(y=ENV2), color = "steelblue") +
  xlab("Time") + ylab("ENV2 value") +
  scale_x_date(date_labels = "%m-%Y") +
  ggtitle("ENV2 vs CCN (2010 - 2018)") +
  geom_line(aes(y=CCN*100), color = "purple") +
  scale_x_date(date_labels="%Y",date_breaks = '1 year') +
  scale_y_continuous(
    name = "ENV2 value",
    sec.axis = sec_axis(~./100,name="CCN value")) +
  theme(
    axis.title.y = element_text(color = 'steelblue'),
    axis.title.y.right = element_text(color = 'purple'))
ggsave('CCNxENV2.png')

# Representation graphique nuage de points
x11();ggplot(toPlot, aes(x=CCN,y=ENV2)) +
  geom_point(alpha=0.75) +
  labs(x='CCN',y='ENV2',title='CCN x ENV2') +
  geom_smooth(method=lm)
ggsave('CCNxENV2_scatter.png')

# Statistiques descriptives
statsIndex = data.frame(means=c(mean(CCN),mean(ENV2)),
                        median=c(median(CCN),median(ENV2)),
                        max = c(max(CCN),max(ENV2)),
                        min = c(min(CCN),min(ENV2)),
                        std = c(sd(CCN),sd(ENV2)),
                        row.names = c("CCN","ENV2"))
```

```
write.xlsx(round(statsIndex,4),'statsIndex.xlsx')

#---------- 5 facteurs de Fama-French ----------

FF <- ts(as.data.frame(read.csv('F-F_5_Factors_US.csv',sep=','))[-1],
         start=c(1990,7),frequency=12)
colnames(FF) = c("Mkt-RF","SMB","HML","RMW","CMA","RF")

# Slice the dataframe to get only from 2010
FF_df = as.data.frame(window(FF,start=c(2010,1)))

x11(width=20,height=10);
ggplot(FF_df,aes(x=seq(as.Date("2010-01-01"),
                       length.out = length(FF_df$RF),by="1␣month"))) +
  geom_line(aes(y=FF_df$`Mkt-RF`, color='blue')) +
  geom_line(aes(y=FF_df$SMB, color='green')) +
  geom_line(aes(y=FF_df$HML, color='red')) +
  geom_line(aes(y=FF_df$RMW, color='purple')) +
  geom_line(aes(y=FF_df$CMA, color='black')) +
  scale_x_date(date_labels="%Y",date_breaks = '1␣year') +
  labs(title="Fama-French␣5␣factors␣:␣evolution␣depuis␣2010",
       x="Months", y="Factor␣values") +
  theme(legend.position = 'bottom') +
  scale_color_discrete(name='Factors',
                       labels=c("Mkt-RF","SMB","HML","RMW","CMA"))

ggsave('FFx5_2010.png')

#---------- 5 portefeuilles (composition fonction des industries) ----------

PFs <- read.csv('5_indus_equi.csv',sep=',')
PFs <- PFs %>% replace_with_na_all(condition = ~.x <= -99)
PFs <- ts(PFs[-1],start=c(1926,7),frequency=12)

colnames(PFs) = c("Comsumer","Manufacturing","High-Tech","Healthcare","Other")

PFs_df = as.data.frame(window(PFs,start=c(2010,1)))

x11(width=20,height=10);
ggplot(PFs_df,aes(x=seq(as.Date("2010-01-01"),
                        length.out = length(PFs_df$Comsumer),by="1␣month"))) +
  geom_line(aes(y=PFs_df$Comsumer, color='blue')) +
  geom_line(aes(y=PFs_df$Manufacturing, color='green')) +
  geom_line(aes(y=PFs_df$`High-Tech`, color='red')) +
  geom_line(aes(y=PFs_df$Healthcare, color='purple')) +
  geom_line(aes(y=PFs_df$Other, color='black')) +
  theme(legend.position = 'bottom') +
  scale_x_date(date_labels = "%Y",breaks = date_breaks("1␣year")) +
  labs(title="5␣industries␣:␣evolution␣des␣rendements␣depuis␣2010",
       x="Months", y="Rendements") +
  scale_color_discrete(name='Industries',
                       labels=c("Comsumer","Manufacturing",
                                "High-Tech","Healthcare","Other"))

ggsave('Industry_2010.png')


# CCN_new (ts) : va de juin 2008 a mai 2018        : mensuel
# ENV2    (df) : va de 2010-01-01 a fin mars 2019 : hebdo
# FF      (ts) : va de 1964 a aujourd'hui         : mensuel
# PFs     (ts) : va de 1926 a aujourd'hui         : mensuel

# donc on ne garde les valeurs que de 01.2010 a 06.2017 avec window
PFs = window(PFs,start=c(2010,1), end=c(2017,6))
```

```
Mkt.RF = window(FF[,'Mkt-RF'],start=c(2010,1), end=c(2017,6))
SMB = window(FF[,'SMB'],start=c(2010,1), end=c(2017,6))
HML = window(FF[,'HML'],start=c(2010,1), end=c(2017,6))
RMW = window(FF[,'RMW'],start=c(2010,1), end=c(2017,6))
CMA = window(FF[,'CMA'],start=c(2010,1), end=c(2017,6))
RF = window(FF[,'RF'],start=c(2010,1), end=c(2017,6))

# Calcul des corr?lations entre les variables explicatives
correl = data.frame(CCN, ENV2, Mkt.RF, SMB, HML, RMW, CMA)
correl = cor(correl)

######### Mod?lisation #########
#------------ Modele simple : marche et CCN ------------
#*
#* On multiplie par 100 pour que ce soit comparable a des returns et au ENV2
CCN = CCN*100

for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model = lm(excessIndus ~ Mkt.RF + CCN)

  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model))
  dfResid = data.frame('W␣stat' = round(resid$statistic,4),
                       'p-value' = round(resid$p.value,4))
  if (indus==1){ # on cree le dataframe des donnees
    resid2factor_CCN = dfResid

  }else{ # sinon on l'incremente
    resid2factor_CCN<-rbind(resid2factor_CCN,dfResid)
  }

  # Recuperation des valeurs arrondies ? 4 d?cimales
  model<-summary(model)
  dfModel<-data.frame('alpha' = getSignificance(1,model),
                      'tStat_alpha' = round(model$coefficients[1,'t␣value'],4),
                      'Std_alpha' = round(model$coefficients[1,'Std.␣Error'],4),
                      'B1_Mkt-RF' = getSignificance(2,model),
                      'tStat_B1' = round(model$coefficients[2,'t␣value'],4),
                      'Std_B1' = round(model$coefficients[2,'Std.␣Error'],4),
                      'B2_CCN' = getSignificance(3,model),
                      'tStat_B2' = round(model$coefficients[3,'t␣value'],4),
                      'Std_B2' = round(model$coefficients[3,'Std.␣Error'],4),
                      'R2' = round(model$adj.r.squared,4),
                      'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
    models2factor_CCN<-dfModel
  }else{ # sinon on l'incremente
    models2factor_CCN<-rbind(models2factor_CCN,dfModel)
  }
}
rownames(models2factor_CCN) = colnames(PFs)
rownames(resid2factor_CCN) <- colnames(PFs)
models2factor_CCN = t(models2factor_CCN)
resid2factor_CCN <- t(resid2factor_CCN)
write.xlsx(models2factor_CCN,'models2factor_CCN.xlsx')
write.xlsx(resid2factor_CCN,'resid2factor_CCN.xlsx')

# Test de Bai et Perron sur Manufacturing & CCNs
excessIndus = PFs[,2] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + CCN, breaks=1)
summary(rupt)

# Test de Bai et Perron sur Healthcare
```

```
excessIndus = PFs[,4] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + CCN, breaks=1)
summary(rupt)

#----------- Modele simple : marche et ENV2 -----------
#*
# meme methode que pour le CCN
for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model = lm(excessIndus ~ Mkt.RF + ENV2)

  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model))
  dfResid = data.frame('W␣stat' = round(resid$statistic,4),
                       'p-value' = round(resid$p.value,4))
  if (indus==1){ # on cree le dataframe des donnees
    resid2factor_ENV2 = dfResid

  }else{ # sinon on l'incremente
    resid2factor_ENV2<-rbind(resid2factor_ENV2,dfResid)
  }

  # Recuperation des valeurs arrondies ? 4 d?cimales
  model<-summary(model)
  dfModel<-data.frame('alpha' = getSignificance(1,model),
                      'tStat_alpha' = round(model$coefficients[1,'t␣value'],4),
                      'Std_alpha' = round(model$coefficients[1,'Std.␣Error'],4),
                      'B1_Mkt-RF' = getSignificance(2,model),
                      'tStat_B1' = round(model$coefficients[2,'t␣value'],4),
                      'Std_B1' = round(model$coefficients[2,'Std.␣Error'],4),
                      'B2_ENV2' = getSignificance(3,model),
                      'tStat_B2' = round(model$coefficients[3,'t␣value'],4),
                      'Std_B2' = round(model$coefficients[3,'Std.␣Error'],4),
                      'R2' = round(model$adj.r.squared,4),
                      'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
    models2factor_ENV2<-dfModel
  }else{ # sinon on l'incremente
    models2factor_ENV2<-rbind(models2factor_ENV2,dfModel)
  }
}
rownames(models2factor_ENV2) = colnames(PFs)
rownames(resid2factor_ENV2) <- colnames(PFs)
models2factor_ENV2 = t(models2factor_ENV2)
resid2factor_ENV2 <- t(resid2factor_ENV2)
write.xlsx(models2factor_ENV2,'models2factor_ENV2.xlsx')
write.xlsx(resid2factor_ENV2,'resid2factor_ENV2.xlsx')

# Test de Bai et Perron sur Manufacturing & ENV2_m
excessIndus = PFs[,2] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + ENV2, breaks=1)
summary(rupt)

# Test de Bai et Perron sur Healthcare
excessIndus = PFs[,4] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + ENV2, breaks=1)
summary(rupt)

#----------- Modele a 5 facteurs -----------
#
for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model = lm(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA)
  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model))
  dfResid = data.frame('W␣stat' = round(resid$statistic ,4),
                       'p-value' = round(resid$p.value ,4))
  if (indus==1){ # on cree le dataframe des donnees
    resid5factor = dfResid

  }else{ # sinon on l'incremente
    resid5factor<-rbind(resid5factor ,dfResid)
  }

  # Recuperation des valeurs arrondies ? 4 d?cimales
  model<-summary(model)
  dfModel<-data.frame('alpha' = getSignificance(1,model),
                      'tStat_alpha' = round(model$coefficients[1,'t␣value'],4),
                      'Std_alpha' = round(model$coefficients[1,'Std.␣Error'],4),
                      'B1_Mkt-RF' = getSignificance(2,model),
                      'tStat_B1' = round(model$coefficients[2,'t␣value'],4),
                      'Std_B1' = round(model$coefficients[2,'Std.␣Error'],4),
                      'B2_SMB' = getSignificance(3,model),
                      'tStat_B2' = round(model$coefficients[3,'t␣value'],4),
                      'Std_B2' = round(model$coefficients[3,'Std.␣Error'],4),
                      'B3_HML' = getSignificance(4,model),
                      'tStat_B3' = round(model$coefficients[4,'t␣value'],4),
                      'Std_B3' = round(model$coefficients[4,'Std.␣Error'],4),
                      'B4_RMW' = getSignificance(5,model),
                      'tStat_B4' = round(model$coefficients[5,'t␣value'],4),
                      'Std_B4' = round(model$coefficients[5,'Std.␣Error'],4),
                      'B5_CMA' = getSignificance(6,model),
                      'tStat_B5' = round(model$coefficients[6,'t␣value'],4),
                      'Std_B5' = round(model$coefficients[6,'Std.␣Error'],4),
                      'R2' = round(model$adj.r.squared,4),
                      'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
    models5factor<-dfModel
  }else{ # sinon on l'incremente
    models5factor<-rbind(models5factor,dfModel)
  }
}
rownames(models5factor) = colnames(PFs)
rownames(resid5factor) <- colnames(PFs)
models5factor = t(models5factor)
resid5factor <- t(resid5factor)
write.xlsx(models5factor, 'models5factor.xlsx')
write.xlsx(resid5factor, 'resid5factor.xlsx')

# Test de Bai et Perron sur Manufacturing
excessIndus = PFs[,2] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA, breaks=1)
summary(rupt)

# Test de Bai et Perron sur Healthcare
excessIndus = PFs[,4] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA, breaks=1)
summary(rupt)

#---------- Modele a 6 facteurs : CCN ----------
#*
# Nous reprenons la meme methode de calcul que pour le 5 facteurs
# en integrant CCNs en variable endogene
for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model6fac = lm(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + CCN)
  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model6fac))
  dfResid = data.frame('W␣stat' = round(resid$statistic ,4),
                       'p-value' = round(resid$p.value ,4))
  if (indus==1){ # on cree le dataframe des donnees
    resid6factor_CCN = dfResid

  }else{ # sinon on l'incremente
    resid6factor_CCN<-rbind(resid6factor_CCN, dfResid)
  }

  # Recuperation des valeurs arrondies ? 4 d?cimales
  model<-summary(model6fac)
  dfModel<-data.frame('alpha' = getSignificance(1,model),
                      'tStat_alpha' = round(model$coefficients[1,'t␣value'],4),
                      'Std_alpha' = round(model$coefficients[1,'Std.␣Error'],4),
                      'B1_Mkt-RF' = getSignificance(2,model),
                      'tStat_B1' = round(model$coefficients[2,'t␣value'],4),
                      'Std_B1' = round(model$coefficients[2,'Std.␣Error'],4),
                      'B2_SMB' = getSignificance(3,model),
                      'tStat_B2' = round(model$coefficients[3,'t␣value'],4),
                      'Std_B2' = round(model$coefficients[3,'Std.␣Error'],4),
                      'B3_HML' = getSignificance(4,model),
                      'tStat_B3' = round(model$coefficients[4,'t␣value'],4),
                      'Std_B3' = round(model$coefficients[4,'Std.␣Error'],4),
                      'B4_RMW' = getSignificance(5,model),
                      'tStat_B4' = round(model$coefficients[5,'t␣value'],4),
                      'Std_B4' = round(model$coefficients[5,'Std.␣Error'],4),
                      'B5_CMA' = getSignificance(6,model),
                      'tStat_B5' = round(model$coefficients[6,'t␣value'],4),
                      'Std_B5' = round(model$coefficients[6,'Std.␣Error'],4),
                      'B6_CCN' = getSignificance(7,model),
                      'tStat_B6' = round(model$coefficients[7,'t␣value'],4),
                      'Std_B6' = round(model$coefficients[7,'Std.␣Error'],4),
                      'R2' = round(model$adj.r.squared,4),
                      'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
    models6factor_CCN<-dfModel
  }else{ # sinon on l'incremente
    models6factor_CCN<-rbind(models6factor_CCN, dfModel)
  }
}
rownames(models6factor_CCN) = colnames(PFs)
rownames(resid6factor_CCN) <- colnames(PFs)
models6factor_CCN = t(models6factor_CCN)
resid6factor_CCN <- t(resid6factor_CCN)
write.xlsx(models6factor_CCN, 'models6factor_CCN.xlsx')
write.xlsx(resid6factor_CCN, 'resid6factor_CCN.xlsx')

# Test de Bai et Perron sur Healthcare
excessIndus = PFs[,4] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + CCN,
                   breaks=1)
summary(rupt)

# Test de Bai Perron sur Manufacturing
excessIndus = PFs[,2] - RF
ruptures = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + CCN,
                       breaks=1)
coef(ruptures)
summary(ruptures)

#------------- Modele a 6 facteurs : ENV2 -------------
#*
```

```
# Nous reprenons la meme methode de calcul que pour le 5 facteurs
# en integrant ENV2_m en variable endogene
for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model6fac = lm(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + ENV2)

  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model6fac))
  dfResid = data.frame('W stat' = round(resid$statistic,4),
                       'p-value' = round(resid$p.value,4))
  if (indus==1){ # on cree le dataframe des donnees
    resid6factor_ENV2 = dfResid

  }else{ # sinon on l'incremente
    resid6factor_ENV2<-rbind(resid6factor_ENV2,dfResid)
  }

  # Recuperation des valeurs arrondies ? 4 d?cimales
  model<-summary(model6fac)
  dfModel<-data.frame('alpha' = getSignificance(1,model),
                      'tStat_alpha' = round(model$coefficients[1,'t value'],4),
                      'Std_alpha' = round(model$coefficients[1,'Std. Error'],4),
                      'B1_Mkt-RF' = getSignificance(2,model),
                      'tStat_B1' = round(model$coefficients[2,'t value'],4),
                      'Std_B1' = round(model$coefficients[2,'Std. Error'],4),
                      'B2_SMB' = getSignificance(3,model),
                      'tStat_B2' = round(model$coefficients[3,'t value'],4),
                      'Std_B2' = round(model$coefficients[3,'Std. Error'],4),
                      'B3_HML' = getSignificance(4,model),
                      'tStat_B3' = round(model$coefficients[4,'t value'],4),
                      'Std_B3' = round(model$coefficients[4,'Std. Error'],4),
                      'B4_RMW' = getSignificance(5,model),
                      'tStat_B4' = round(model$coefficients[5,'t value'],4),
                      'Std_B4' = round(model$coefficients[5,'Std. Error'],4),
                      'B5_CMA' = getSignificance(6,model),
                      'tStat_B5' = round(model$coefficients[6,'t value'],4),
                      'Std_B5' = round(model$coefficients[6,'Std. Error'],4),
                      'B6_ENV2' = getSignificance(7,model),
                      'tStat_B6' = round(model$coefficients[7,'t value'],4),
                      'Std_B6' = round(model$coefficients[7,'Std. Error'],4),
                      'R2' = round(model$adj.r.squared,4),
                      'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
    models6factor_ENV2<-dfModel
  }else{ # sinon on l'incremente
    models6factor_ENV2<-rbind(models6factor_ENV2,dfModel)
  }
}

rownames(models6factor_ENV2) = colnames(PFs)
rownames(resid6factor_ENV2) <- colnames(PFs)
models6factor_ENV2 = t(models6factor_ENV2)
resid6factor_ENV2 <- t(resid6factor_ENV2)
write.xlsx(models6factor_ENV2,'models6factor_ENV2.xlsx')
write.xlsx(resid6factor_ENV2,'resid6factor_ENV2.xlsx')


# Test de Bai et Perron sur Healthcare
excessIndus = PFs[,4] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + ENV2,
                   breaks=1)
summary(rupt)

# Test de Bai Perron sur Manufacturing
```

```
excessIndus = PFs[,2] - RF
ruptures = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + ENV2,
                       breaks=1)
coef(ruptures)
summary(ruptures)

#———————— Modele a 7 facteurs : CCN & ENV2 ————————
#*
# Nous reprenons la meme methode de calcul que pour le 5 facteurs
# en integrant ENV2_m en variable endogene
for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model7fac = lm(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + ENV2 + CCN)

  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model7fac))
  dfResid = data.frame('W␣stat' = round(resid$statistic,4),
                       'p-value' = round(resid$p.value,4))

  if (indus==1){ # on cree le dataframe des donnees
    resid7factor = dfResid

  }else{ # sinon on l'incremente
    resid7factor<-rbind(resid7factor,dfResid)
  }

  # Recuperation des valeurs
  model<-summary(model7fac)
  dfModel <- data.frame('alpha' = getSignificance(1,model),
                        'tStat_alpha' = round(model$coefficients[1,'t␣value'],4),
                        'Std_alpha' = round(model$coefficients[1,'Std.␣Error'],4),
                        'B1_Mkt-RF' = getSignificance(2,model),
                        'tStat_B1' = round(model$coefficients[2,'t␣value'],4),
                        'Std_B1' = round(model$coefficients[2,'Std.␣Error'],4),
                        'B2_SMB' = getSignificance(3,model),
                        'tStat_B2' = round(model$coefficients[3,'t␣value'],4),
                        'Std_B2' = round(model$coefficients[3,'Std.␣Error'],4),
                        'B3_HML' = getSignificance(4,model),
                        'tStat_B3' = round(model$coefficients[4,'t␣value'],4),
                        'Std_B3' = round(model$coefficients[4,'Std.␣Error'],4),
                        'B4_RMW' = getSignificance(5,model),
                        'tStat_B4' = round(model$coefficients[5,'t␣value'],4),
                        'Std_B4' = round(model$coefficients[5,'Std.␣Error'],4),
                        'B5_CMA' = getSignificance(6,model),
                        'tStat_B5' = round(model$coefficients[6,'t␣value'],4),
                        'Std_B5' = round(model$coefficients[6,'Std.␣Error'],4),
                        'B6_ENV2' = getSignificance(7,model),
                        'tStat_B6' = round(model$coefficients[7,'t␣value'],4),
                        'Std_B6' = round(model$coefficients[7,'Std.␣Error'],4),
                        'B7_CCN' = getSignificance(8,model),
                        'tStat_B7' = round(model$coefficients[8,'t␣value'],4),
                        'Std_B7' = round(model$coefficients[8,'Std.␣Error'],4),
                        'R2' = round(model$adj.r.squared,4),
                        'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
    models7factor<-dfModel
  }else{ # sinon on l'incremente
    models7factor<-rbind(models7factor,dfModel)
  }
}

rownames(models7factor) <- colnames(PFs)
rownames(resid7factor) <- colnames(PFs)
models7factor <- t(models7factor)
```

```
resid7factor <- t(resid7factor)
write.xlsx(models7factor, 'models7factor.xlsx')
write.xlsx(resid7factor, 'resid7factor.xlsx')

# Test de Bai et Perron sur Healthcare
excessIndus = PFs[,4] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA +
                   ENV2 + CCN, breaks=1)
summary(rupt)

# Test de Bai Perron sur Manufacturing
excessIndus = PFs[,2] - RF
ruptures = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA +
                       ENV2 + CCN, breaks=1)
coef(ruptures)
summary(ruptures)

#------------ Modele parmis 49 pf : CCN & ENV2 ------------
#*
# On cherche parmis 49 portfeuilles d'industries s'il en existe un ou un des
# deux facteurs est significatif

PFs <- read.csv('49_indus_equi.csv',sep=',')
PFs <- PFs %>% replace_with_na_all(condition = ~.x < -99)
PFs <- window(ts(PFs[-1],start=c(1926,7),frequency=12),
             start=c(2010,1),end=c(2017,6))

for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model = lm(excessIndus ~ Mkt.RF + CCN + ENV2)

  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model))
  dfResid = data.frame('W␣stat' = round(resid$statistic,4),
                       'p-value' = round(resid$p.value,4))
  if (indus==1){ # on cree le dataframe des donnees
    resid49 = dfResid

  }else{ # sinon on l'incremente
    resid49<-rbind(resid49,dfResid)
  }

  # Recuperation des valeurs arrondies ? 4 d?cimales
  model<-summary(model)
  dfModel<-data.frame('alpha' = getSignificance(1,model),
                      'tStat_alpha' = round(model$coefficients[1,'t␣value'],4),
                      'Std_alpha' = round(model$coefficients[1,'Std.␣Error'],4),
                      'B1_Mkt-RF' = getSignificance(2,model),
                      'tStat_B1' = round(model$coefficients[2,'t␣value'],4),
                      'Std_B1' = round(model$coefficients[2,'Std.␣Error'],4),
                      'B2_CCN' = getSignificance(3,model),
                      'tStat_B2' = round(model$coefficients[3,'t␣value'],4),
                      'Std_B2' = round(model$coefficients[3,'Std.␣Error'],4),
                      'B3_ENV2' = getSignificance(4,model),
                      'tStat_B3' = round(model$coefficients[4,'t␣value'],4),
                      'Std_B3' = round(model$coefficients[4,'Std.␣Error'],4),
                      'R2' = round(model$adj.r.squared,4),
                      'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
    models49<-dfModel
  }else{ # sinon on l'incremente
    models49<-rbind(models49,dfModel)
  }
}
```

```
rownames(models49) = colnames(PFs)
rownames(resid49) <- colnames(PFs)

# Enregistrement avec tri en fonction de tstat de CCN
models49ccn <- models49[order(-abs(models49$tStat_B2)),]
models49ccn = t(models49ccn)
write.xlsx(models49ccn, 'models49_CCN.xlsx')

# Enregistrement avec tri en fonction de tstat de ENV2
models49env2 <- models49[order(-abs(models49$tStat_B3)),]
models49env2 = t(models49env2)
write.xlsx(models49env2, 'models49_ENV2.xlsx')

# Enregistrement des residus
resid49 <- t(resid49)
write.xlsx(resid49, 'resid49.xlsx')

#—————— Modele final a 6 facteurs top 3 ENV2 : CCN ——————
#*
# R?cup?ration des s?ries des "Paper","Chems","Ships"
PFs_df <- as.data.frame(PFs)
PFs <- ts(PFs_df[c("Paper","Chems","Ships")], start=c(2010,1),frequency=12)

for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model6facFin = lm(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + ENV2)

  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model6facFin))
  dfResid = data.frame('W␣stat' = round(resid$statistic,4),
                       'p-value' = round(resid$p.value,4))
  if (indus==1){ # on cree le dataframe des donnees
    residFin6factor_ENV2 = dfResid

  }else{ # sinon on l'incremente
    residFin6factor_ENV2<-rbind(residFin6factor_ENV2,dfResid)
  }

  # Recuperation des valeurs arrondies ? 4 d?cimales
  model<-summary(model6facFin)
  dfModel<-data.frame('alpha' = getSignificance(1,model),
                      'tStat_alpha' = round(model$coefficients[1,'t␣value'],4),
                      'Std_alpha' = round(model$coefficients[1,'Std.␣Error'],4),
                      'B1_Mkt-RF' = getSignificance(2,model),
                      'tStat_B1' = round(model$coefficients[2,'t␣value'],4),
                      'Std_B1' = round(model$coefficients[2,'Std.␣Error'],4),
                      'B2_SMB' = getSignificance(3,model),
                      'tStat_B2' = round(model$coefficients[3,'t␣value'],4),
                      'Std_B2' = round(model$coefficients[3,'Std.␣Error'],4),
                      'B3_HML' = getSignificance(4,model),
                      'tStat_B3' = round(model$coefficients[4,'t␣value'],4),
                      'Std_B3' = round(model$coefficients[4,'Std.␣Error'],4),
                      'B4_RMW' = getSignificance(5,model),
                      'tStat_B4' = round(model$coefficients[5,'t␣value'],4),
                      'Std_B4' = round(model$coefficients[5,'Std.␣Error'],4),
                      'B5_CMA' = getSignificance(6,model),
                      'tStat_B5' = round(model$coefficients[6,'t␣value'],4),
                      'Std_B5' = round(model$coefficients[6,'Std.␣Error'],4),
                      'B6_ENV2' = getSignificance(7,model),
                      'tStat_B6' = round(model$coefficients[7,'t␣value'],4),
                      'Std_B6' = round(model$coefficients[7,'Std.␣Error'],4),
                      'R2' = round(model$adj.r.squared,4),
                      'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
```

```
    modelFin6factor_ENV2<-dfModel
  }else{ # sinon on l'incremente
    modelFin6factor_ENV2<-rbind(modelFin6factor_ENV2,dfModel)
  }
}
rownames(modelFin6factor_ENV2) = colnames(PFs)
rownames(residFin6factor_ENV2) <- colnames(PFs)
modelFin6factor_ENV2 = t(modelFin6factor_ENV2)
residFin6factor_ENV2 <- t(residFin6factor_ENV2)
write.xlsx(modelFin6factor_ENV2,'modelFin6factor_ENV2.xlsx')
write.xlsx(residFin6factor_ENV2,'residFin6factor_ENV2.xlsx')

# Test de Bai et Perron sur Paper
excessIndus = PFs[,1] - RF
rupt = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + ENV2,
                   breaks=1)
summary(rupt)

# Test de Bai Perron sur Chems
excessIndus = PFs[,2] - RF
ruptures = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + ENV2,
                       breaks=1)
summary(ruptures)

# Test de Bai Perron sur Ships
excessIndus = PFs[,3] - RF
ruptures = breakpoints(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA + ENV2,
                       breaks=1)
summary(ruptures)

#---------- Modele final a 5 facteurs ----------
#*

for (indus in 1:length(colnames(PFs))){
  excessIndus = PFs[,indus] - RF
  model5facFin = lm(excessIndus ~ Mkt.RF + SMB + HML + RMW + CMA )

  # Test de Shapiro-Wilks sur les residus du modele
  resid = shapiro.test(residuals(model5facFin))
  dfResid = data.frame('W␣stat' = round(resid$statistic,4),
                       'p-value' = round(resid$p.value,4))
  if (indus==1){ # on cree le dataframe des donnees
    residFin5factor = dfResid

  }else{ # sinon on l'incremente
    residFin5factor<-rbind(residFin5factor,dfResid)
  }

  # Recuperation des valeurs arrondies ? 4 d?cimales
  model<-summary(model5facFin)
  dfModel<-data.frame('alpha' = getSignificance(1,model),
                      'tStat_alpha' = round(model$coefficients[1,'t␣value'],4),
                      'Std_alpha' = round(model$coefficients[1,'Std.␣Error'],4),
                      'B1_Mkt-RF' = getSignificance(2,model),
                      'tStat_B1' = round(model$coefficients[2,'t␣value'],4),
                      'Std_B1' = round(model$coefficients[2,'Std.␣Error'],4),
                      'B2_SMB' = getSignificance(3,model),
                      'tStat_B2' = round(model$coefficients[3,'t␣value'],4),
                      'Std_B2' = round(model$coefficients[3,'Std.␣Error'],4),
                      'B3_HML' = getSignificance(4,model),
                      'tStat_B3' = round(model$coefficients[4,'t␣value'],4),
                      'Std_B3' = round(model$coefficients[4,'Std.␣Error'],4),
                      'B4_RMW' = getSignificance(5,model),
                      'tStat_B4' = round(model$coefficients[5,'t␣value'],4),
                      'Std_B4' = round(model$coefficients[5,'Std.␣Error'],4),
                      'B5_CMA' = getSignificance(6,model),
                      'tStat_B5' = round(model$coefficients[6,'t␣value'],4),
                      'Std_B5' = round(model$coefficients[6,'Std.␣Error'],4),
                      'R2' = round(model$adj.r.squared,4),
                      'F_stat' = round(model$fstatistic[1],4))

  if (indus==1){ # on cree le dataframe des donnees
    modelFin5factor<-dfModel
  }else{ # sinon on l'incremente
    modelFin5factor<-rbind(modelFin5factor,dfModel)
  }
}
rownames(modelFin5factor) = colnames(PFs)
rownames(residFin5factor) <- colnames(PFs)
modelFin5factor = t(modelFin5factor)
residFin5factor <- t(residFin5factor)
write.xlsx(modelFin5factor,'modelFin5factor.xlsx')
write.xlsx(residFin5factor,'residFin5factor.xlsx')

#----------- Changement structurel -----------
#*sur le secteur chemicals avec indice ENV2

excessChem = PFs[,2] - RF
# F statistic test sur Chems avec ENV2
Fs = Fstats(excessChem ~ Mkt.RF + SMB + HML + ENV2)
png("FstatsChemicals.png")
plot(Fs, alpha=0.05, main="F-statistics␣(test␣de␣rupture)␣sur␣Chemicals")
dev.off()
sctest(Fs)
# => pas significatif

# Betas glissants : fenetres de 30 mois
fen = 30
# Avec etoiles pour exportation excel
for (i in 1:(length(PFs[,2])-fen+1)){

  model20 = summary(lm(excessChem[i:(i+fen)] ~ Mkt.RF[i:(i+fen)]
                       + SMB[i:(i+fen)] + HML[i:(i+fen)] + RMW[i:(i+fen)]
                       + CMA[i:(i+fen)] + ENV2[i:(i+fen)]))
  dfBetas <- data.frame('alpha' = getSignificance(1,model20),
                        'B_Mkt-RF' = getSignificance(2,model20),
                        'B_SMB' = getSignificance(3,model20),
                        'B_HML' = getSignificance(4,model20),
                        'B_RMW' = getSignificance(5,model20),
                        'B_CMA' = getSignificance(6,model20),
                        'B_ENV2' = getSignificance(7,model20))

  if (i==1){ # on cree le dataframe des donnees
    betasGlissant <- dfBetas
  }else{ # sinon on l'incremente
    betasGlissant<-rbind(betasGlissant,dfBetas)
  }
}
rownames(betasGlissant) = paste(1:61,30:90,sep="␣-␣")
write.xlsx(betasGlissant,'evolBetasChem.xlsx')

# Sans ?toile pour la repr?sentation graphique
for (i in 1:(length(PFs[,2])-fen+1)){

  model20 = summary(lm(excessChem[i:(i+fen)] ~ Mkt.RF[i:(i+fen)]
                       + SMB[i:(i+fen)] + HML[i:(i+fen)] + RMW[i:(i+fen)]
                       + CMA[i:(i+fen)] + ENV2[i:(i+fen)]))
  dfBetas <- data.frame('alpha' = model20$coefficients[1,"Estimate"],
                        'B_Mkt-RF' = model20$coefficients[2,"Estimate"],
                        'B_SMB' = model20$coefficients[3,"Estimate"],
                        'B_HML' = model20$coefficients[4,"Estimate"],
                        'B_RMW' = model20$coefficients[5,"Estimate"],
                        'B_CMA' = model20$coefficients[6,"Estimate"],
                        'B_ENV2' = model20$coefficients[7,"Estimate"])

  if (i==1){ # on cree le dataframe des donnees
    betasGlissant <- dfBetas
  }else{ # sinon on l'incremente
    betasGlissant<-rbind(betasGlissant,dfBetas)
  }
}
rownames(betasGlissant) = paste(1:61,30:90,sep="␣-␣")

# Repr?sentation graphique des betas glissants
x11(width=20,height=10);
ggplot(betasGlissant,aes(x=seq(as.Date("2012-05-01"),
      length.out = length(betasGlissant$B_Mkt.RF),by="1␣month"))) +
  geom_line(aes(y=betasGlissant$B_Mkt.RF, color='B_Mkt.RF')) +
  geom_line(aes(y=betasGlissant$B_SMB, color='B_SMB')) +
  geom_line(aes(y=betasGlissant$B_HML, color='B_HML')) +
  geom_line(aes(y=betasGlissant$B_ENV2/10, color='B_ENV2/10')) +

  labs(title="Betas␣glissants␣sur␣30␣mois␣de␣2010␣?␣2017", x="Time",
       y="Valeur␣des␣oefficients", fill="Betas")
ggsave('BetasGlissants.png')

x11();ggplot(betasGlissant, aes(x=B_HML,y=B_ENV2)) +
  geom_point(alpha=0.75) +
  labs(x='B_HML',y='B_ENV2',title='B_HML␣x␣B_ENV2') +
  geom_smooth(method=lm)
ggsave('B_HMLxB_ENV2.png')
```

Temporary page!

LaTeX was unable to guess the total number of pages correctly. As there was some unprocessed data that should have been added to the final page this extra page has been added to receive it.

If you rerun the document (without altering it) this surplus page will go away, because LaTeX now knows how many pages to expect for this document.