On cherche à prédire la chronique des demandes d'indémnité du chomage aux USA par les chroniques de popularité de recherches Google en liens avec le chômage. La chronique *initialia.claims* du package `bsts` contient ainsi 10 recherches Google. Nous aplliquons ainsi le modèle complet tel que décrit précédement.

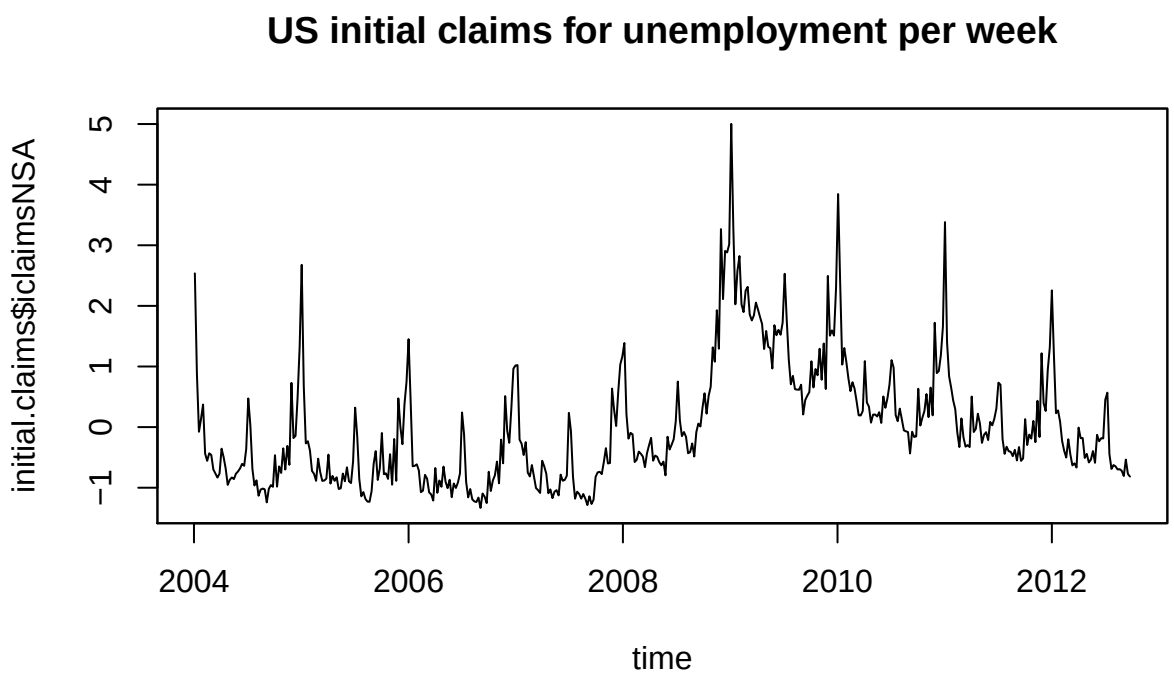
```
  library(lubridate)
  library(bsts)
  library(ggplot2)
  library(reshape2)
  library(zoo)
```

nous tirons 2000 échantillons avec un burn-in de 400 tirages. Pour la *spicke and slab prior* nous prenons une taille de modèle à priori de 5, soit $\pi_t = \pi = 5/10$

```
data(iclaims)
names(initial.claims)
```

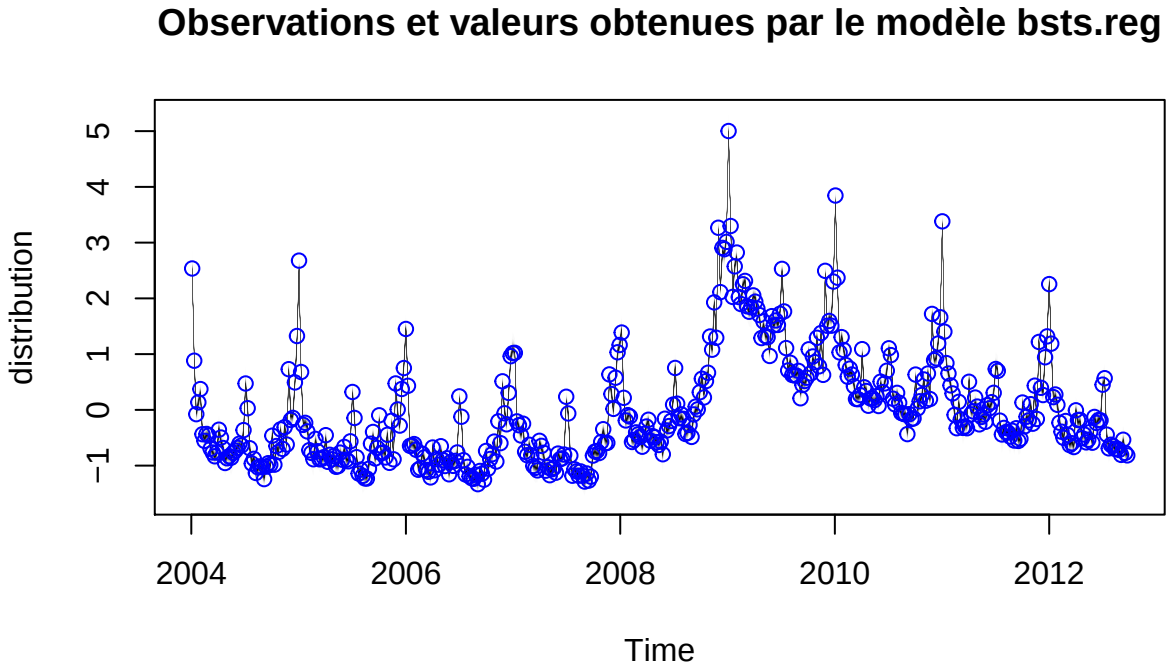
```
##  [1] "iclaimsNSA"                  "michigan.unemployment"
##  [3] "idaho.unemployment"          "pennsylvania.unemployment"
##  [5] "unemployment.filing"         "new.jersey.unemployment"
##  [7] "department.of.unemployment" "illinois.unemployment"
##  [9] "rhode.island.unemployment"   "unemployment.office"
## [11] "filing.unemployment"
```

```
plot(initial.claims$iclaimsNSA,main="US initial claims for unemployment per week",xlab = "time")
```




```
# ajout au modèle de mu_t = mu_t-1 + delta_t-1 N(0,sigma.level).^2)
#        dela_t = delta_t-1 + N(0, sigma.slope^2)
ss <- AddLocalLinearTrend(list(), initial.claims$iclaimsNSA)
# ajout au modèle d'une composante stationnaire annuelle
ss <- AddSeasonal(ss, initial.claims$iclaimsNSA, nseasons = 52)

# modele avec une regression avec une distribution à priori spike and slab
# avec les paramètres par defaut ici
# comme les recherches google sont similaire, on attend un taille de
# modèle inférieure à 10, on prend 5.
```

```
bsts.reg <- bsts(iclaimsNSA ~ ., state.specification = ss, data =
                 initial.claims,
                 expected.model.size = 5,
                 niter = 2000, ping=0, seed=2016)

# affichage de la serie et des valeurs obtenues selon le modèle
plot(bsts.reg,main="Observations et valeurs obtenues par le modèle bsts.reg")
```



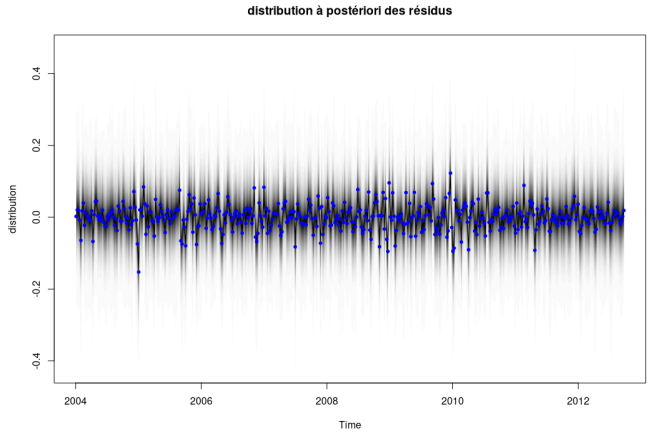

```
# on attend 400 tirage avant de les utiliser pour estimer les distributions
burn <- SuggestBurn(0.2, bsts.reg)
# on verifie ainsi que le modèle correspond bien à la serie à prédire.
```

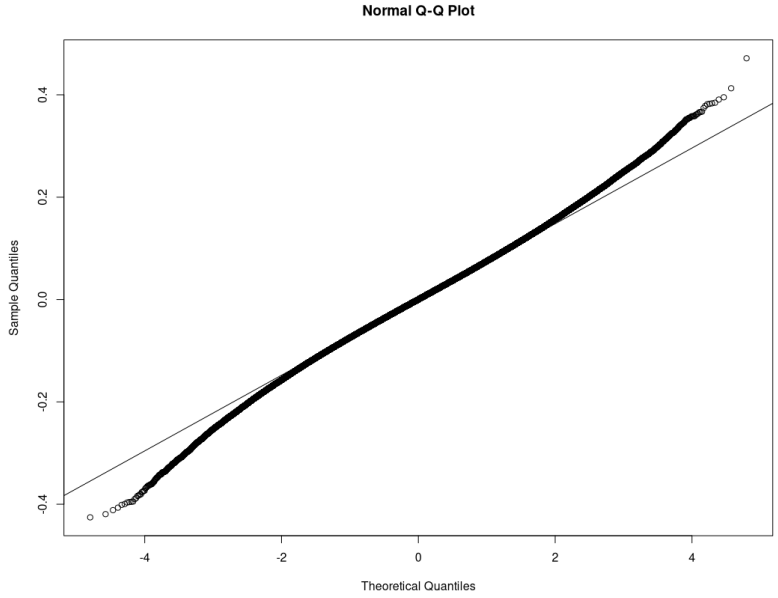
On peut vérifier le caractère gaussien de la distribution des résidus du modèle.

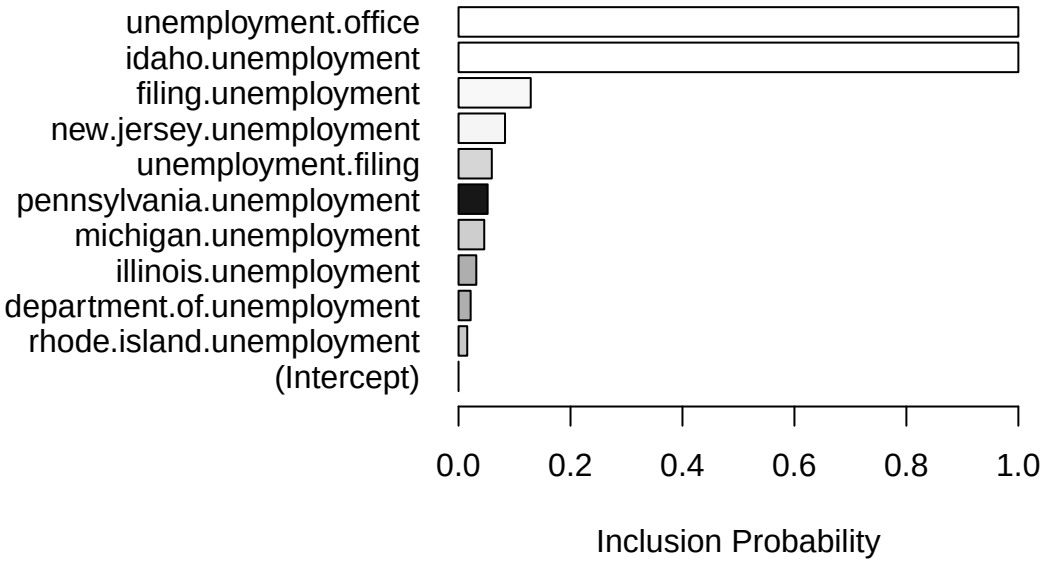
```
r <- residuals(bsts.reg,burn = SuggestBurn(0.2, bsts.reg))
PlotBstsResiduals(bsts.reg,burn,main="distribution à postériori des résidus")
qqnorm(r)
qqline(r)
```

Après les tirages succéssifs, on peut estimer la probabilité à postérior des variables explicatives dans le modèle.

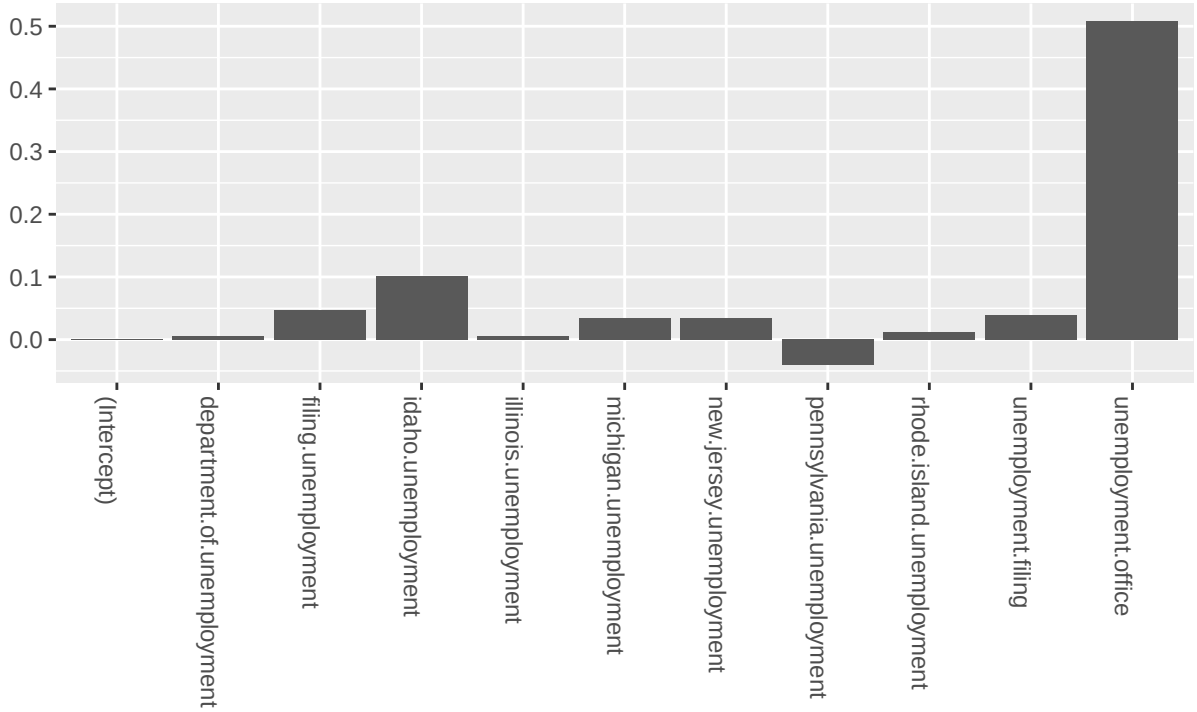
```
plot(bsts.reg, "coef",burn = SuggestBurn(0.3, bsts.reg))
```


distribution à postériori des résidus
distribution
0.4
0.2
0.0
-0.2
-0.4
2004
2006
2008
2010
2012
Time


Normal Q-Q Plot
Sample Quantiles
0.4
0.2
0.0
-0.2
-0.4
-4
-2
0
2
4
Theoretical Quantiles


unemployment.office
idaho.unemployment
filing.unemployment
new.jersey.unemployment
unemployment.filing
pennsylvania.unemployment
michigan.unemployment
illinois.unemployment
department.of.unemployment
rhode.island.unemployment
(Intercept)
0.0
0.2
0.4
0.6
0.8
1.0
Inclusion Probability

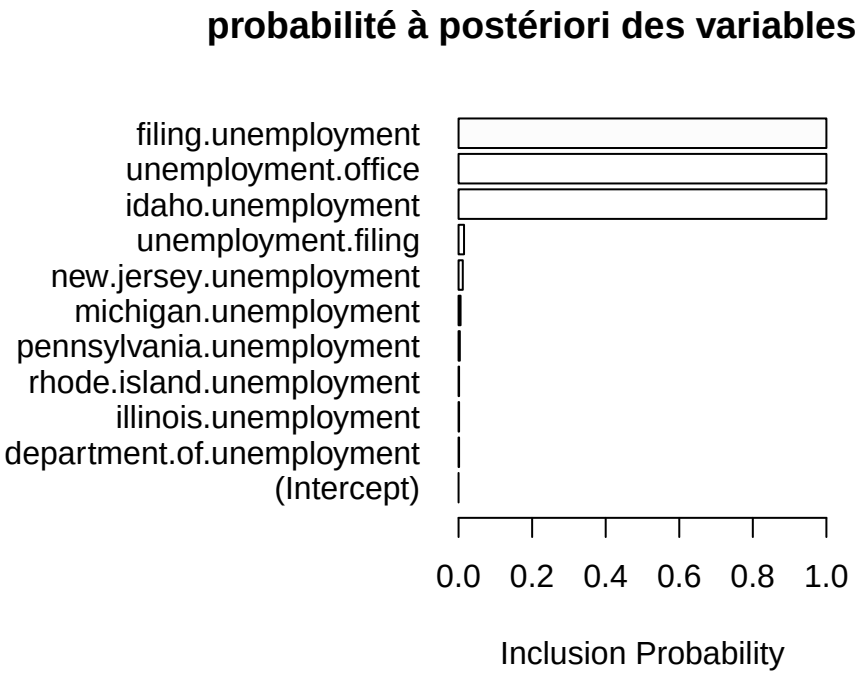
Les chroniques `unemployment.office` et `idaho unemployment` ont une probabilité à postériori trés forte d'être dans le modèle. Pour les estimations des coéfficients de la regression :



Les chroniques `unemployment.office` et `idaho unemployment` ont ainsi les probabilités estimées à postériori de contribuer au modèle les plus importantes.

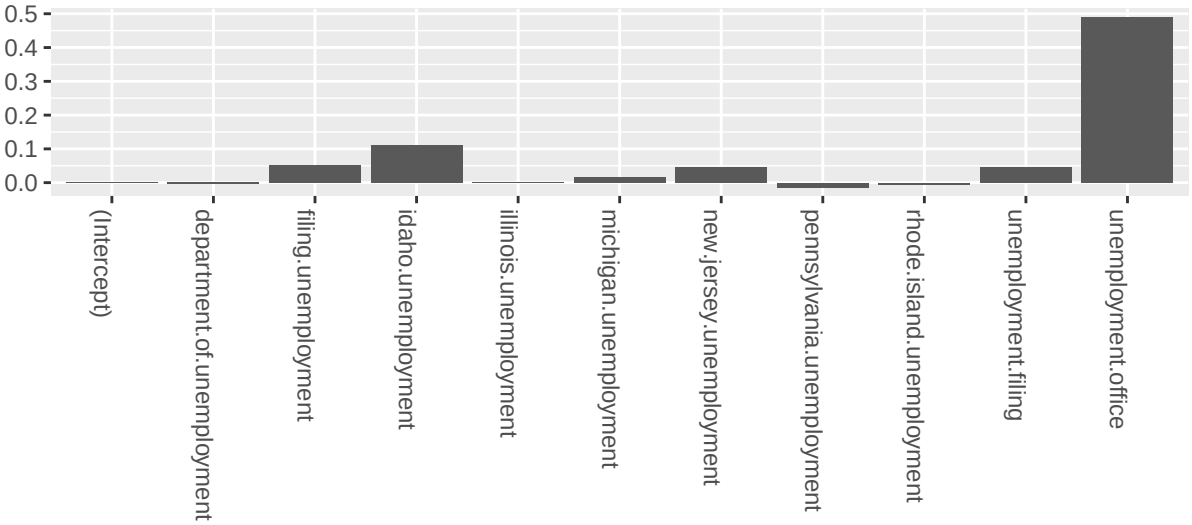
On peut supposer que le modèle n'a que 3 composantes avec `unemployment.office` et `idaho unemployment` obligatoirement incluses. Nous pouvons essayer un modèle avec une telle distribution à priori en imposant à la distribition à priori de β de prendre en compte ces chroniques.

```
prior.spikes <- rep(0.1,11)
prior.spikes[3] <- 1
prior.spikes[11] <- 1

# on génère à partir de ces coéficient à priori la distribution spike and slab à priori
prior <- SpikeSlabPrior(x=model.matrix(iclaimsNSA ~ ., data=initial.claims),
                        y=initial.claims$iclaimsNSA,
                        expected.model.size = 5,
                        prior.inclusion.probabilities = prior.spikes)

bsts.reg.priors <- bsts(iclaimsNSA ~ ., state.specification = ss,
                        data = initial.claims,
                        niter = 2000,
                        prior=prior,
                        ping=0, seed=2016)
burn <- SuggestBurn(0.3, bsts.reg.priors)

# probabilité à postériori d'inclusion des variables
plot(bsts.reg.priors, "coef",burn = SuggestBurn(0.3, bsts.reg),
     main="probabilité à postériori des variables explicatives")
```



Ainsi les 3 variable explicatives conservées à postériori sont s`unemployment.office`,`idaho unemployment`, et `filing.unemployment`. Les autres ont une probabilité trés faible de faire partie du modèle.

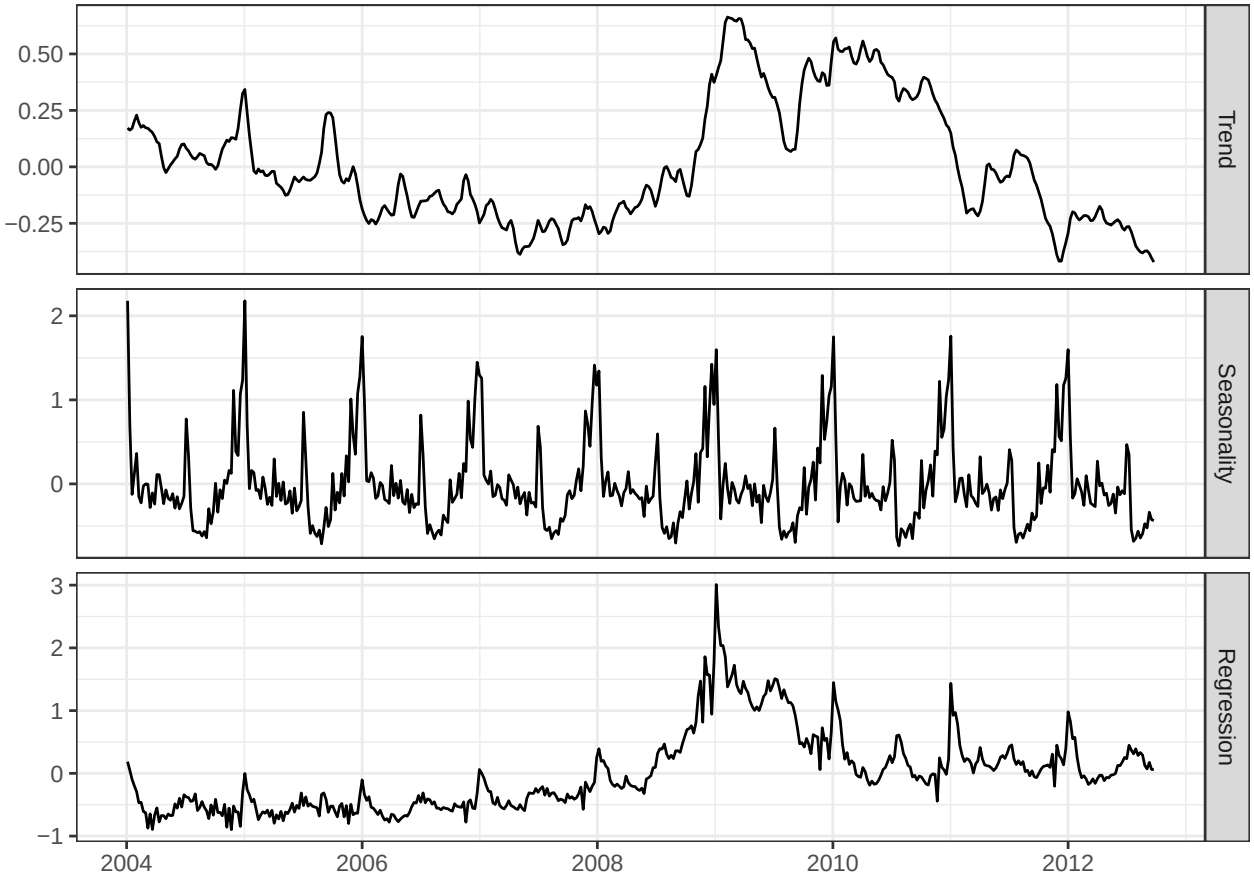
```
# coefficients
coeff <- data.frame(melt(apply(bsts.reg.priors$coefficients[-(1:burn),], 2, PositiveMean)))
coeff$Variable <- as.character(row.names(coeff))
ggplot(data=coeff, aes(x=Variable, y=value)) +
  geom_bar(stat="identity", position="identity") +
  theme(axis.text.x=element_text(angle = -90, hjust = 0)) +
  xlab("") + ylab("")
```



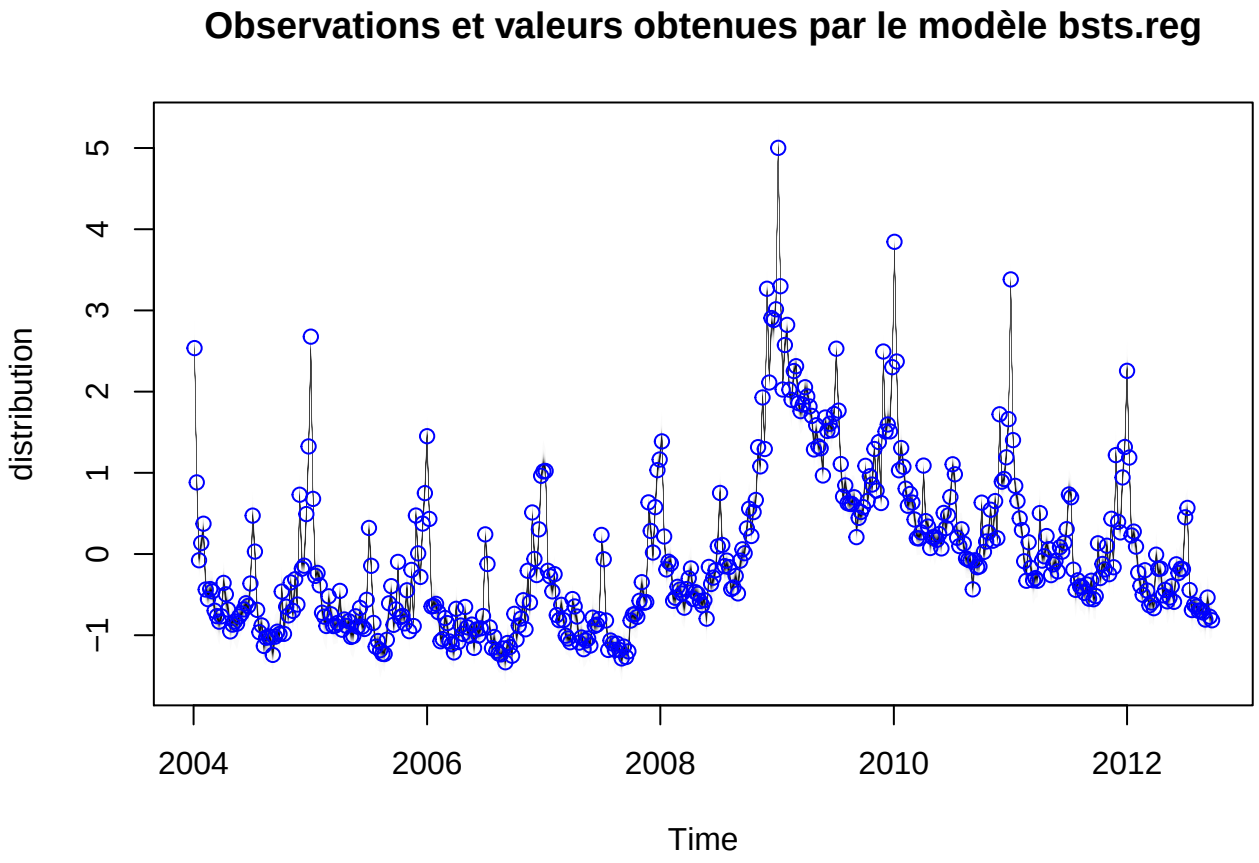
Nous pouvons afficher les composantes du modèle.

```
# composantes du modèle
components.withreg <- cbind.data.frame(
  colMeans(bsts.reg.priors$state.contributions[-(1:burn),"trend",]),
  colMeans(bsts.reg.priors$state.contributions[-(1:burn),"seasonal.52.1",]),
```

```
  colMeans(bsts.reg.priors$state.contributions[-(1:burn),"regression",]),
  as.Date(time(initial.claims)))
names(components.withreg) <- c("Trend", "Seasonality", "Regression", "Date")
components.withreg <- melt(components.withreg, id.vars="Date")
names(components.withreg) <- c("Date", "Component", "Value")

ggplot(data=components.withreg, aes(x=Date, y=Value)) + geom_line() +
  theme_bw() + theme(legend.title = element_blank()) + ylab("") + xlab("") +
  facet_grid(Component ~ ., scales="free") + guides(colour=FALSE)
```



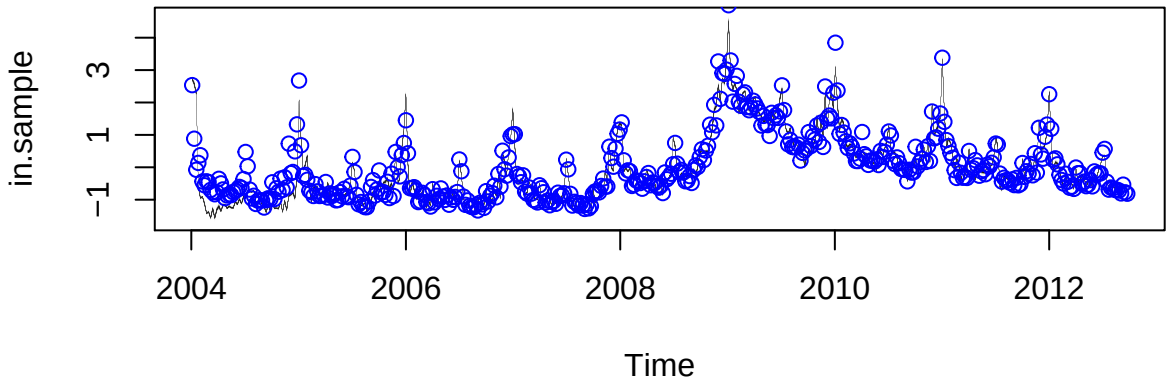

```
plot(bsts.reg.priors,main="Observations et valeurs obtenues par le modèle bsts.reg")
```



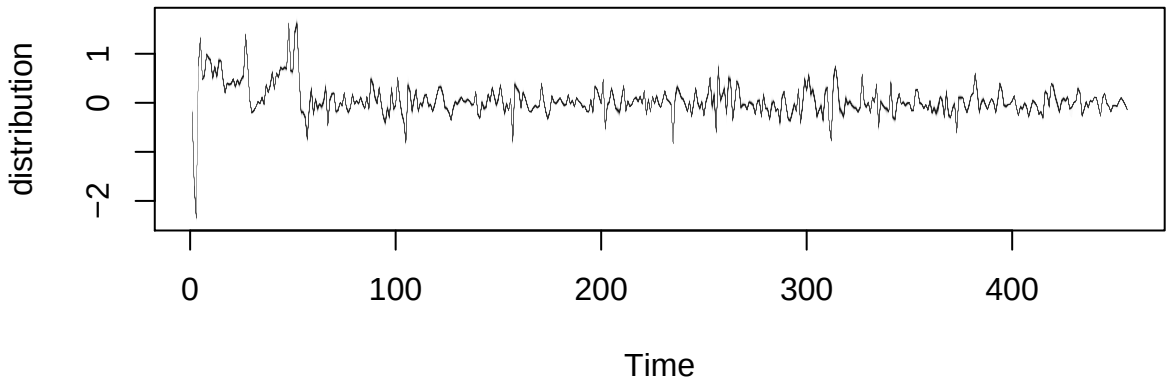
On voit que les données subissent une rupture de tendance durant l'année 2009, probablement dues à la crise économique. Le modèle réussis s'adapter à cette rupture de tendance.

Nous allons considérer comme évaluation de l'erreur de prédiction la somme cumulative des erreurs de prédiction de t sachant $t-1$.

```
# affichage prediction pour le prochain temps
PlotBstsForecastDistribution(bsts.reg.priors)
```



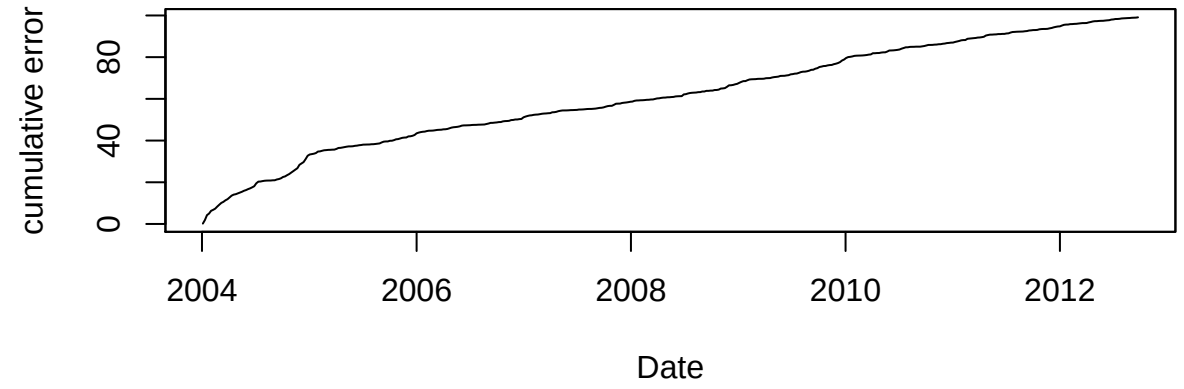

```
# evolution de l'erreur au cours du temps
# renvoit la distribution postérieure de l'erreur de prediction pour t sachant t-1
errors <- bsts.prediction.errors(bsts.reg.priors, burn = burn,standardize = TRUE)$in.sample
PlotDynamicDistribution(errors)
```




```
## NULL
```

L'erreur de prédiction pour les premier t est importante. nous pouvons nous intérresser également aux erreurs de prédiction pour le temps suivant cumulées.

```
cumulative.errors <- matrix(nrow = 1, ncol = ncol(errors))
cumulative.errors[1,] <- cumsum(abs(colMeans(errors)))
idx <- bsts.reg.priors$timestamp.info$timestamps
plot(zoo(cumulative.errors[1, ], order.by = idx),xlab="Date",ylab="cumulative error")
```



Nous pouvoir voir ainsi que le modèle a su bien s'adapter à la rupture de tendance de 2009.