Introduction aux séries temporelles, tendance et composante saisonnière

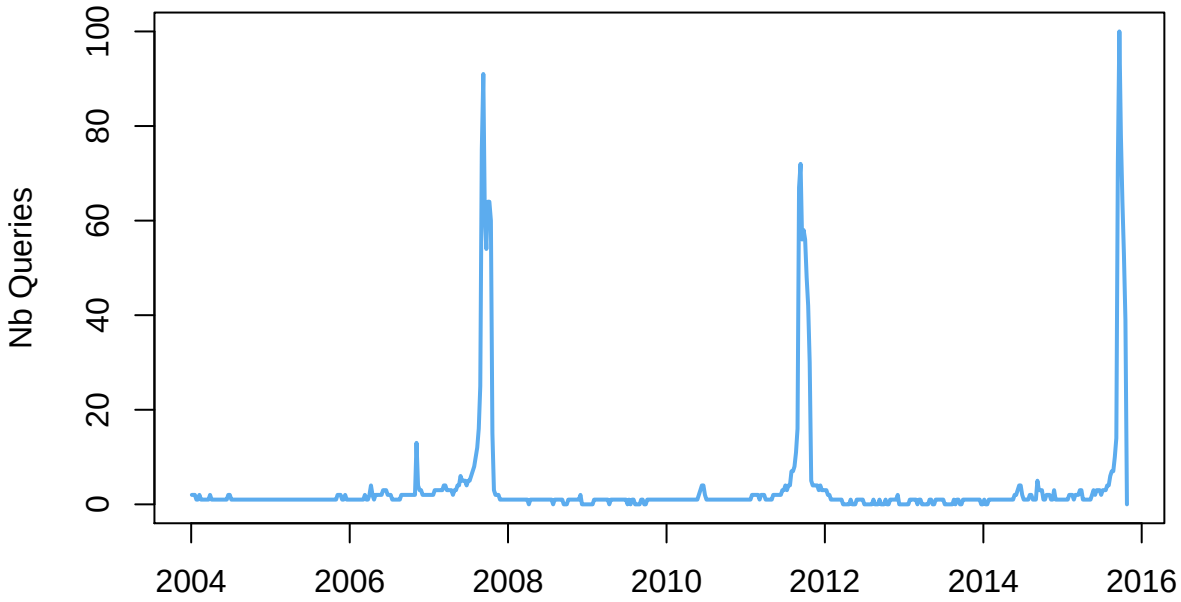
MAP-STA2 : Séries chronologiques

Yannig Goude *yannig.goude@edf.fr*

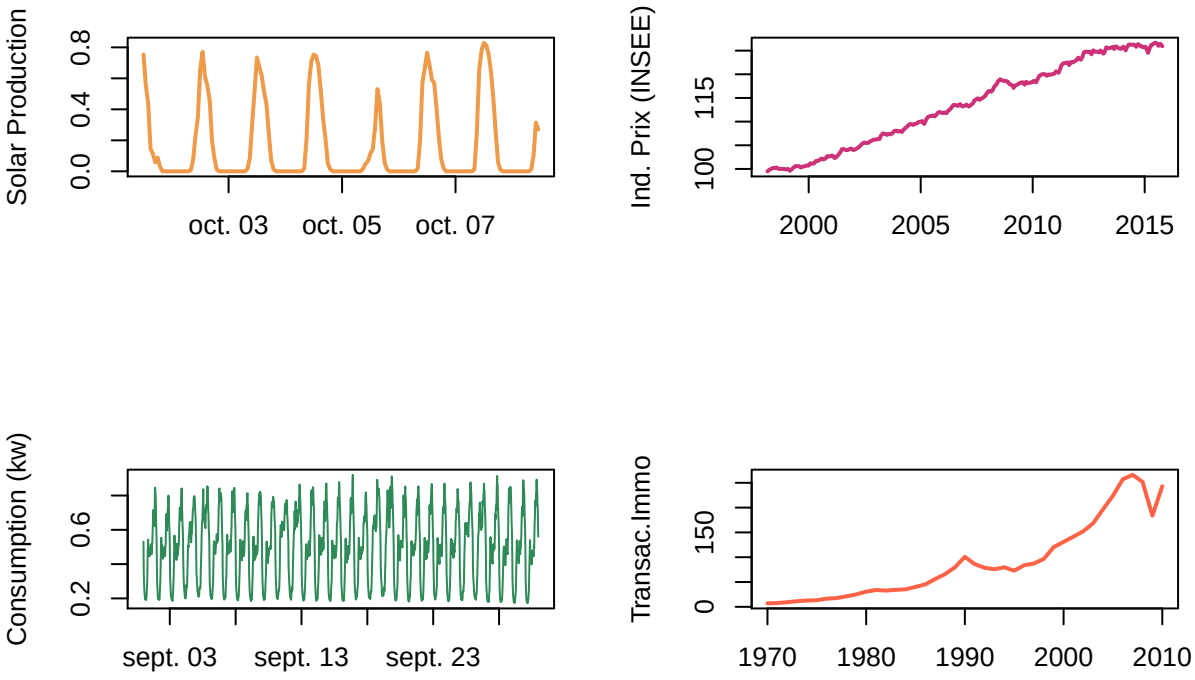
2015-2016

Exemples de séries temporelles

Nous étudions dans ce cours des séries temporelles à temps discret, ie une suite réelle $(x_t)_{1\leq t\leq n}$ où t représente le temps. Ce temps est mesuré à une fréquence donnée, appelée fréquence d'échantillonnage. Par exemple, les données du nombre de requétes google "rubgy world cup" par semaine (source https://www.google.fr/trends/),



la production horaire d'un panneau photovoltaïque, l'indice des prix à la consommation des ménages Français (source INSEE: http://www.insee.fr/fr/bases-de-donnees/) au pas mensuel, la consommation électrique résidentielle moyenne en Irlande au pas demi-horaire, les montants de transactions immobilières en France en milliards d'euros depuis 1970 (sources: https://www.data.gouv.fr/fr).



Décomposition d'une série temporelle

L'objectif principal de l'analyse d'une série temporelle est la prévision de ses futures réalisations. Afin de réaliser cet objectif, une premiére étape de modélisation de la série est nécessaire. Cette étape consiste à sélectionner, parmi une famille de modèles correspondant à des approximations de la réalité, celui qui décrit le mieux la série en question.

Quelques exemples de modéles de série temporelle:

- les lissages exponentiels
- les modéles de régression (régression linéaire, modéles non-paramétriques...)
- les modéles du type ARIMA
- les modéles de données fonctionnelles

Une série temporelle Y_t est communément décomposée en:

- une tendance T_t correspondant à une évolution à long terme de la série, par exemple:
  - tendance linéaire: $T_t = a + bt$
  - tendance quadratique: $T_t = a + bt + ct^2$
  - tendance logarithmique: $T_t = log(t)$
- une saisonnalité S_t correspondant à un phénoméne périodique de période identifiée
- une erreur ε_t qui est la partie aléatoire de la série (idéalement stationnaire)

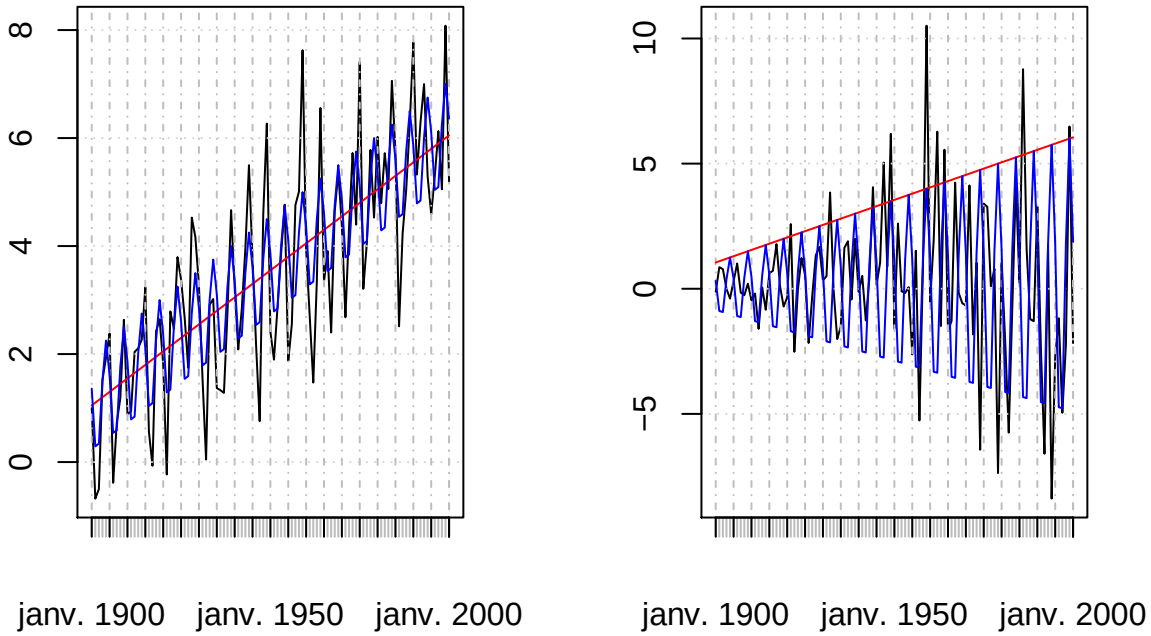
On ajoute parfois une autre composante, le cycle C_t qui correspond à un phénoméne répétitif régulier (donc prévisible) de période inconnue ou changeante.

Cette décomposition peut-étre additive $Y_t = T_t + S_t + \varepsilon_t$ ou multiplicative $Y_t = T_t * S_t * \varepsilon_t$. Il est également possible de combiner ces deux décompositions: $Y_t = (T_t + S_t) * \varepsilon_t$ ou $Y_t = (T_t * S_t) + \varepsilon_t \ldots$

Nous nous intéressons ici à la modélisation de la composante déterministe de la série: T_t et S_t.

Rq: un passage au *log* permet de se ramener à un modéle additif si le modéle étudié est totalement multiplicatif.

Exemple:



Modélisation de la partie déterministe

La tendance

Il existe différents procédés permettant d'anlyser puis/ou de corriger la tendance d'une série temporelle.

Moyenne mobile La moyenne mobile est une méthode simple permettant d'extraire les composantes basses fréquences d'une série temporelle autrement dit sa tendance. Elle est également connue comme une méthode de lissage car elle agit comme un filtre passe bas et donc élimine le bruit.

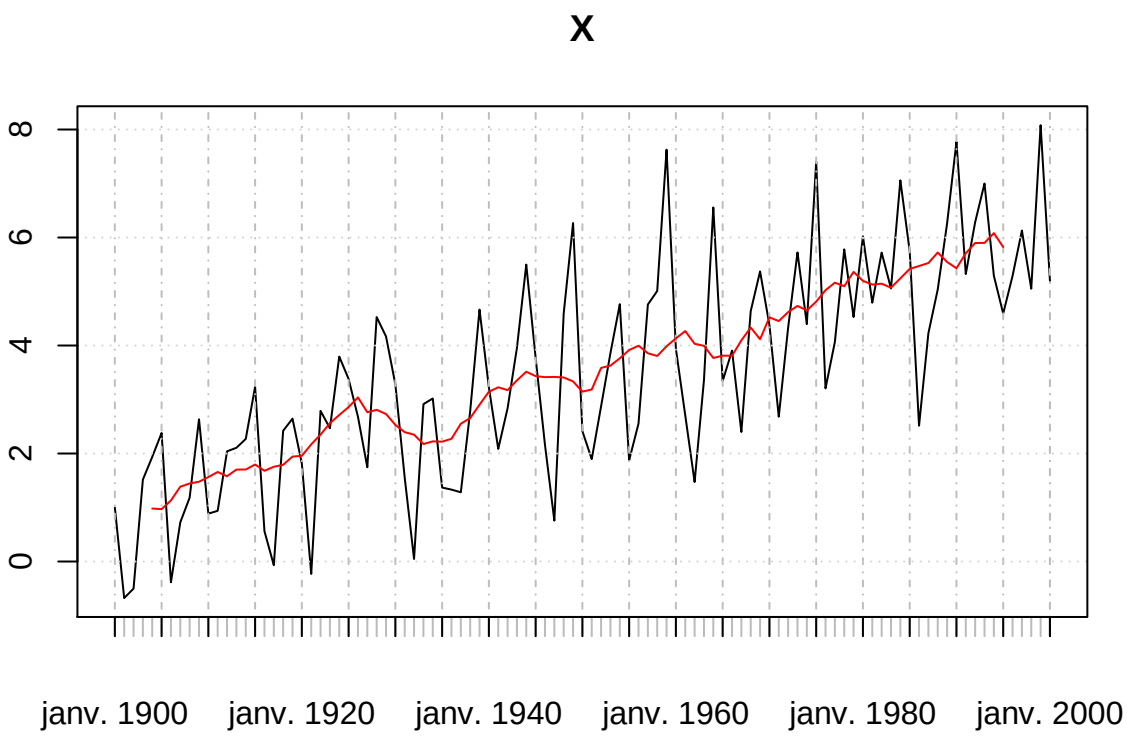
Le calcul de la moyenne mobile dépend d'un paramètre l appelé la largeur de fenêtre. Ce paramétre correspond au nombre d'observations inclues dans le calcul de la moyenne glissante éffectuée. Plus l est grand plus le lissage est important (jusqu'à atteindre la fonction constante égale à la moyenne).

La moyenne mobile se calcule ainsi:

$$\widehat{y_t} = \frac{1}{2l+1} \sum_{i=t-l}^{t+l} y_t$$

Et en r, une des nombreuses alternatives est la fonction `filter`:

```
MA<-filter(X, filter=array(1/10,dim=10), method = c("convolution"),
             sides = 2, circular = FALSE)
MA<-xts(MA,order.by=Date)

plot(X,type='l')
lines(MA,col='red')
```



Remarquons que la moyenne mobile est un estimateur non-paramétrique de la tendance, au sens ou nous ne supposons pas de structure a-priori de cette tendance (par ex. linéaire ou polynomiale).

Différenciation Pour nettoyer une série de sa tendance et/où de sa saisonnalité, nous pouvons procéder par différenciation. Celà fonctionne pour des séries à tendance polynomiale.

Notons Δ l'opérateur de différenciation: $\Delta y_t = y_t - y_{t-1}$.

L'opérateur de différenciation d'ordre k correspondant est: $\Delta^k y_t = \Delta(\Delta^{k-1} y_t)$

Proposition soit un processus y_t admettant une tendance polynomiale d'ordre k:

$$y_t = \sum_{j=0}^{k} a_j t^j + \varepsilon_t$$

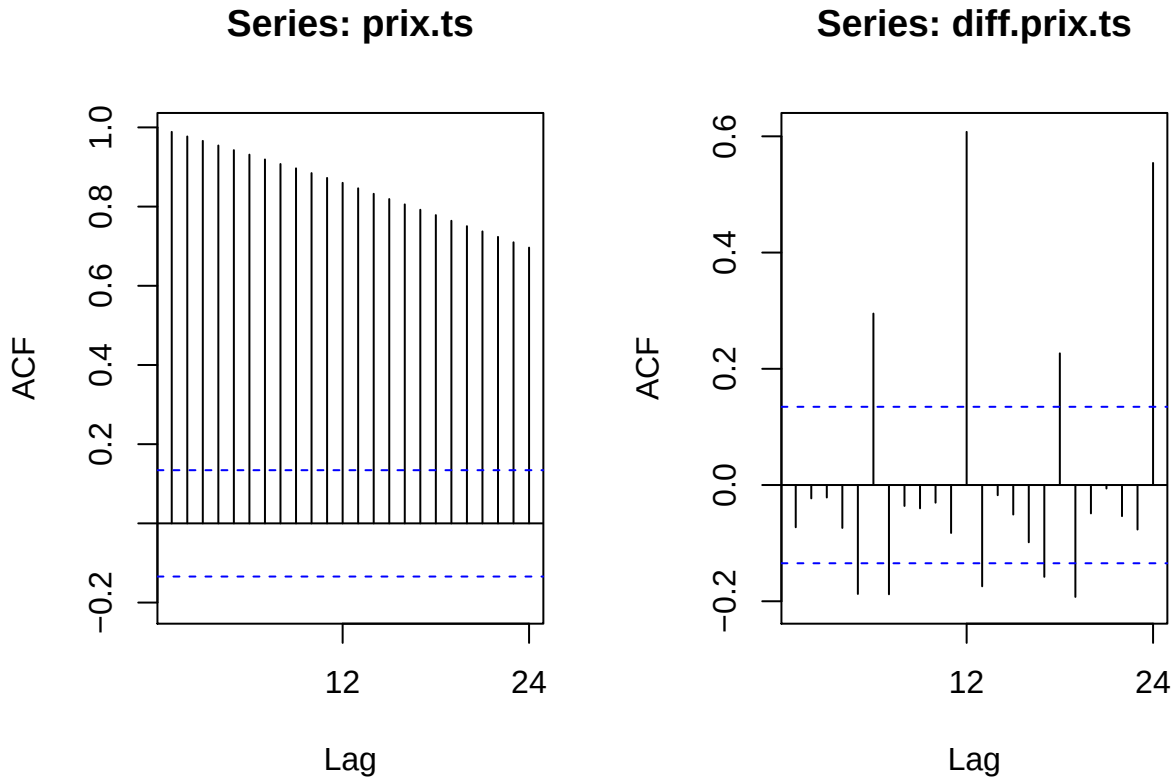
alors le processus Δy_t admet une tendance polyomiale d'ordre $k-1$.

Preuve à faire

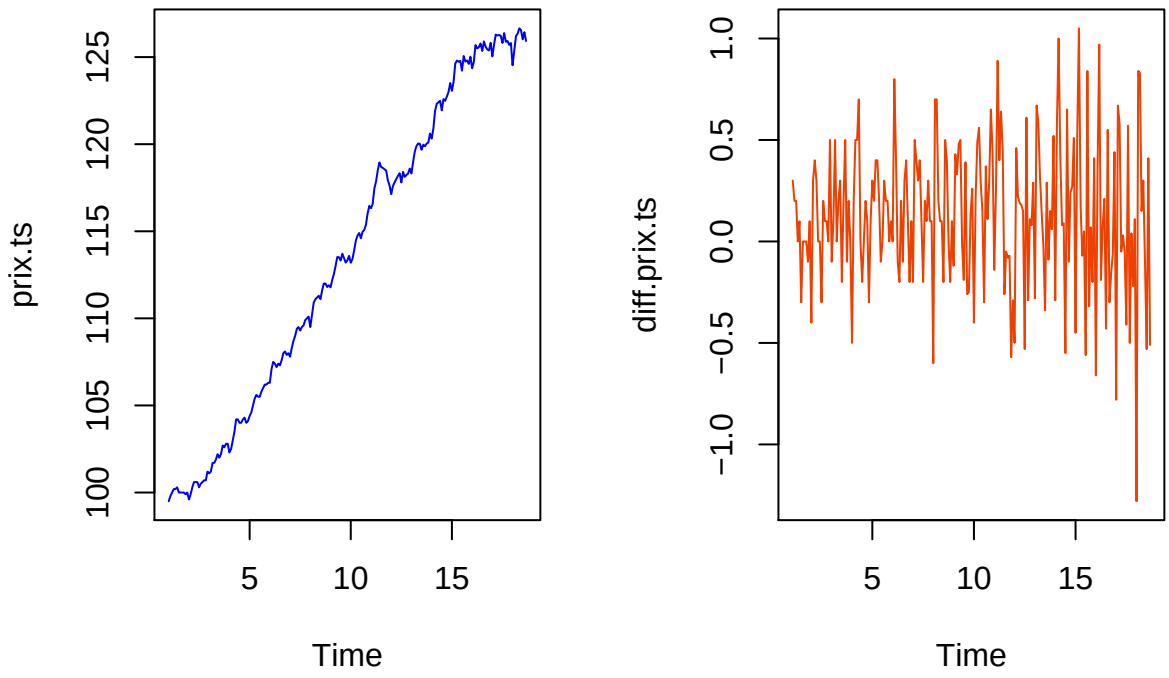
Il en découle que pour éliminer une tendance polynomiale d'ordre k on peut effectuer une différenciation d'ordre k.

La fonction permettant de différencier une série temporelle est la fonction `diff` dont voici un exemple d'utilisation sur l'indice des prix à la consommation des ménages:

```
par(mfrow = c(1, 2))
prix.ts <- ts(dataPRIX$PrixConso, frequency = 12)
Acf(prix.ts, na.action = na.omit)

diff.prix.ts <- diff(prix.ts, lag = 1, differences = 1)
Acf(diff.prix.ts, na.action = na.omit)
```




```
plot(prix.ts, col = "blue")
plot(diff.prix.ts, col = "orangered2")
```



Estimation paramétrique de la tendance Après avoir représenté la série, il est souvent possible d'inférer une représentation paramétrique de sa tendance. Dans ce cas, on procède par régression (linéaire le plus souvent mais potentiellent non-linéaire) pour estimer cette tendance.

Par exemple, dans le cas d'un processus y admettant une tendance polynomiale d'ordre k: $y_t = \sum_{j=0}^{k} a_j t^j + \varepsilon_t$, un estimateur de la tendance pourra être obtenu ainsi:

$\widehat{T} = X\widehat{a}$

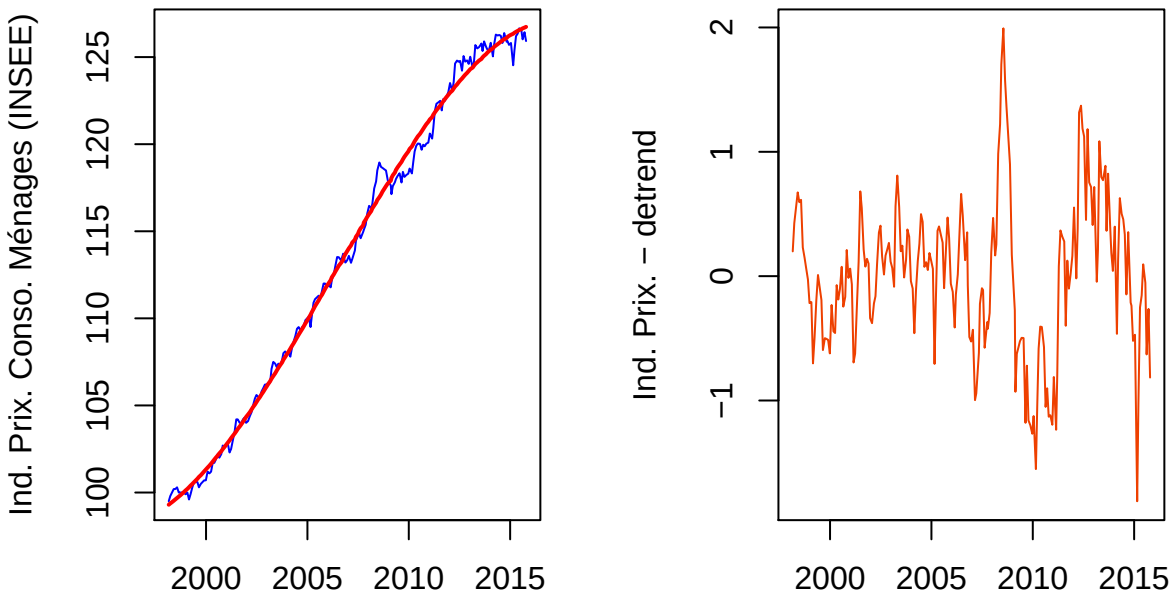
ou X est la matrice dont les colonnes sont les vecteurs $(1, ..., t^j)$ et:

$$\widehat{a} = (X'X)^{-1}X'Y$$

avec $Y = (y_1, ..., y_t)$

En pratique, la fonction `lm` ou `nlm` dans le cas non linéaire permet d'estimer ce type de tendance.

```
time <- c(1:nrow(dataPRIX))
dataPRIX$time <- time
reg <- lm(PrixConso ~ time + I(time^2) + I(time^3), data = dataPRIX)
par(mfrow = c(1, 2))
plot(dataPRIX$Date, dataPRIX$PrixConso, type = "l", xlab = "",
    ylab = "Ind. Prix. Conso. Ménages (INSEE)", col = "blue")
lines(dataPRIX$Date, reg$fitted, col = "red", lwd = 2)
plot(dataPRIX$Date, dataPRIX$PrixConso - reg$fitted, type = "l",
    xlab = "", ylab = "Ind. Prix. - detrend", col = "orangered2")
```



Estimation non-paramétique de la tendance Dans certains cas, une représentation paramétrique de la tendance n'est pas évidente. Le modèle sous-jacent à ce type de données est:

$$y_t = f(t) + \varepsilon_t$$

où f est une fonction **régulière** sur laquelle on ne fait pas d'hypothèse paramétrique, $t = 1, 2, .., n$. On ne fait pour l'instant pas d'hypothèses précises sur ε_t, considérés comme stationnaires. On pourra dans ce cas considérer une estimation non-paramétrique de cette tendance. Plusieurs approches sont possibles.

Estimateur à noyaux

définition on appelle noyau une fonction $K : \mathbb{R}^d \to \mathbb{R}$ telle que $\int K^2 < \infty$ et $\int K = 1$

définition soit un réel $h > 0$ (paramètre de fenêtre), soit un noyau K. On appelle estimateur à noyau de f associé à la fenêtre h et au noyau K la fonction $\widehat{f}_h$ définie par:

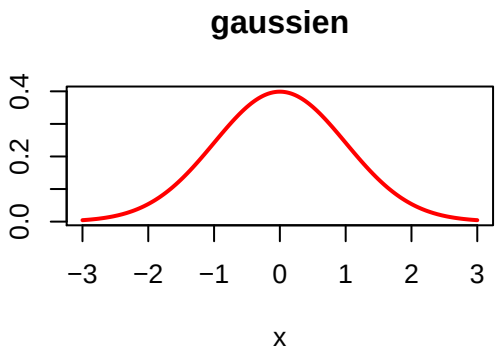
$$\widehat{f}_h(x) = \frac{\sum_{t=1}^{n} y_t K(\frac{x-t}{h})}{\sum_{t=1}^{n} K(\frac{x-t}{h})}$$

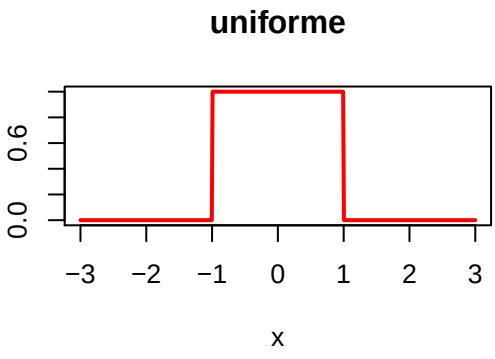
c'est une estimation non-paramétrique de la tendance de la série. La régularité de cet estimateur dépend de h la taille de fenêtre du noyau.

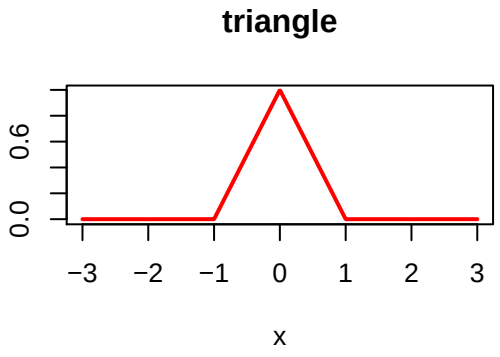
exemple de noyaux:

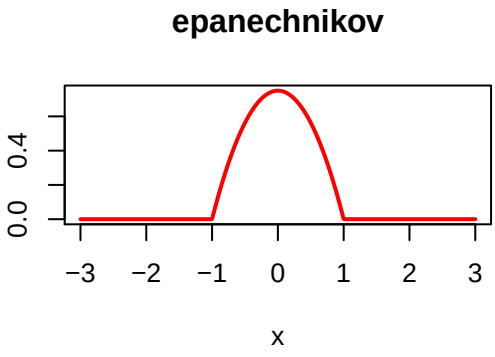
- gaussien: $K(x) = \exp(-x^2/)/2pi$
- uniforme: $K(x) = 1_{|x|\leq 1}/2$

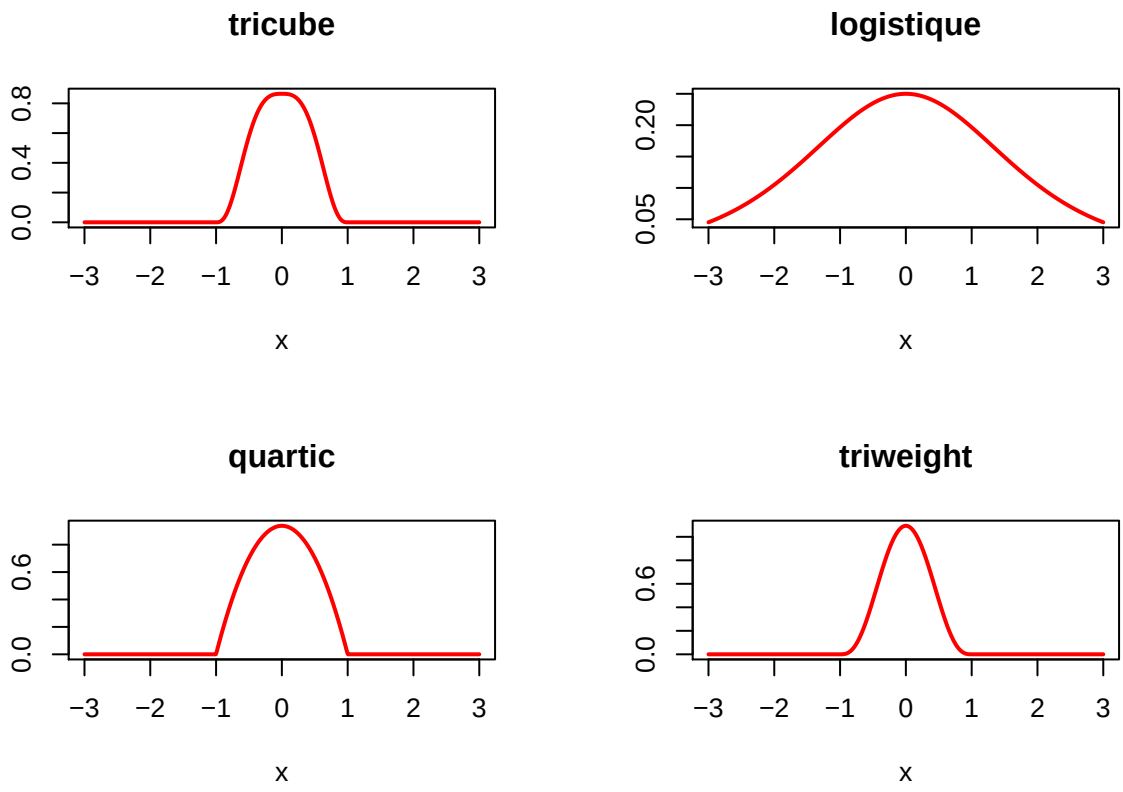
- triangle: $K(x) = (1 - |x|)1_{|x|\leq 1}$
- epanechnikov $K(x) = \frac{3}{4}(1 - x^2)1_{|x|\leq 1}$
- tricube $K(x) = \frac{70}{81}(1 - |x|^3)^3 1_{|x|\leq 1}$
- logistique $K(x) = 1/(\exp(x) + 2 + \exp(-x))$
- quartic: $K(x) = \frac{15}{16}(1 - x^2)1_{|x|\leq 1}$
- triweight!: $K(x) = \frac{35}{32}(1 - x^2)^3 1_{|x|\leq 1}$







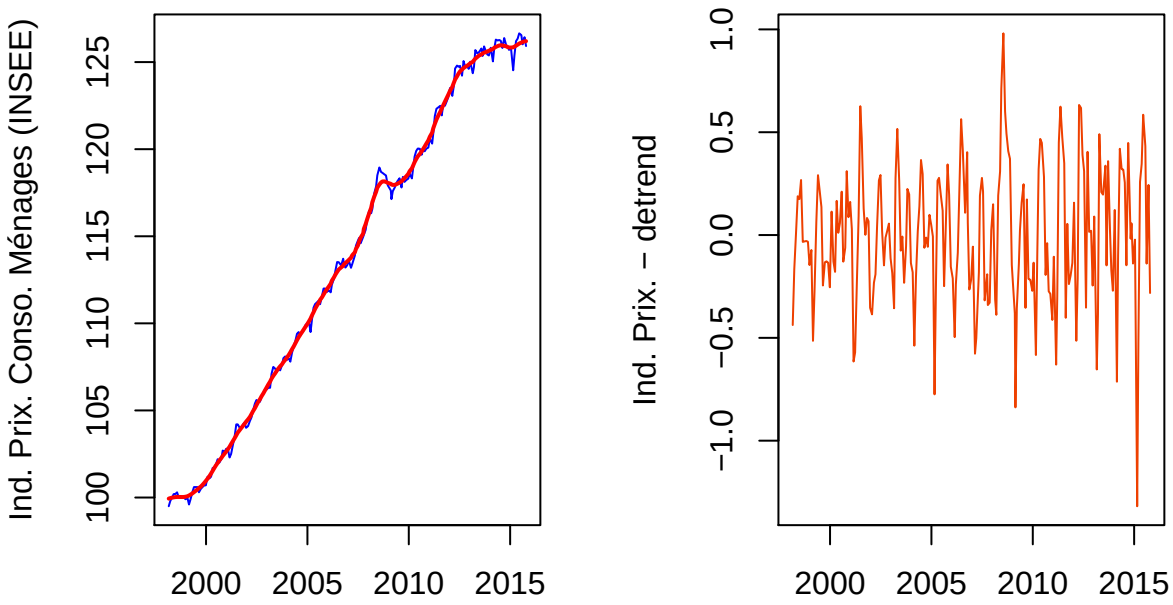




Une fonction permettant d'effectuer une régression à noyau en r est `ksmooth` du package `stats` disponible dans la distribution r de base.

Un exemple d'utilisation sur les données de l'indice des prix à la consommation des ménages:

```
noyau <- ksmooth(dataPRIX$time, dataPRIX$PrixConso, kernel = c("normal"),
    bandwidth = 10)
par(mfrow = c(1, 2))
plot(dataPRIX$Date, dataPRIX$PrixConso, type = "l", xlab = "",
    ylab = "Ind. Prix. Conso. Ménages (INSEE)", col = "blue")
lines(dataPRIX$Date, noyau$y, col = "red", lwd = 2)
plot(dataPRIX$Date, dataPRIX$PrixConso - noyau$y, type = "l",
    xlab = "", ylab = "Ind. Prix. - detrend", col = "orangered2")
```



Polynômes locaux

définition soit un réel $h > 0$ (paramètre de fenêtre), soit un noyau K. On note $W_t(x) = \frac{K(\frac{x-t}{h})}{\sum_{t=1}^{n} K(\frac{x-t}{h})}$ (on ne fait pas apparaitre ici la dépendance à h pour simplifier).

On appelle estimateur polynomial local de degré q de f associé à la fenêtre h et au noyau K la fonction $\widehat{f}_h$ définie par:

$$\widehat{f}_h(x) = \text{argmin}_P \sum_{t=1}^{n} W_t(x)||y_t - P(x_t - x)||^2$$

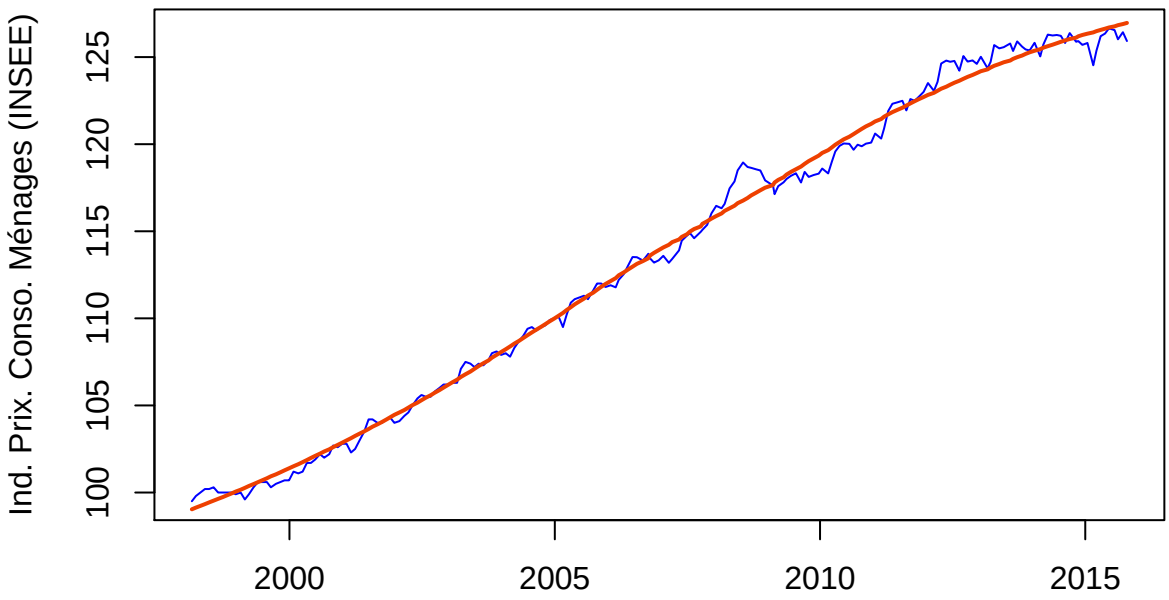
avec $P(x) = \sum_{j=0}^{q} a_j x^j$ un polynôme de degrés q.

Le principe est donc, pour chaque valeur de x (ici le temps car on estime une tendance ou une composante périodique de la série), on estime une fonction polynômiale approximant le mieux les données localement, la notion de voisinage dépendant encore de h la taille de fenêtre. Autrement formulé, il s'agit d'estimer sur les données un développement limité de la fonction f.

On remarque que pour $q = 0$ on retrouve l'estimateur à noyau précédant qui consiste à résoudre $\sum_{t=1}^{n} W_t(x)||y_t - a||^2$.

La fonction r implémentant les polynômes locaux est la fonction `loess` dont voilà un exemple d'utilisation:

```
lo <- loess(PrixConso ~ time, data = dataPRIX, degree = 2, span = 0.7)
plot(dataPRIX$Date, dataPRIX$PrixConso, type = "l", xlab = "",
    ylab = "Ind. Prix. Conso. Ménages (INSEE)", col = "blue")
lines(dataPRIX$Date, lo$fitted, col = "orangered2", lwd = 2)
```



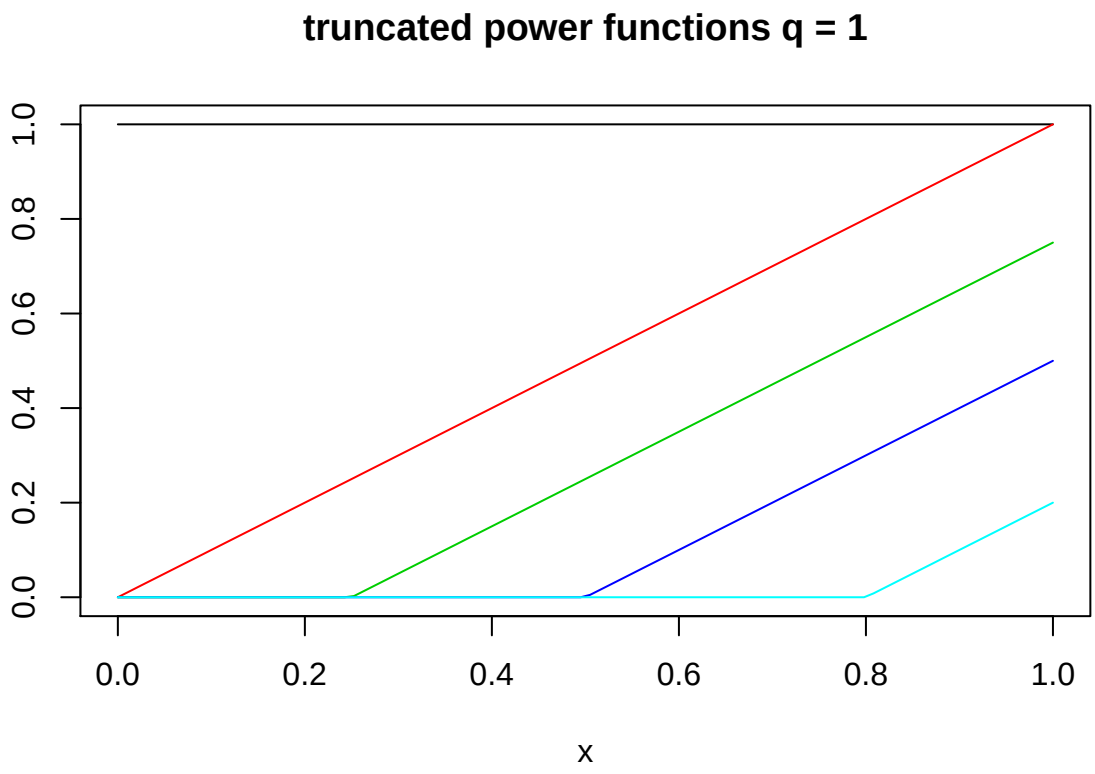
Notons que le noyau utilisé dans cette fonction est le noyau tricube.

Estimation semi-paramétrique de la tendance Une autre alternative pour estimer la fonction f est de procéder par projection sur des bases de fonctions adaptés, par exemple des fonctions splines polynomiales par morceau.

définition soit $1 \leq n_1 \leq ... \leq n_k \leq n$ un vecteur de coefficients appelés noeuds définissant une partition du temps, q un entier >0, alors: $(1, x, x^2, ..., x^q, (x - n_1)_+^q, ..., (x - n_k)_+^q)$ est appelée base de truncated power functions de degré q et est de classe C^{q-1}.

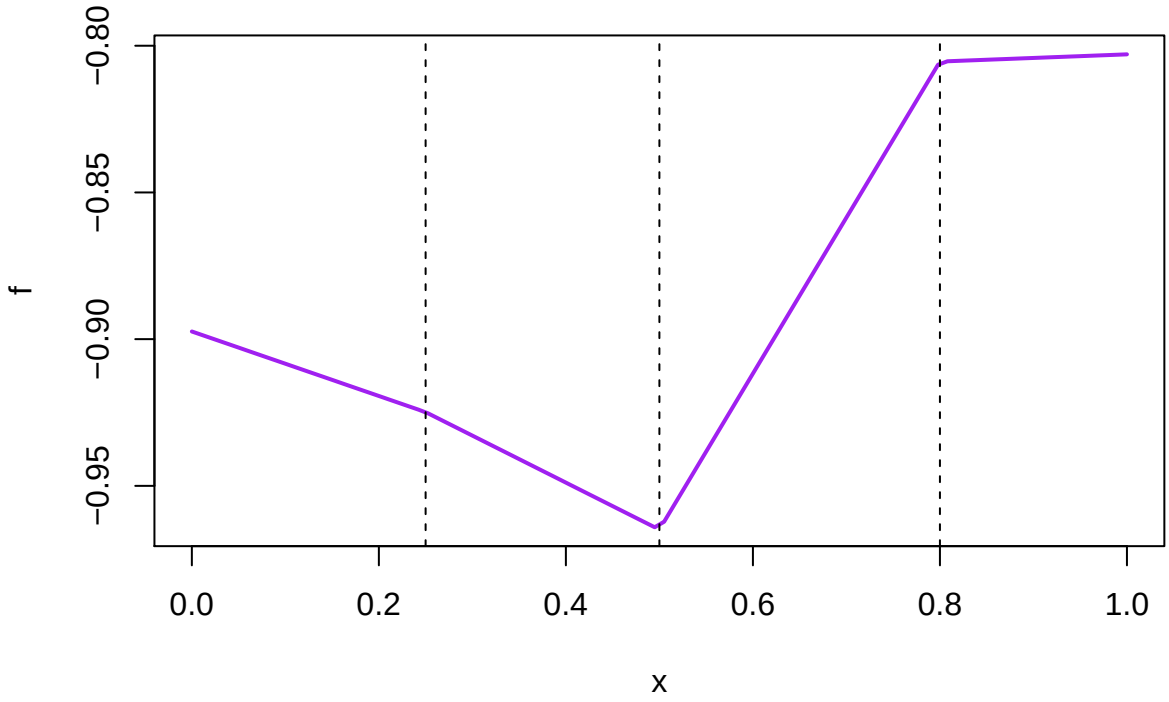
Par exemple, pour $q = 1$ on obtient une base de fonction spline linéaire dont l'aspect est le suivant:

```
n <- 100
const <- rep(1, n)
f1 <- function(x) x
f2 <- function(x) pmax(x - 0.25, 0)
f3 <- function(x) pmax(x - 0.5, 0)
f4 <- function(x) pmax(x - 0.8, 0)

x <- seq(0, 1, length = n)
design <- as.matrix(data.frame(const = const, f1 = f1(x), f2 = f2(x),
    f3 = f3(x), f4 = f4(x)))
matplot(x, y = design, type = "l", lty = 1, ylab = "", main = "truncated power functions q = 1")
```



ce qui permet de modéliser des tendances linéaires par morceau du type:

```
set.seed(150)
coef <- runif(5, -1, 1)
f <- design %*% coef
plot(x, f, type = "l", col = "purple", lwd = 2)
abline(v = c(0.25, 0.5, 0.8), lty = "dashed")
```



En pratique les coefficients de projection sur la base sont estimés par régression linéaire sur la matrice de design: $X = (1, x, x^2, ..., x^q, (x - n_1)_+^q, ..., (x - n_k)_+^q)$.

Le choix des "cassures" donc des noeuds est déterminant. $q = 3$ est un choix courant car il permet de contourner ce problème du choix des noeuds en le substituant à un problème de régularisation en résolvant le problème de régression pénalisée suivant:

$$\min_{f \in S_3} (y_t - f(t))^2 + \lambda \int f''(x)^2 dx$$

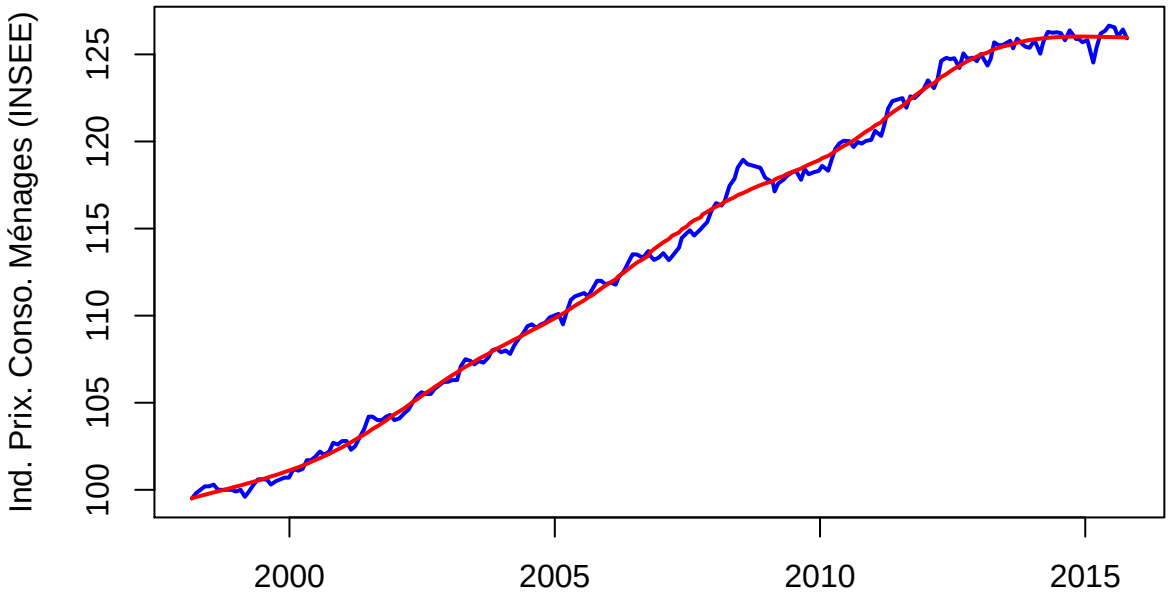
où S_3 est l'espace engendré par $X = (1, x, x^2, ..., x^3, (x - n_1)_+^q, ..., (x - n_k)_+^3)$ en prenant k suffisamment grand, $\lambda > 0$ est un paramètre à calibrer, par exemple par validation croisée.

en pratique on pourra utiliser la fonction `gam` du package `mgcv`.

```
library(mgcv)
```

```
## Loading required package: nlme
##
## Attaching package: 'nlme'
##
## The following object is masked from 'package:forecast':
##
##     getResponse
##
## This is mgcv 1.8-9. For overview type 'help("mgcv-package")'.
```

```
g <- gam(PrixConso ~ s(time, k = 10), data = dataPRIX)
plot(dataPRIX$Date, dataPRIX$PrixConso, type = "l", xlab = "",
    ylab = "Ind. Prix. Conso. Ménages (INSEE)", col = "blue",
    lwd = 2)
lines(dataPRIX$Date, g$fitted, col = "red", lwd = 2)
```



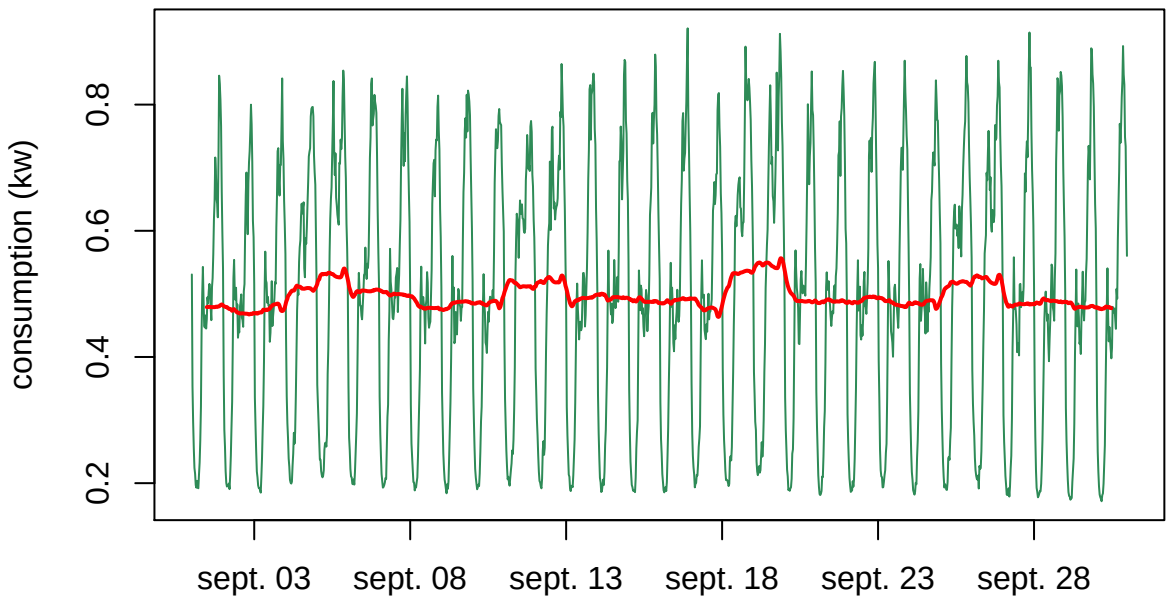
La Saisonnalité

Les traitements s'appliquant à la tendance s'appliquent également à la saisonnalité avec les variantes suivantes.

Moyenne mobile La moyenne mobile, en choisissant un paramètre de fenêtre l égal à la période de la série, permet de désaisonnaliser une série. En effet, par définition, la composante périodique est d'intégrale nulle sur la période.

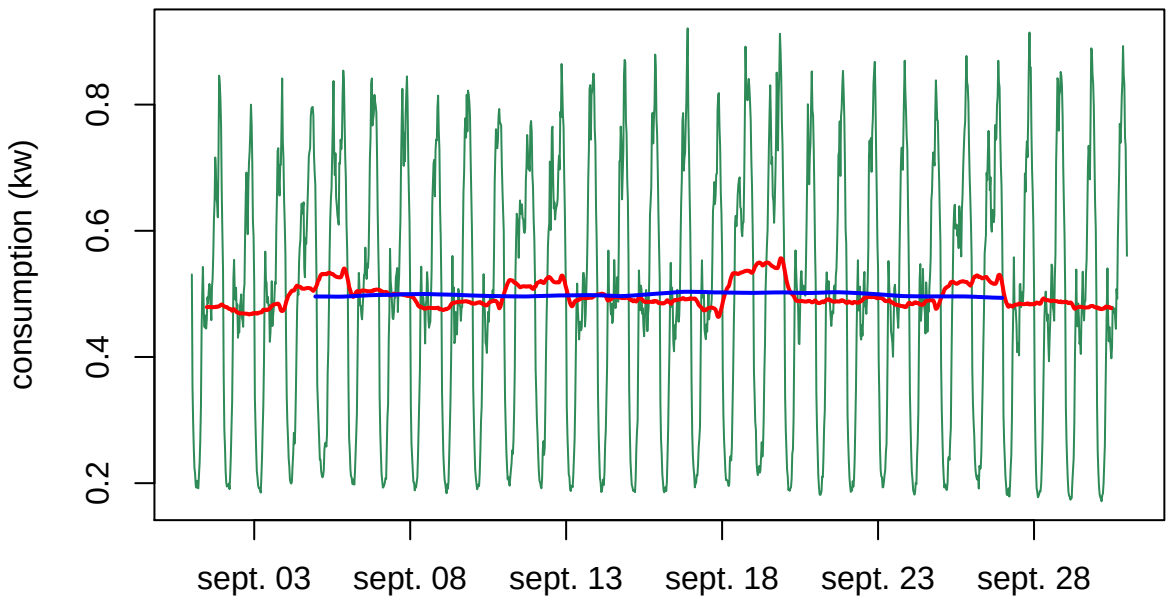
En reprenant l'exemple des données Irlandaises, si on choisit $l = 48$, cela permet d'extraire la composante périodique de période une journée:

```
MA <- filter(dataIRISH$Conso, filter = array(1/48, dim = 48),
    method = c("convolution"), sides = 2, circular = FALSE)

plot(dataIRISH$Date, dataIRISH$Conso, type = "l", xlab = "",
    ylab = "consumption (kw)", col = "seagreen4", lwd = 1)
lines(dataIRISH$Date, MA, col = "red", lwd = 2)
```



Il subsiste ici une composante périodique de période 1 semaine qu'on peut ensuite éliminer via un deuxième filtrage par moyenne mobile.

```
MA2 <- filter(MA, filter = array(1/(7 * 48), dim = 7 * 48), method = c("convolution"),
    sides = 2, circular = FALSE)

plot(dataIRISH$Date, dataIRISH$Conso, type = "l", xlab = "",
    ylab = "consumption (kw)", col = "seagreen4", lwd = 1)
lines(dataIRISH$Date, MA, col = "red", lwd = 2)
lines(dataIRISH$Date, MA2, col = "blue", lwd = 2)
```



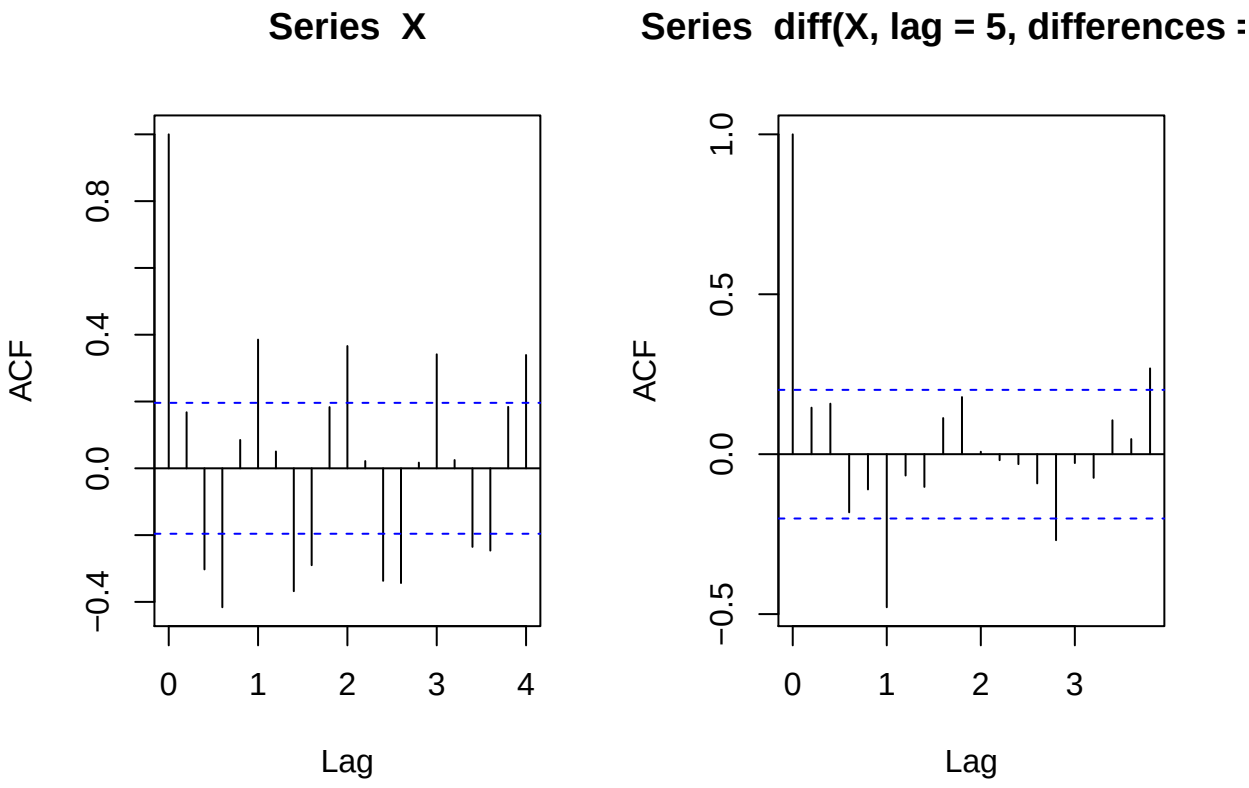
Différenciation De même que pour nettoyer un processus de sa tendance, il est possible de le désaisonnalisé par différentiation.

Proposition soit un processus y_t admettant une saisonnalité additive de période τ, alors le processus $\Delta_\tau y_t = y_t - y_{t-\tau}$ est un processus désaisonnalisé.

Preuve $y_t = S_t + \varepsilon_t$, alors $\Delta_\tau y_t = \varepsilon_t - \varepsilon_{t-\tau}$ car par définition $S_t = S_{t-\tau}$.

remarque 1 le processus $\varepsilon_t - \varepsilon_{t-\tau}$ est stationnaire, mais il peut être auto-corrélé. Par exemple si ε_t est un bruit blanc, $\varepsilon_t - \varepsilon_{t-\tau}$ a une autocorrélation d'ordre τ de $-1/2$. Voir exemple ci-dessous.

```
n <- 100
t <- c(1:n)
w = 2 * pi/5
S <- cos(w * t)
eps <- rnorm(n, 0, 1)

X <- S + eps
X <- ts(X, frequency = 5)

par(mfrow = c(1, 2))
acf(X)
acf(diff(X, lag = 5, differences = 1))
```



remarque 2 la différenciation peut s'appliquer pour un lag donnée et à un ordre donné ce qui donne l'opérateur Δ_τ^k, les paramètres associés dans la fonction `diff` sont `lag` et `differences`.

Estimation paramétrique de la saisonnalité Un modèle paramétrique naturel pour modéliser un processus saisonnier est la décomposition en série de Fourier. Soit un processus y_t admettant une saisonnalité de période τ alors le modèle suivant est généralement proposé:

$y_t = \sum_{j=1}^{q} a_j cos(\omega_j t) + b_j sin(\omega_j t) + \varepsilon_t$

où $\omega_j = 2j\pi/\tau$, q est à déterminer par une méthode de sélection de modèle sur les données.

Les coefficients a_j et b_j sont obtenus par moindre carrés sur les données.

Estimation non-paramétrique de la saisonnalité De même que pour la moyenne mobile, il est possible en choisissant la bonne valeur de fenêtre d'estimer la composante saisonnière d'un processus par méthode à noyau et polynômes locaux.

Estimation semi-paramétrique de la saisonnalité Il est possible de définir des bases de splines avec contraintes au bord du support, on parle alors de splines cycliques.