# Détection de rupture dans les séries temporelles

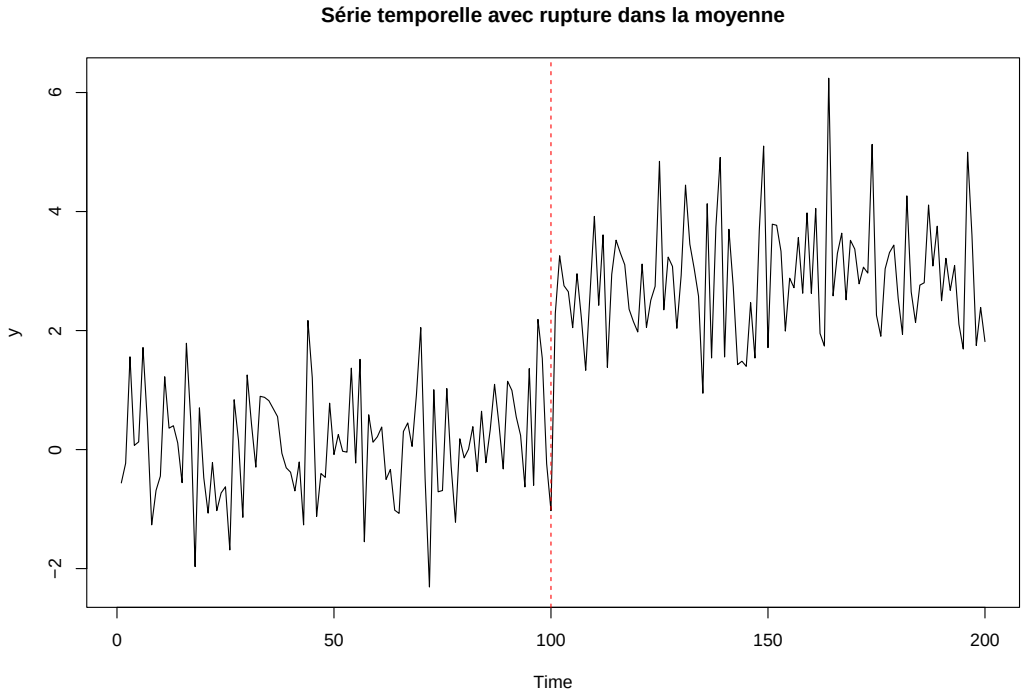
Julien JACQUES

# Qu'est-ce qu'une rupture ?

- Une **rupture** (ou "changepoint") est un moment où les propriétés statistiques d'une série changent.
- Cela peut concerner :
  - La moyenne
  - La variance
  - Les deux

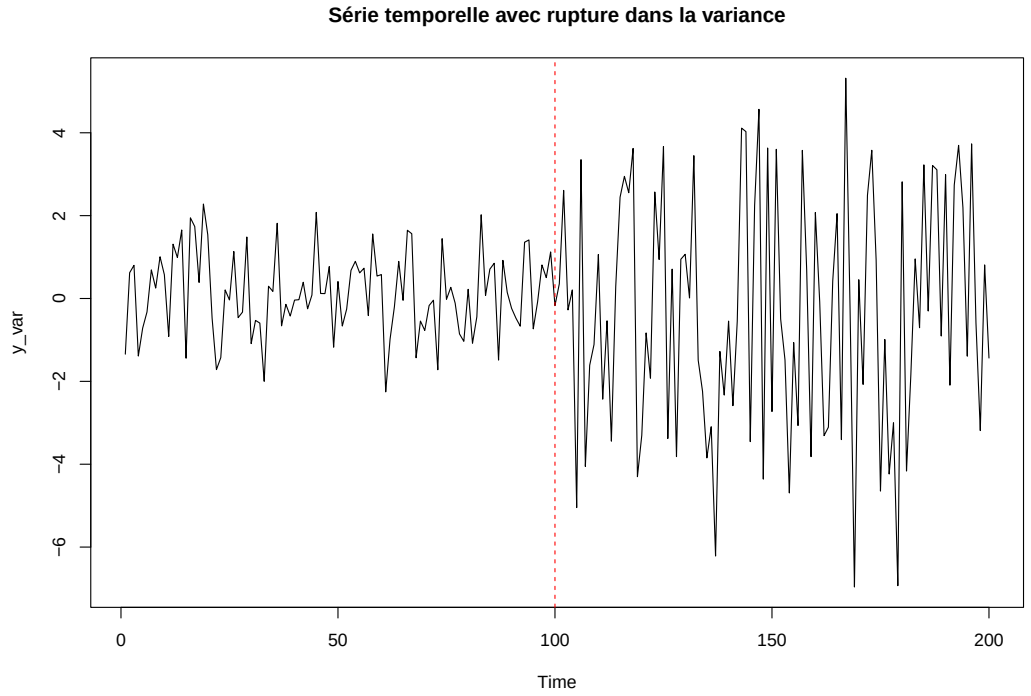
# Exemple de rupture dans la moyenne

```
set.seed(123)
y1 <- rnorm(100, mean = 0, sd = 1)
y2 <- rnorm(100, mean = 3, sd = 1)
y <- c(y1, y2)
plot.ts(y, main = "Série temporelle avec rupture dans la mo
abline(v = 100, col = "red", lty = 2)
```



# Exemple de rupture dans la variance

```r
set.seed(456)
y1 <- rnorm(100, mean = 0, sd = 1)
y2 <- rnorm(100, mean = 0, sd = 3)
y_var <- c(y1, y2)
plot.ts(y_var, main = "Série temporelle avec rupture dans l
abline(v = 100, col = "red", lty = 2)
```



# Le package changepoint

```
library(changepoint)
```

- Fournit des outils pour détecter des ruptures dans des séries temporelles
- Fonctions principales :
  - `cpt.mean()` : détection de rupture dans la moyenne
  - `cpt.var()` : détection de rupture dans la moyenne
  - `cpt.meanvar()` : détection de rupture dans la moyenne et la variance

Ces trois fonctions sont paramétrables. En particluier :

- la nature de l'algorithme utilisé :
  - PELT (Pruned Exact Linear Time)
  - BinSeg (Binary Segmentation)
  - AMOC (At Most One Change)
- le type de test utilisé pour localiser les ruptures (paramétrique, non paramétrique)
- le type et l'ampleur de la pénalisation appliquée afin de limiter la sursegmentation

# L'algorithme BinSeg

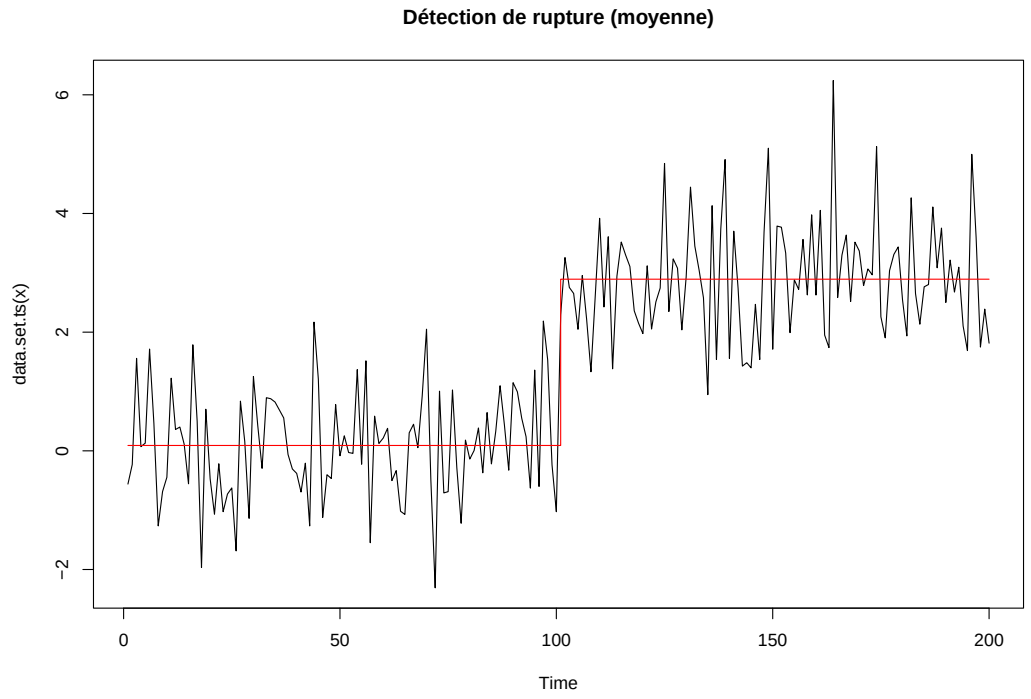
L'algorithme BinSeg (1974) pour la détection de rupture dans la moyenne fonctionne ainsi :

- on cherche l'endroit de coupure dans la série tel que la variance inter-groupe soit la plus grande possible
- on fait un test de comparaison de moyennes (t.test) des données avant et après rupture
- si le test détecte une rupture, on recommence la procédure dans chaque sous groupe
- et ainsi de suite jusqu'à ce que plus aucune rupture ne soit détectée

# Détection de rupture : Moyenne

Utilisation de la fonction `cpt.mean()` avec ses paramètres par défaut

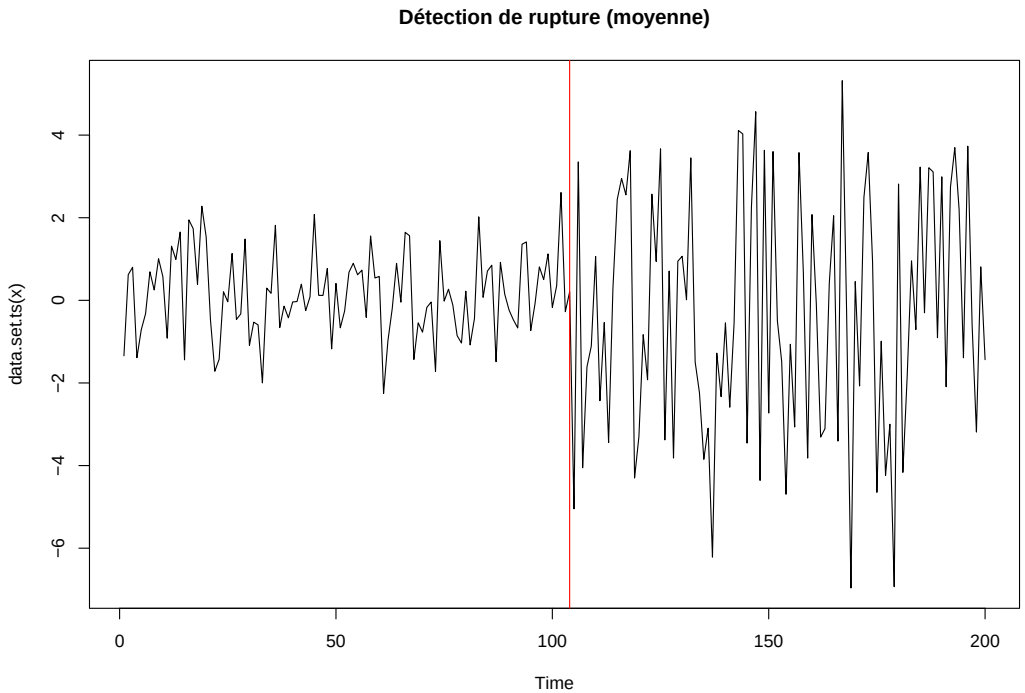
```
cpt_result <- cpt.mean(y, method="PELT")
plot(cpt_result, main="Détection de rupture (moyenne)")
```



# Détection de rupture : Variance

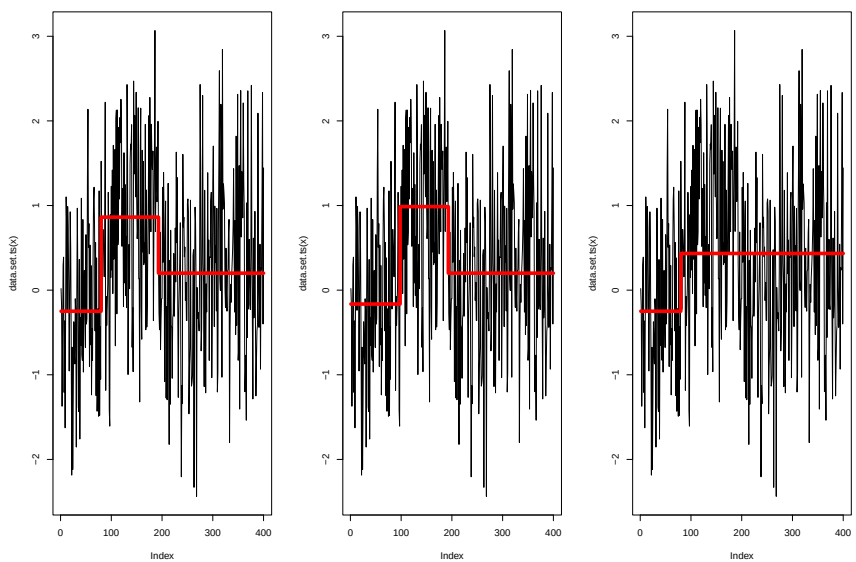
Utilisation de la fonction `cpt.var()` avec ses paramètres par défaut

```
cpt_result <- cpt.var(y_var, method="PELT")
plot(cpt_result, main="Détection de rupture (moyenne)")
```



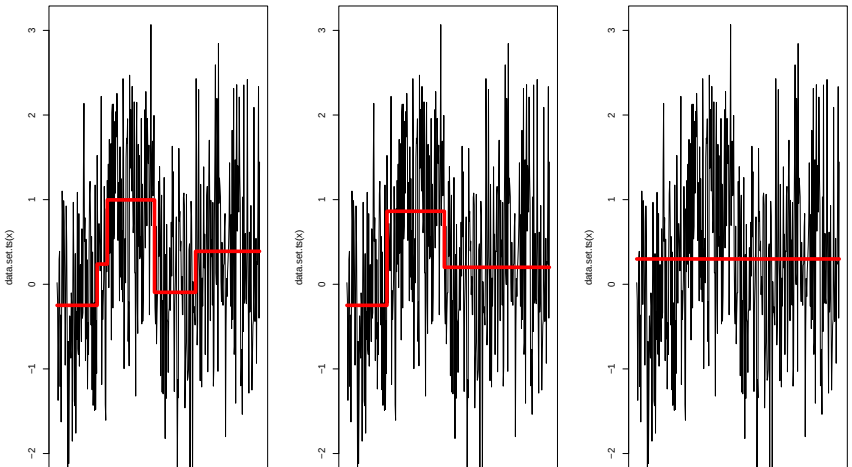
# Influence du type de méthode

```
set.seed(10)
x <- c(rnorm(100, 0, 1), rnorm(100, 1, 1), rnorm(100, 0, 1)
       rnorm(100, 0.2, 1))
par(mfrow=c(1,3))
m.binseg <- cpt.mean(x, method = "BinSeg")
plot(m.binseg, type = "l", xlab = "Index", cpt.width = 4)
m.PELT <- cpt.mean(x, method = "PELT")
plot(m.PELT, type = "l", xlab = "Index", cpt.width = 4)
m.AMOC <- cpt.mean(x, method = "AMOC")
plot(m.AMOC, type = "l", xlab = "Index", cpt.width = 4)
```



# Influence de la pénalité

```
set.seed(10)
x <- c(rnorm(100, 0, 1), rnorm(100, 1, 1), rnorm(100, 0, 1),
       rnorm(100, 0.2, 1))
par(mfrow=c(1,3))
m.binseg1 <- cpt.mean(x, method = "BinSeg",penalty = "Manual",
                      pen.value = "1 * log(n)")
plot(m.binseg1, type = "l", xlab = "Index", cpt.width = 4)
m.binseg2 <- cpt.mean(x, method = "BinSeg",penalty = "Manual",
                      pen.value = "2 * log(n)")
plot(m.binseg2, type = "l", xlab = "Index", cpt.width = 4)
m.binseg3 <- cpt.mean(x, method = "BinSeg",penalty = "Manual",
                      pen.value = "5 * log(n)")
plot(m.binseg3, type = "l", xlab = "Index", cpt.width = 4)
```



# Remarque

Pour choisir l'algorithme de segmentation et autres paramètres associés, on peut s'appuyer sur :

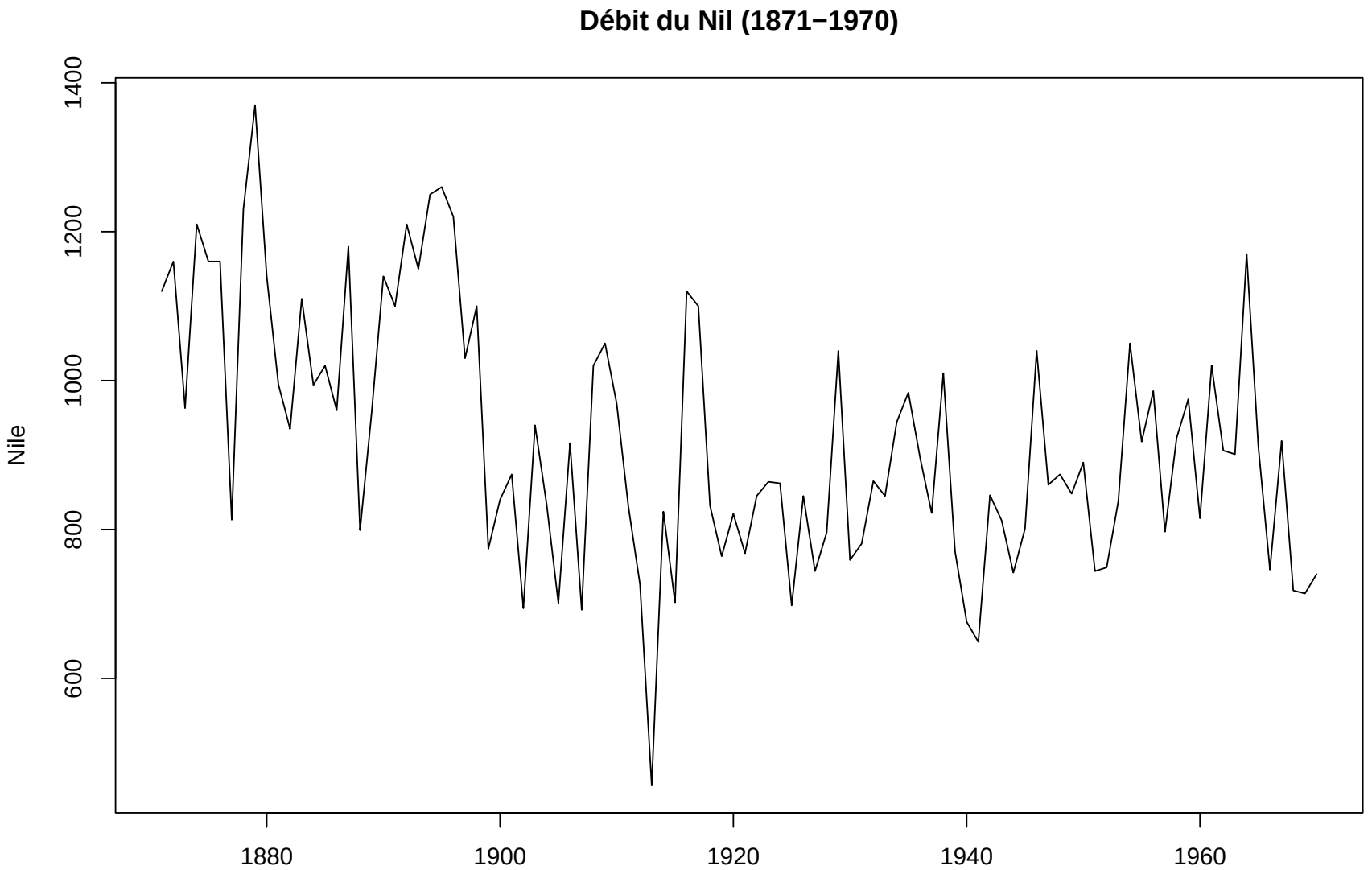
- des critères théoriques : quel modèle est le plus adapté aux données ? Avec variance constante ? Avec résidus gaussiens ? Un qui rendrait compte de segments de longueurs très variables ?...
- des critères empiriques : on choisit la méthode qui donne les résultats les plus "exploitables" sur des données réelles, et les plus corrects sur des données simulées

# Références

Killick, Rebecca, and Idris A. Eckley. 2014. “changepoint: An R Package for Changepoint Analysis.” Journal of Statistical Software 58 (3). http://www.jstatsoft.org/v58/i03.

# Application

```
library(strucchange)
data(Nile)
plot(Nile, main = "Débit du Nil (1871-1970)")
```



# Autres outils

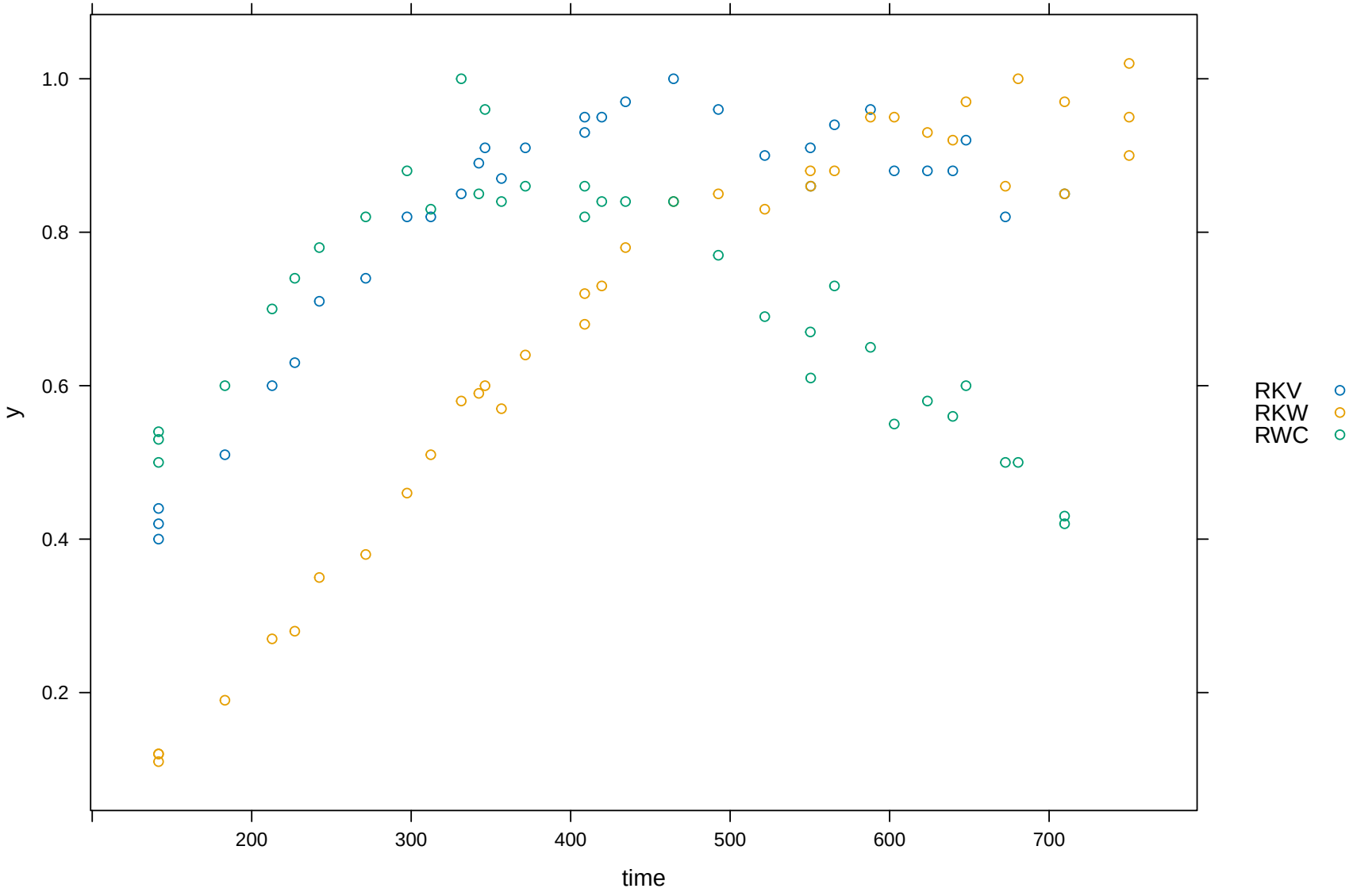
- modèle de régression avec détection de rupture

```
library(strucchange)
library(segmented)
```

# Détection de rupture en régression

On

mesure des organes (RKV, RKW, RWC) de plantes à différents temps



# Détection de rupture en régression

On va chercher dans chaque série, à trouver une rutpure dans le temps

```
fit<-segreg(y~ 0+group+seg(time, by=group, npsi=2), data=plant)
summary(fit)
```

```
##
##  ***Regression Model with Segmented Relationship(s)***
##
## Call:
## segreg(formula = y ~ 0 + group + seg(time, by = group, npsi =
##     data = plant)
##
## Estimated Break-Point(s):
##                           Est. St.Err
## psi1.time:groupRKV 314.581 21.388
## psi2.time:groupRKV 442.541 22.193
## psi1.time:groupRKW 447.462 24.044
## psi2.time:groupRKW 602.895 49.229
## psi1.time:groupRWC 331.136  8.308
## psi2.time:groupRWC 652.188 43.611
##
```

# Détection de rupture en régression

On affiche les résultats obtenus

```
plot(y~time, data=plant)
plot(fit, term=1:3, add=TRUE, leg=NA, psi.lines=TRUE)
```