# [ENERCOOP]

Présentation, Problématique, Plan

**Présentation (**Lien Enercoop**)**

- Fournisseur français d'électricité d'origine renouvelable
- Activité fonctionnant exclusivement par des accords de gré à gré avec des petits producteurs français d'énergie renouvelable.
- Fonctionnement coopératif


*1 - https://midipyrenees.enercoop.fr/actualites/les-chiffres-cles-de-la-cooperative-avril-2017*

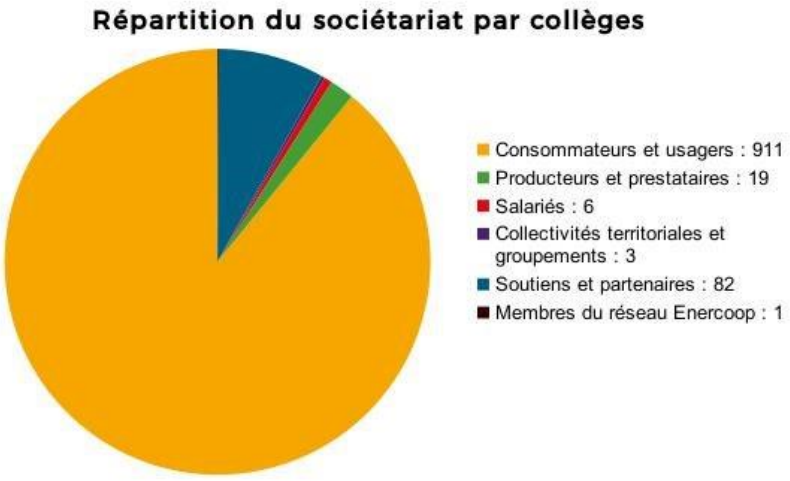
**Problématique**

Les sources d'énergies renouvelables sont intermittentes :

- Non disponible en permanence
- Forte variation de la disponibilité sans possibilité de contrôle

La demande en électricité des utilisateurs varie au cours du temps, et dépend de paramètres comme la météo (température, luminosité, etc.)

**Objectif :** Mettre en adéquation l'offre et la demande en prédisant la demande en électricité.

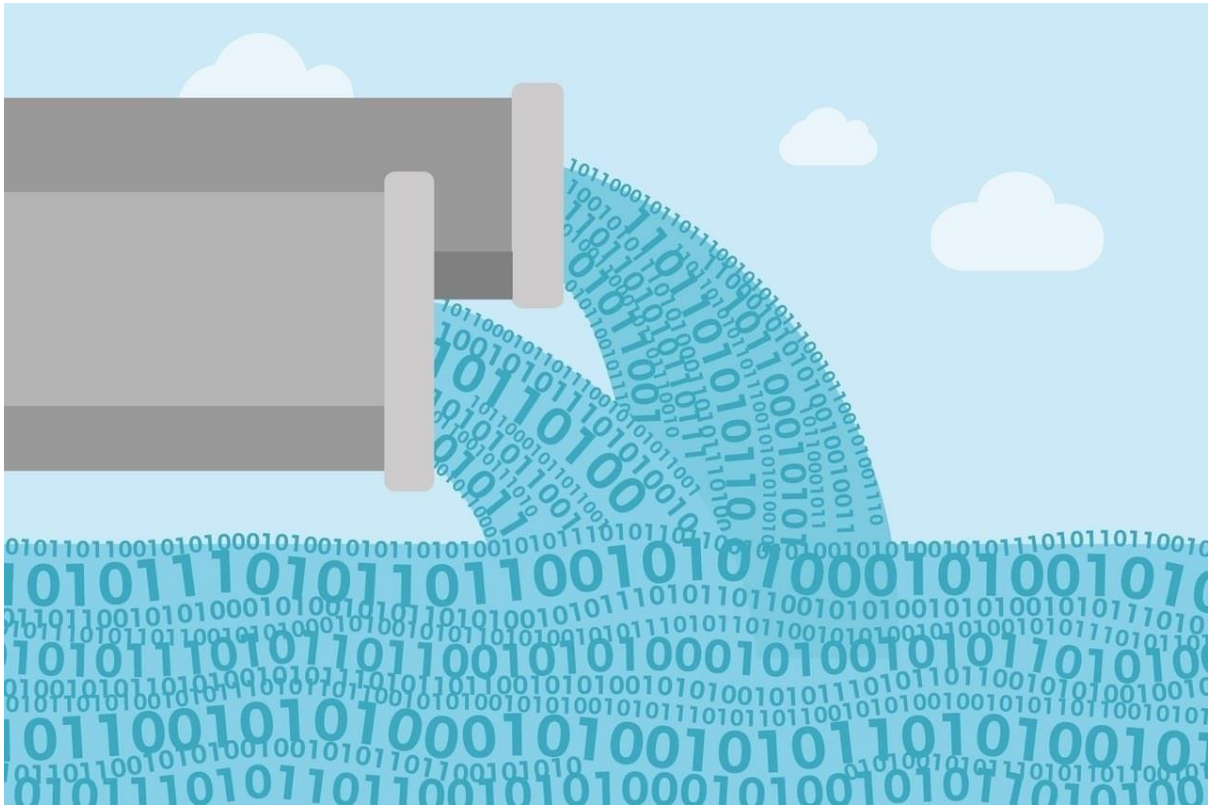



*2 - https://midipyrenees.enercoop.fr/actualites/les-chiffres-cles-de-la-cooperative-avril-2017*

**Plan :**

1. Sources et retraitement des données
2. Mission 1 : Corrigez les données de consommation mensuelles de l'effet température (dues au chauffage électrique) en utilisant une régression linéaire.
3. Mission 2 : Effectuez une désaisonnalisation de la consommation que vous aurez obtenue après correction, grâce aux moyennes mobiles.
4. Mission 3 : Effectuez une prévision de la consommation (corrigée de l'effet température) sur un an, en utilisant la méthode de Holt Winters (lissage exponentiel) puis la méthode ARMA sur la série temporelle désaisonnalisée.

## Sources et retraitement des données



**Importation des données énergie**

Source : [Réseau de Transport de l'Electricité (RTE)]

RTE distribue l'électricité en France et produit des données sur son réseau accessibles via l'outil Eco2mix

```
df_energie <- as.data.frame(read.xls('../fichiers_csv/energie.xls'))
colnames(df_energie)
df_energie$Mois <- as.vector(df_energie$Mois)
df_energie$Qualite <- as.vector(df_energie$Qualite)
df_energie$Territoire <- as.vector(df_energie$Territoire)
```

*3 - Import des données et transformation des colonnes en vecteur pour faciliter la manipulation*

```
df_energie <- df_energie %>%
  select(Mois, Territoire, Consommation.totale) %>%
  filter(Territoire == "France")
```

*4 - On ne s'intéresse qu'aux données de la France entière*

```
mois = rep(c('JAN','FEV','MAR','AVR','MAI','JUIN','JUIL','AOUT','SEP','OCT','NOV','DEC'),8)
annee = c(rep('2010',12),rep('2011',12),rep('2012',12),rep('2013',12),rep('2014',12),rep('2015',12),rep('2016',12)
```

```
df_energie <- df_energie[1:96,]
df_energie <- cbind.data.frame(df_energie,mois,annee)
df_energie$mois <- as.vector(df_energie$mois)
df_energie$annee <- as.vector(df_energie$annee)
```

*5 - On prépare le data.frame pour un pivot :*

- *Création des variables mois et année*
- *Ajout des variables au data.frame de base*
- *Transformation des variables en vecteur pour faciliter la manipulation*

```
df_energie <- df_energie[,c("annee", "mois", "Consommation.totale", "Mois","Territoire")]
```

```
df_energie <- cast(df_energie,annee ~ mois,value='Consommation.totale',sum)
df_energie <- df_energie[,c("annee", 'JAN','FEV','MAR','AVR','MAI','JUIN','JUIL','AOUT','SEP','OCT','NOV','DEC')]
```

*6 - **Pivot : Utiliser une valeurs unique située dans une colonne pour construire le nouveau data.frame***

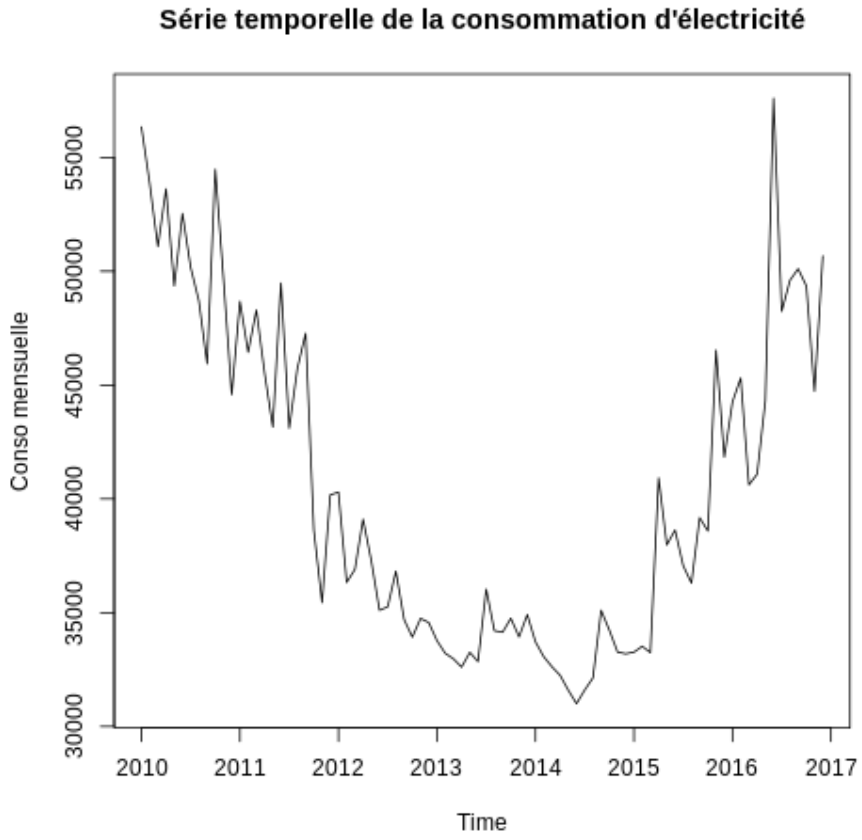
```
df_energie <- df_energie[1:7,]
```

```
df_energie
```

| annee | JAN | FEV | MAR | AVR | MAI | JUIN | JUIL | AOUT | SEP | OCT | NOV | DEC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2010 | 56342 | 48698 | 48294 | 38637 | 37284 | 34567 | 36031 | 33069 | 35104 | 40918 | 46532 | 57600 |
| 2011 | 53873 | 45937 | 45543 | 35442 | 35107 | 33771 | 34204 | 32625 | 34230 | 37977 | 41837 | 48241 |
| 2012 | 51086 | 54476 | 43156 | 40176 | 35257 | 33219 | 34141 | 32247 | 33269 | 38628 | 44260 | 49602 |
| 2013 | 53619 | 49639 | 49480 | 40292 | 36821 | 32973 | 34751 | 31591 | 33195 | 37063 | 45310 | 50108 |
| 2014 | 49359 | 44580 | 43104 | 36331 | 34695 | 32608 | 33935 | 31004 | 33266 | 36301 | 40617 | 49350 |
| 2015 | 52536 | 48676 | 45739 | 36898 | 33927 | 33256 | 34912 | 31603 | 33521 | 39170 | 41085 | 44727 |
| 2016 | 50161 | 46440 | 47270 | 39102 | 34746 | 32852 | 33718 | 32132 | 33245 | 38590 | 44293 | 50670 |

*7 - On fera abstraction de l'année 2017 (voir valeur manquante df_base_meteo)*

*Affichage du data.frame*




**Importation des données météo**

Source : [Simulateur DJU CEGIBAT]

CEGIBAT :

Simulateur de DJU : Cet outil, réalisé en partenariat avec Météo France, permet de calculer les degrés jour (DJ ou DJU) chauffage ou climatisation sur une période, une station météo et un seuil de température donnés.

DJU : Le **degré jour** est une valeur représentative de l'écart entre la température d'une journée donnée et un seuil de température préétabli (18 °C dans le cas des DJU ou Degré Jour Unifié). Sommés sur une période, ils permettent de calculer les besoins de chauffage et de climatisation d'un bâtiment.

```
files_to_read <- list.files(path = '../fichiers_csv/meteo_data',full.names = TRUE)
```

```
liste_meteo <- lapply(files_to_read,function(i){
    list_csv <- read.xls(i)


})
```

```
df_meteo <- lapply(liste_meteo,function(i){
    a <- data.frame(i,stringsAsFactors = FALSE)

station <- as.vector(a[1,'X.1'])

mat <- as.matrix(a)
mat <- mat[-c(1,2,3,4,5,6,7),]
colnames(mat) <- mat[1,]
mat <- mat[-1,]
colnames(mat)[1] <- 'Annee'
nb_row <- length(mat[,'Annee'])
mat<- cbind(rep(station,nb_row),mat)
colnames(mat)[1] <- 'Station'
as.data.frame(mat,stringsAsFactors = FALSE)
mat[1:7,] #Valeur manquantes en 2017
})
```

*8 -*

- *Récupération de tous les fichiers xls par station*
- *Suppression des valeurs non pertinentes*
- *Résultat obtenu : Liste de data.frame par station*

```
df_base_meteo <- ldply(df_meteo, rbind)
df_base_meteo <- rename(df_base_meteo,c('Station' = 'station','Annee'='annee','JAN'='JAN','F\xc9V'='FEV','M
```

```
df_base_meteo[] <- as.vector(unlist(df_base_meteo, use.names = FALSE))
df_base_meteo[-2] <- as.numeric(unlist(df_base_meteo[-2], use.names = FALSE))
```

```
Warning message in eval(expr, envir, enclos):
“NAs introduits lors de la conversion automatique”
```

*9 -*

- *Empilement des data.frames pour en avoir qu'un seul*
- *Conversion en vecteur de la colonne 'stations'*
- *Conversion en vecteurs numériques des colonnes mensuelles*


```
df_base_meteo[] <- as.vector(unlist(df_base_meteo, use.names = FALSE))
df_base_meteo[-2] <- as.numeric(unlist(df_base_meteo[-2], use.names = FALSE))
```

```
Warning message in eval(expr, envir, enclos):
“NAs introduits lors de la conversion automatique”
```

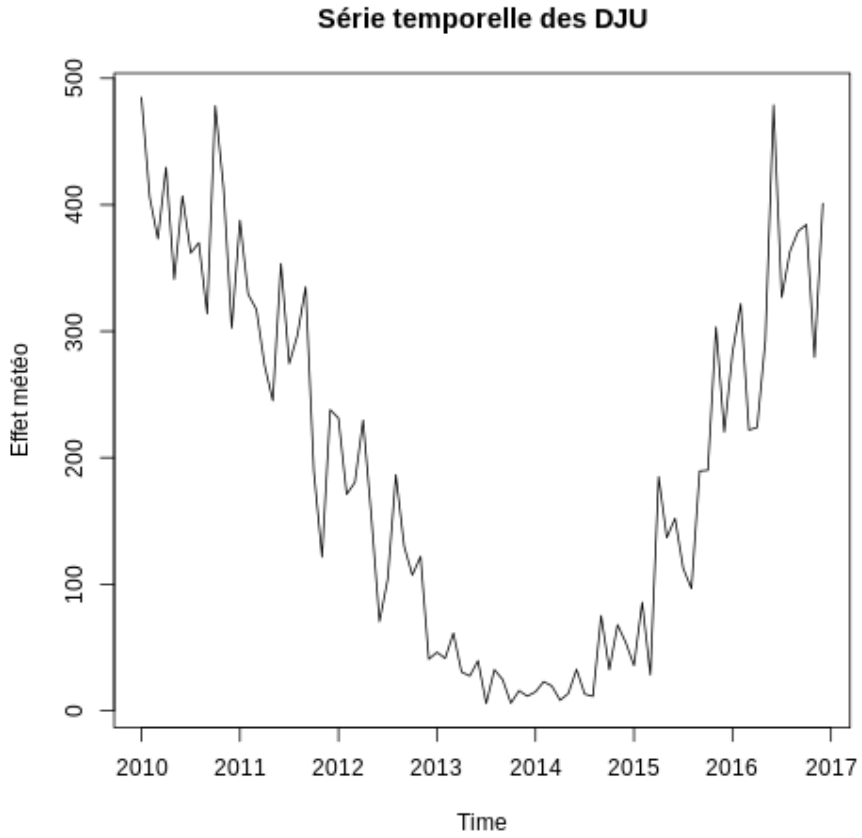

```
df_base_meteo <- ddply(df_base_meteo,~annee,summarise, JAN = mean(JAN),
      FEV = mean(FEV),
      MAR = mean(MAR),
      AVR = mean(AVR),
MAI = mean(MAI),
JUN = mean(JUN),
JUI = mean(JUI),
AOU = mean(AOU),
SEP = mean(SEP),
OCT = mean(OCT),
NOV = mean(NOV),
DEC = mean(DEC))
```

*10 - Moyenne DJU en France par mois et par an*

```
df_base_meteo
```

| annee | JAN | FEV | MAR | AVR | MAI | JUN | JUI | AOU | SEP | OCT | NOV | DEC |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2010 | 484.9267 | 369.9756 | 317.2278 | 189.6422 | 154.08889 | 40.69889 | 5.656667 | 22.85444 | 75.07222 | 184.95667 | 303.3189 | 478.8556 |
| 2011 | 406.2789 | 313.8867 | 273.3411 | 121.5056 | 70.45333 | 46.10111 | 32.323333 | 19.43889 | 32.31667 | 136.76111 | 220.3189 | 326.8511 |
| 2012 | 373.1278 | 478.2522 | 245.1367 | 237.8778 | 103.84111 | 41.35444 | 24.552222 | 8.21000 | 67.81889 | 152.04111 | 282.9667 | 363.1200 |
| 2013 | 429.6233 | 413.0800 | 353.4844 | 231.2144 | 186.51889 | 61.25778 | 5.921111 | 13.47556 | 53.80667 | 112.49667 | 321.7444 | 379.1200 |
| 2014 | 341.1244 | 302.5311 | 274.3767 | 171.0933 | 130.10000 | 30.49000 | 15.743333 | 32.52667 | 35.64222 | 96.57444 | 221.9400 | 384.3467 |
| 2015 | 406.9656 | 387.6867 | 296.7800 | 180.4811 | 107.18111 | 27.36556 | 11.420000 | 13.06667 | 85.59333 | 189.47000 | 223.8689 | 279.3678 |
| 2016 | 361.8989 | 329.3178 | 334.9833 | 229.5767 | 121.88111 | 39.40111 | 14.776667 | 11.26111 | 28.29556 | 190.01556 | 292.3822 | 401.0267 |

*11 - Affichage des données météo*




## Correction de l'effet météo

1. Décomposition de la tendance et de l'effet météo dans des bases que l'on pourra estimer par régression linéaire.
2. On effectue la régression linéaire
3. On calcule la série corrigée de l'effet météo.

```
# Création de la base tendancielle et saisonnière
## Base tendancielle
t=1:84
```

```
## Base saisonnière
### On va crée 12 indicatrices qui vont contenir des 1 sur chacun des mois

for (i in 1:12)
{
  su=rep(0,times=12)
  su[i]=1
  s=rep(su,times=7)
  assign(paste("s",i,sep=""),s)
}
```

*12 - Décomposition de la tendance et de l'effet météo dans des bases que l'on pourra estimer par régression linéaire.*

```
reg=lm(log_df_energie_ts~t+s1+s2+s3+s4+s5+s6+s7+s8+s9+s10+s11+s12-1)
summary(reg)

## Affichage des coefficients
reg$coefficients

## Coefficients initiaux
a=mean(reg$coefficients[2:13])
b=reg$coefficients[1]
c=reg$coefficients[2:13]-mean(reg$coefficients[2:13])
```

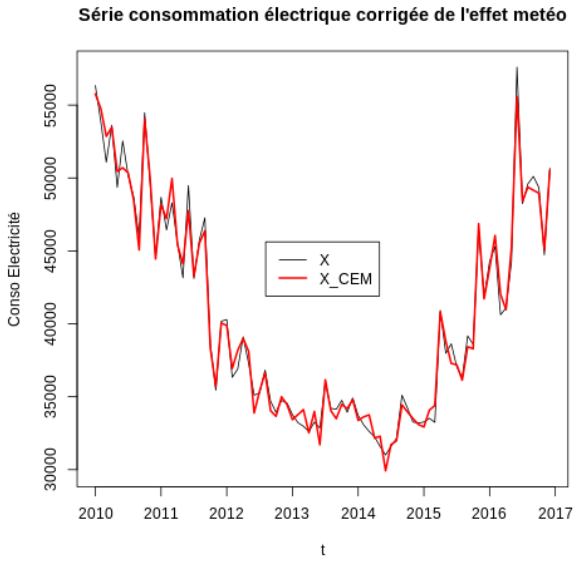
*13 - On effectue la régression linéaire*

| | |
|---|---|
| **t** | -0.00189339730756694 |
| **s1** | 10.6842439434416 |
| **s2** | 10.6575443922021 |
| **s3** | 10.6398597959598 |
| **s4** | 10.6766240087405 |
| **s5** | 10.6521589534717 |
| **s6** | 10.7094346175149 |
| **s7** | 10.6703110638878 |
| **s8** | 10.6785974863623 |
| **s9** | 10.6930507631885 |
| **s10** | 10.6819120557572 |
| **s11** | 10.6668205791248 |
| **s12** | 10.6768954364411 |

*14 - Coefficients*

```
log_df_energie_ts_cem=log_df_energie_ts-(c[1]*s1+c[2]*s2+c[3]*s3+c[4]*s4+c[5]*s5+c[6]*s6+c[7]*s7+c[8]*s8+c[9]*s9+c[10]*s10+
df_energie_ts_cem=exp(log_df_energie_ts_cem)
```

*15 - On calcule la série corrigée de l'effet météo.*




## Correction de la saisonnalité

Nous allons utilisé les moyennes mobiles : Une **moyenne mobile** *M* est une combinaison linéaire d'instants passés et futurs de la série temporelle.

La désaisonnalisation s'effectue en deux phases comprenant chacune 4 étapes :

1. Estimation de la tendance
2. Estimation de la somme composante saisonnière-perturbation
3. Estimation de la composante saisonnière
4. Estimation de la série corrigée des variations saisonnières

---

***Phase 1 :***

---

**Etape 1 : Primo-estimation de la tendance**

```
m2_12=function(x){
  y=(1/12)*stats::filter(x,c(0.5,rep(1,times=11),0.5))
  return(y)
}
t1=m2_12(log_df_energie_ts_cem)
```

**Etape 2 : Primo-estimation de la somme composante saisonnière-perturbation**

```
sig1=log_df_energie_ts_cem-t1
```

**Etape 3 : Primo-estimation de la composante saisonnière**

```
m3=function(x){

  y=(1/3)*stats::filter(x,rep(1,times=3))

  return(y)

}
s1=m3(m3(sig1))
shat1=s1-m2_12(s1)
```

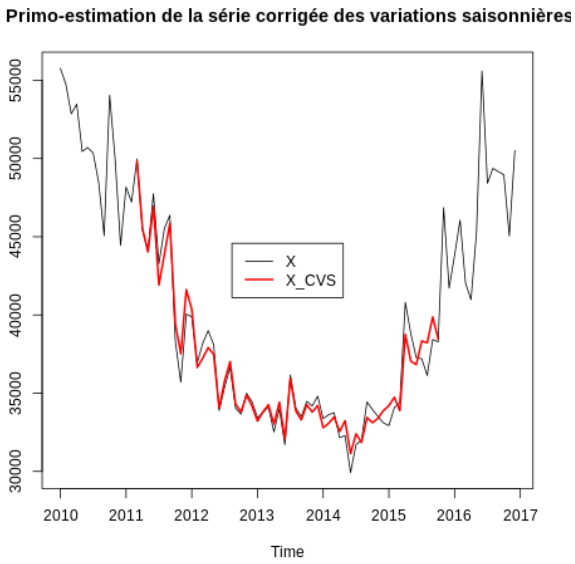
**Etape 4 : Primo-estimation de la série corrigée des variations saisonnières**

```
log_df_energie_ts_cem_cvs1=log_df_energie_ts_cem-shat1
df_energie_ts_cem_cvs1=exp(log_df_energie_ts_cem_cvs1)

ts.plot(df_energie_ts_cem,df_energie_ts_cem_cvs1,col=c(1,2),lwd=c(1,2))

legend("center",legend=c("X","X_CVS"),col=c(1,2),lwd=c(1,2))
```

*16 - Méthodologie de la phase 1*




*17 - Affichage de l'estimation après la phase 1*

## *Phase 2 :*

**Etape 1 : Estimation définitive de la tendance**

```
m13h=function(x){

  y=(1/16796)*stats::filter(x,c(-325,-468,0,1100,2475,3600,4032,3600,2475,1100,0,-468,-325))

  return(y)

}
t2=m13h(df_energie_ts_cem_cvs1)
```

**Etape 2 : Estimation définitive de la somme composante saisonnière-perturbation (2)**

```
sig2=log_df_energie_ts_cem-t2
```

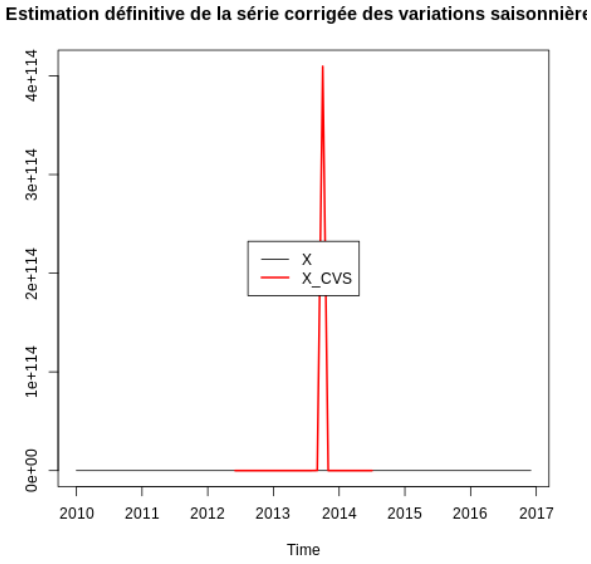
**Etape 3 : Estimation définitive de la composante saisonnière**

```
m5=function(x){
  y=(1/5)*stats::filter(x,rep(1,times=5))
  return(y)
}
s2=m3(m5(sig2))
shat2=s2-m2_12(s2)
```

**Estimation définitive de la série corrigée des variations saisonnières**

```
log_df_energie_ts_cem_cvs2=log_df_energie_ts_cem-shat2
df_energie_ts_cem_cvs2=exp(log_df_energie_ts_cem_cvs2)
```

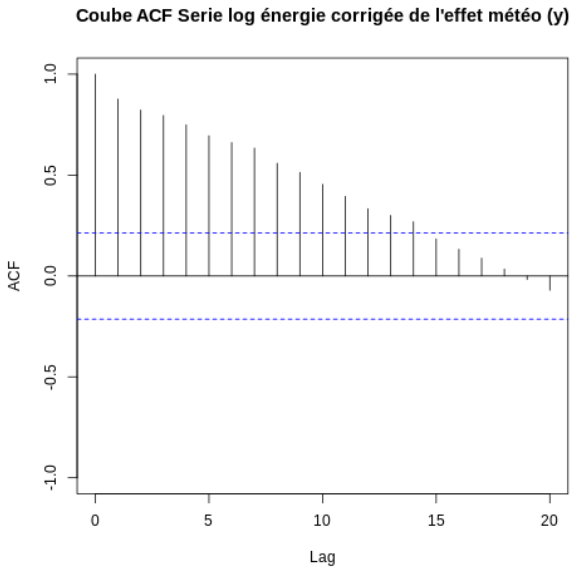
*18 - Méthodologie phase 2*




*19 - Affichage de l'estimation définitive*

# Prédictions


*Affichage de la courbe ACF de la série*




*20 - En statistiques, on aime bien avoir des valeurs indépendantes -- or les séries temporelles ont des valeurs naturellement corrélées. Pour observer les corrélation on utilise une courbe d'autocorrélation (ACF)Il y a trois observations possibles :*

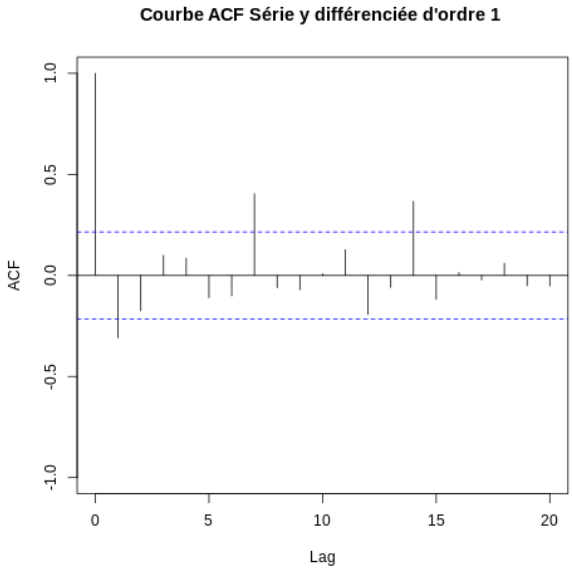
- *l'ACF qui est presque nulle correspond à des données non corrélées (processus stationnaire)*
- *l'ACF tantôt positive, tantôt négative à des données périodiques*
- *Décroissance lente de la courbe ACF (processus non stationnaire)*

La sortie ACF présente une décroissance lente vers 0, ce qui traduit un problème de non-stationnarité.

---

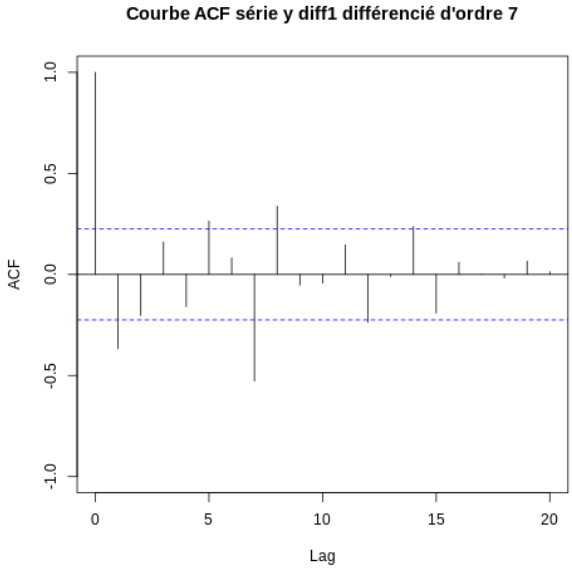
## *Stationnarisation du processus par différentiation d'ordre 1*

---



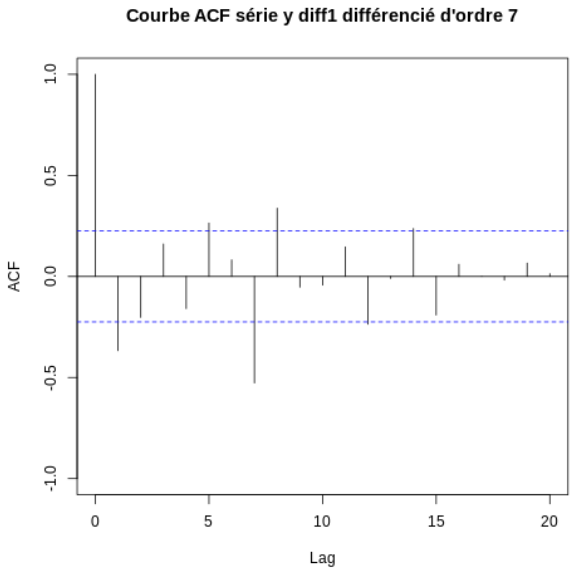

La sortie ACF de la série ainsi différenciée présente encore une décroissance lente vers 0 pour les multiples de 7.

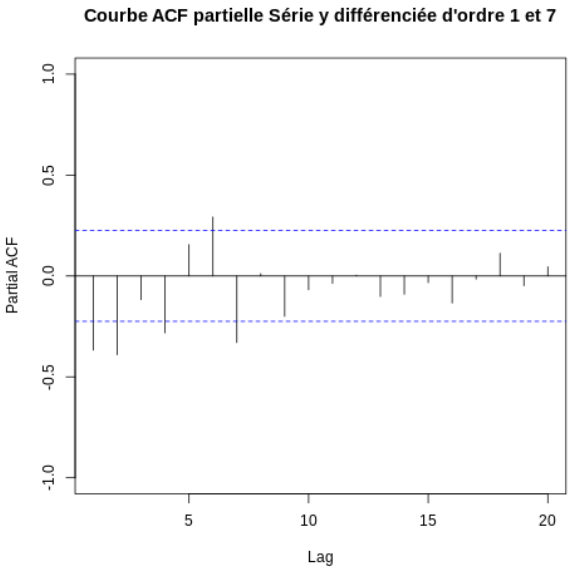
## *Stationnarisation du processus par différentiation d'ordre 1 et d'ordre 7*




La sortie ACF de la série doublement différenciée semble pouvoir être interprétée comme un autocorrélogramme simple empirique. On identifiera donc un modèle ARMA sur la série (I−B)(I−B7)ln(Xt) .

## *Identification, estimation et validation de modèlesen s'appuyant sur les corrélogrammes simples et partiels estimés*






## *Modèles potentiels*

```
Series: y
ARIMA(1,1,1)(1,1,1)[7]

Coefficients:
          ar1      ma1     sar1    sma1
      0.0989  -0.5870  -0.7047  0.2333
s.e.  0.1795   0.1338   0.2370  0.3339

sigma^2 estimated as 0.003244:  log likelihood=110.64
AIC=-211.29   AICc=-210.43   BIC=-199.63

Training set error measures:
                      ME       RMSE        MAE        MPE      MAPE      MASE
Training set 0.00840153 0.05272918 0.03832083 0.07864327 0.3619782 0.2794862
                    ACF1
Training set -0.01281039
```

| | ar1 | ma1 | sar1 | sma1 |
|---|---|---|---|---|
| **t.stat** | 0.550738 | -4.388501 | -2.973915 | 0.698735 |
| **p.val** | 0.581813 | 0.000011 | 0.002940 | 0.484717 |

| Retard | p-value |
|---|---|
| 5 | 0.56472 |
| 10 | 0.61219 |
| 15 | 0.47034 |
| 20 | 0.68206 |

*21 - t-test : Ce modèle ayant des paramètres non significatifs, on en teste un second.*

*Box-cox : Les p-valeurs ne sont pas inférieures au niveau de test de 5% donc on ne rejette pas l'hypothèse nulle qui est : Le résidu suit un bruit blanc*

```
Series: y
ARIMA(0,1,1)(1,1,1)[7]

Coefficients:
          ma1     sar1    sma1
      -0.5284  -0.7029  0.2328
s.e.   0.0966   0.2432  0.3415

sigma^2 estimated as 0.003213:  log likelihood=110.49
AIC=-212.98   AICc=-212.42   BIC=-203.66

Training set error measures:
                      ME       RMSE        MAE        MPE      MAPE      MASE
Training set 0.008203305 0.05284306 0.03823245 0.07674292 0.3611799 0.2788416
                     ACF1
Training set 0.02017024
```

| | ma1 | sar1 | sma1 |
|---|---|---|---|
| **t.stat** | -5.472472 | -2.890699 | 0.681856 |
| **p.val** | 0.000000 | 0.003844 | 0.495330 |

| Retard | p-value |
|---|---|
| 5 | 0.52125 |
| 10 | 0.63756 |
| 15 | 0.49246 |
| 20 | 0.71822 |

*22 - Modèle 2 : On retire l'ordre du paramètre le moins significatif du modèle 1*

*t-test : Ce modèle ayant des paramètres non significatifs, on en teste un second.*

*Box-cox : Les p-valeurs ne sont pas inférieures au niveau de test de 5% donc on ne rejette pas l'hypothèse nulle qui est : Le résidu suit un bruit blanc*

```
Series: y
ARIMA(0,1,1)(1,1,0)[7]

Coefficients:
          ma1     sar1
      -0.5227  -0.5331
s.e.   0.0957   0.0998

sigma^2 estimated as 0.003197:  log likelihood=110.32
AIC=-214.64   AICc=-214.31   BIC=-207.65

Training set error measures:
                      ME       RMSE       MAE        MPE      MAPE      MASE
Training set 0.00889992 0.05307359 0.0380294 0.08323569 0.3591356 0.2773607
                     ACF1
Training set 0.01729725
```

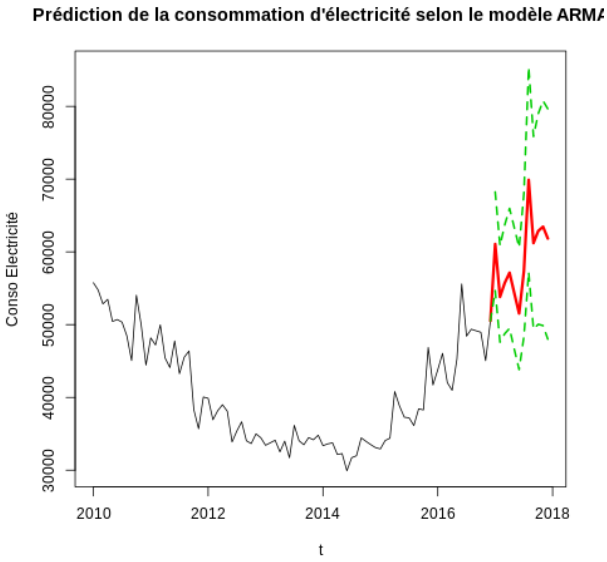
| | ma1 | sar1 |
|---|---|---|
| **t.stat** | -5.461541 | -5.344334 |
| **p.val** | 0.000000 | 0.000000 |

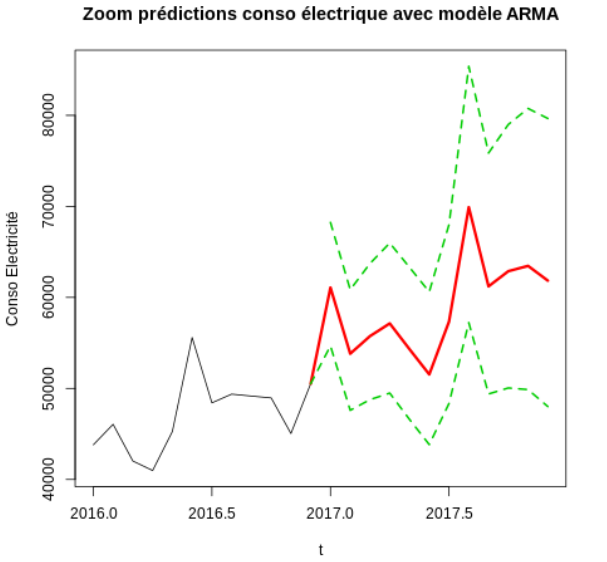
| Retard | p-value |
|---|---|
| 5 | 0.50995 |
| 10 | 0.67542 |
| 15 | 0.65759 |
| 20 | 0.86621 |

*23 - Modèle 3 : On retire l'ordre du paramètre le moins significatif du modèle 2*

*t-test : Ce modèle ayant des paramètres significatifs, on utilisera ce modèle pour nos predictions*

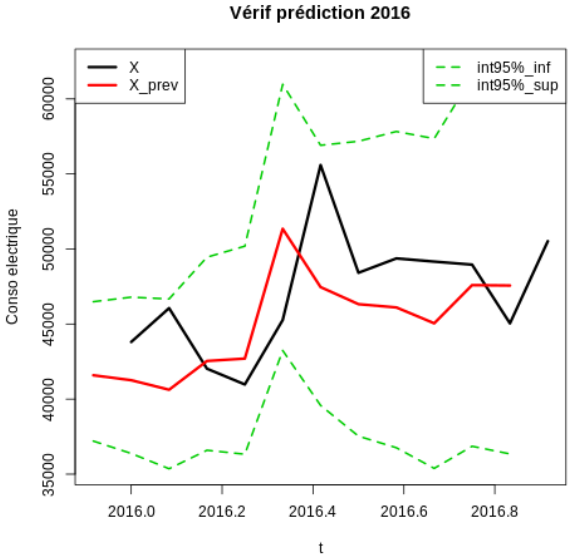
*Box-cox : Les p-valeurs ne sont pas inférieures au niveau de test de 5% donc on ne rejette pas l'hypothèse nulle qui est : Le résidu suit un bruit blanc*




## Conclusion : Analyse à posteriori

*Nous allons vérifier l'efficacité du modèle en tronquant la série de l'année 2016, qu'on cherche ensuite à prévoir à partir de l'historique 2010-2015.*




```
rmse=sqrt(mean((x_a_prevoir-pred_tronc)^2))
rmse

mape=mean(abs(1-pred_tronc/x_a_prevoir))*100
mape
```

4072.80526653024

7.16187251995351

*24 - Tests complémentaires*