

ÉCOLE DES MINES DE NANCY



Using Bayesian approach in Time Series

SORRY ARIMA, BUT I'M GOING BAYESIAN

Author :
Pierre GAUTHIER

TUTEUR : DENIS
VILLEMONAIS

9 décembre 2019

Table des matières

1 Modélisation avec les processus ARIMA	3
1.1 Généralités sur les séries temporelles	3
1.2 Les processus AR et MA	4
1.3 Le modèle SARIMA	5
1.4 Détermination des paramètres du modèle SARIMA : Méthode de Box et Jenkins	6
1.5 Simulation d'une chronique avec le modèle SARIMA	6
2 Approche Bayésienne	12
2.1 Méthode de Monté-Carlo et Échantillonnage de Gibbs	12
2.2 Le modèle d'espace d'état	15
2.3 Filtre de Kapman	16
2.4 Utilisation d'une loi à priori <i>spike and slab</i>	17
2.5 Entrainement des paramètres	18
3 Utilisation du modèle sur la chronique <i>AirPassenger</i>	20
4 Prédiction des données du chômage avec la <i>spike and slab prior</i>	27
5 Comparaison de l'approche bayésienne et de l'approche fréquentiste	39

Introduction

Nous allons dans le présent travail explorer les solution présentées par le post de Kim Larsen sur le blog de *MultiThreaded* [8]. Les résultats de ce post s'appuient fortement sur [10].

Nous allons tout d'abord présenter les méthodes de prédiction usuelles ARIMA pour ensuite étudier l'approche bayésienne.

1 Modélisation avec les processus ARIMA

1.1 Généralités sur les séries temporelles

Nous allons rappeler brièvement des éléments du cours de modélisation des séries temporelles par des modèles ARIMA tels que introduits par F. Sur [12].

On définit tout d'abord une série temporelle par une suite d'observation d'une variable X_t au cours du temps comme sur la figure 1.1

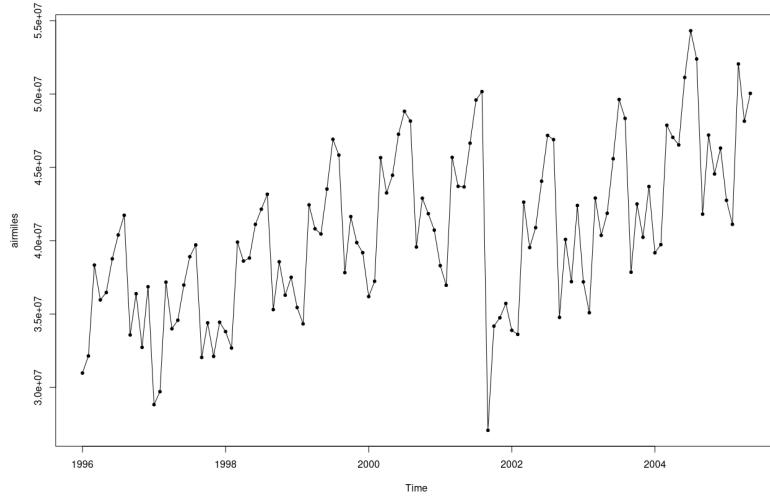


FIGURE 1 – chronique *airmiles* du package *TSA* qui donne le nombre de passagers transportés sur les lignes aériennes aux USA de janvier 1996 à mai 2007

Un processus aléatoire d'une suite de variable (X_1, \dots, X_T) est stationnaire au second ordre si il vérifie les propriétés suivantes :

$$\left\{ \begin{array}{l} \forall t, \mathbb{E}(X_t) = m \quad (\text{moyenne constante}) \\ \forall (t, s), \text{cov}(X_t, X_s) = \gamma(|t - s|), \quad (\text{autocovariance symétrique, invariante par translation}) \\ \forall t, \text{var}(X_t) = \gamma(0) \quad (\text{homoscédasticité}) \end{array} \right.$$

Un bruit blanc gaussien avec variable (ϵ_t) i.i.d est un cas particulier de bruit blanc fort vérifiant :

$$\left\{ \begin{array}{l} \forall t, \mathbb{E}(\epsilon_t) = 0 \\ \forall t, \text{Var}(\epsilon_t) = \sigma^2 \end{array} \right.$$

Autocorrélation ACF

On définit pour l'étude des séries temporelles le coefficient d'autocorrection $\rho(h)$ dit aussi ACF (*auto-correlation function*) :

$$\forall h \geq 0, \rho(h) = \frac{\text{Cov}(X_t, X_{t+h})}{\sqrt{\text{Var}(X_t) \text{Var}(X_{t+h})}} = \frac{\gamma(h)}{\gamma(0)}$$

On utilise en pratique le corrélogramme empirique $\hat{\rho}(h)$ pour identifier les propriétés de la série temporelle

$$\hat{\rho}(h) = \frac{T}{T-h} \frac{\sum_{t=h+1}^T (X_t - \bar{X}_T)(X_{t-h} - \bar{X}_T)}{\sum_{t=1}^T (X_t - \bar{X}_T)^2} \quad (1)$$

Autocorrélation partielle PACF

Dans l'idée d'un modèle auto-projectif nous construisons une prévision linéaire sur la valeur X_t sachant X_{t-1}, \dots, X_{t-h} :

$$\mathbb{E}(X_t | X_{t-1}, \dots, X_{t-h}) = a_1(h)X_{t-1} + \dots + a_h(h)X_{t-h}$$

$r(h) = a_h(h)$ est la corrélation partielle d'ordre h . On montre ainsi également que

$$a_h(h) = \text{corr}(X_t - \mathbb{E}(X_t | X_{t-1}, \dots, X_{t-h+1}), X_{t-h} - \mathbb{E}(X_{t-h} | X_{t-1}, \dots, X_{t-h+1}))$$

1.2 Les processus AR et MA

Les processus autorégressifs AR

(X_t) est un processus autorégressif d'ordre p si il peut s'écrire sous la forme

$$\begin{aligned} X_t &= \sum_{i=1}^p \varphi_i X_{t-i} + \varepsilon_t \quad \forall t \\ \forall i \quad \varphi_i &\in \mathbb{R} \\ (\varepsilon_t) &\text{ un bruit blanc} \end{aligned} \quad (2)$$

en définissant l'opérateur retard B tel que $BX_t = X_{t-1}$ on peut définir un polynôme Φ tel que

$$\Phi(B)X_t = \varepsilon_t \text{ avec } \Phi(B) = 1 - \varphi_1 B - \dots - \varphi_p B^p$$

(On peut ajouter la prise en compte d'une moyenne à 2)

On montre que pour un processus $AR[p]$ on a une relation directe entre les coefficients d'autocorrelations avec

$$\forall h > 0 \quad \rho(h) = \sum_{i=1}^p \varphi_i \rho(h-i)$$

Ainsi dans le cas d'un $AR[1]$, $\rho(h) = \varphi^h$ De plus pour le coefficient d'autocorrelation partielle on a :

$$\forall h > p, r(h) = 0 \quad (3)$$

Les processus à moyenne mobile MA

Un processus $MA[q]$ s'exprime sous la forme

$$\begin{aligned} X_t &= \varepsilon_t - \sum_{i=1}^q \theta_i \varepsilon_{t-i} \\ X_t &= \Theta(B)\varepsilon_t \quad \Theta(B) = 1 - \theta_1 B - \theta_2 B^2 - \dots - \theta_q B^q \end{aligned} \quad (4)$$

(On peut de même ajouter une moyenne dans 4)

La fonction d'autocorrelation pour un processus $MA[q]$ est alors :

$$\begin{cases} \frac{\theta_h + \theta_{h+1}\theta_1 + \dots + \theta_q\theta_{q-h}}{1 + \theta_1^2 + \dots + \theta_q^2}, & \text{si } 1 \leq h \leq q \\ 0 & \text{si } h > q \end{cases}$$

La fonction d'autocorélation partielle ne possède pas d'expression simple. Nous nous intéressons seulement au cas d'un Ma[1] où on montre que l'autocorélation partielle est donnée par

$$r(h) = \frac{\theta^h(\theta^2 - 1)}{1 - \theta^{2h+2}}$$

1.3 Le modèle SARIMA

Processus ARMA

Les processus ARMA $[p, q]$ sont ainsi la superposition d'un processus AR $[p]$ et d'un processus AM $[q]$, et s'écrivent sous la forme

$$\begin{aligned} \forall t, X_t &= \theta_0 + \sum_{i=1}^p \phi_i X_{t-i} - \sum_{j=1}^q \theta_j \varepsilon_{t-j} + \varepsilon_t \\ \Phi(B)X_t &= \theta_0 + \Theta(B)\varepsilon_t \end{aligned} \quad (5)$$

(Le paramètre $\theta_0 \in \mathbb{R}$ peut être pris nul en recentrant la chronique).

Si les racines des polynômes sont de module différent de 1, on peut écrire :

$$X_t = \frac{\Theta(B)}{\Phi(B)}\varepsilon_t = \sum_{j=0}^{\infty} \psi_j \varepsilon_{t-j} \quad (6)$$

Processus ARIMA

Dans le cas où le processus présente une tendance, on applique une différenciation d'ordre d à l'aide de l'opérateur retard tel que $(1 - B)^d X_t$ soit stationnaire. Un processus ARIMA $[p, d, q]$ s'écrit alors sous la forme

$$\Phi(B)(1 - B)^d X_t = \theta_0 + \Theta(B)\varepsilon_t \quad (7)$$

Processus SARIMA

Les processus SARIMA permettent de traiter des séries temporelles avec une saisonnalité. De manière similaire on peut stationnariser la série par différentiation en prenant $(1 - B^\tau)X_t$ si la série X_t présente une saisonnalité de période τ et utiliser un ARMA saisonnier pour les corrélations de période τ qui apparaissent.

$$\phi(B^\tau)X_t = \theta(B^\tau)\varepsilon_t \quad (8)$$

Le processus SARIMA $(p, d, q)([P, D, Q]_\tau)$ s'écrit alors ainsi :

$$\Phi_p(B)\Phi_P(B^\tau)(1 - B)^d(1 - B^\tau)^D X_t = \theta_0 + \Theta_q(B)\Theta_Q(B^\tau)\varepsilon_t \quad (9)$$

Paramètres des processus

Connaissant les ordres p et q d'un processus ARMA, les paramètres (ϕ_i) , (θ_j) , et σ^2 la variance du bruit blanc sont déterminés par la maximisation de la vraisemblance.

$$\mathcal{L}((X_1, \dots, X_T), \varphi, \theta, \sigma) = p((X_1, \dots, X_T) | \varphi, \theta, \sigma) \quad (10)$$

Par composition linéaires d'éléments gaussiens la vraisemblance est un vecteur gaussien dont la densité est :

$$\mathcal{L}((X_1, \dots, X_T), \varphi, \theta, \sigma) = \frac{1}{(2\pi)^{T/2}\sqrt{\det \Omega}} \exp\left(-\frac{1}{2}X'\Omega^{-1}X\right) \quad (11)$$

où $\sigma^2\Omega$ est la matrice de covariance des X_t qui dépend de φ et θ .

1.4 Détermination des paramètres du modèle SARIMA : Méthode de Box et Jenkins

La méthode de Box et Jenkins est une procédure pour déterminer les paramètres du processus de modélisation d'une série temporelle.

- On commence par transformer la chronique pour qu'elle soit stationnaire.
- Avec le corrélogramme et le corrélogramme partiel de la chronique nous identifions les autocorrélations pour h petit pour déterminer p, q . Ensuite nous identifions les autocorrélations saisonnières pour déterminer P, D, Q
- On estime les paramètres θ, φ, σ . (Et θ_0 si on considère l'intercept)
- On valide le modèle en vérifiant que les résidus sont des bruits blancs.

Pour tester la « blancheur » des résidus on utilise le test du porte-manteau avec la statistique de test $Q(h)$. On prend l'autocorrélation des résidus

$$\hat{\rho}_\varepsilon(h) = \frac{\sum_{t=1}^{n-h} \hat{\varepsilon}_t \hat{\varepsilon}_{t+h}}{\sum_{t=1}^n \hat{\varepsilon}_t^2}$$

que l'on somme pour constituer Q qui, sous l'hypothèse qu'il n'a pas de corrélations entre les résidus, converge en loi vers la loi du Khi-2 à h degrés de libertés.

$$Q(h) = n \sum_{i=1}^h \hat{\rho}_\varepsilon(j)^2 \frac{\mathcal{L}}{n \rightarrow +\infty} \chi^2(h) \quad (12)$$

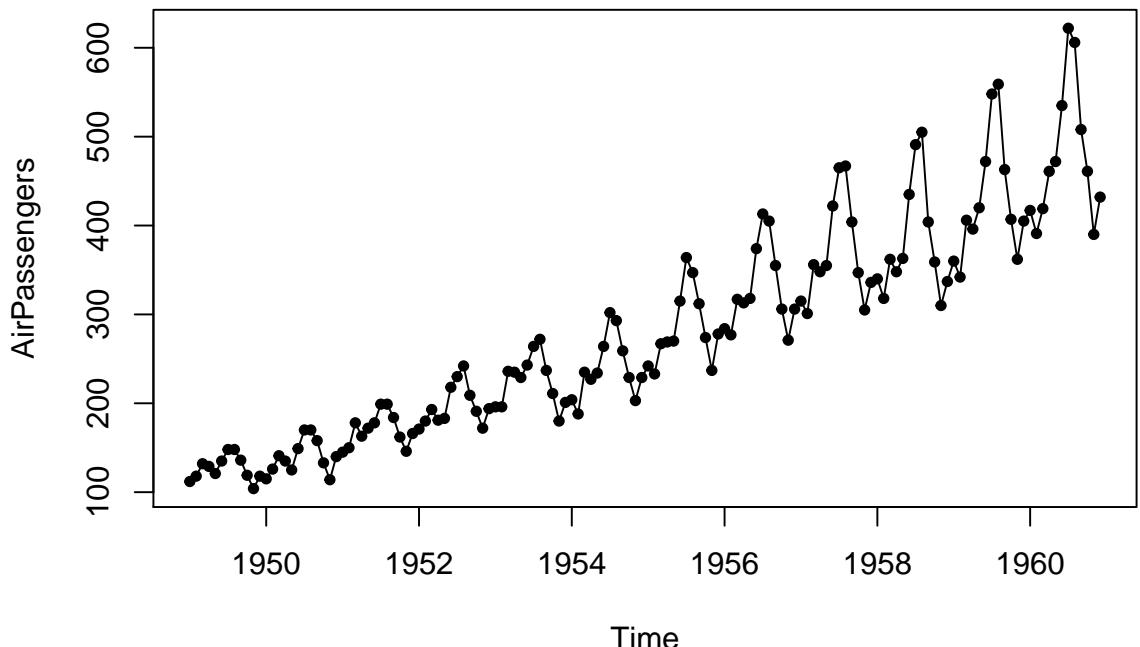
Le test pour un risque α est rejeté si $Q(h) > \chi^2_{1-\alpha}(h)$.

Le choix entre différents modèles se fait par la minimisation d'un critère d'information comme l'*AIC* ou le *BIC*.

1.5 Simulation d'une chronique avec le modèle SARIMA

De manière semblable à l'article nous construisons un modèle SARIMA qui correspond à la chronique *AirPassenger* en utilisant la méthode de Box et Jenkins

chronique AirPassenger du package TSA



L'affichage brut de la chronique indique un modèle multiplicatif avec une tendance, ce qui nous pousse à prendre le logarithme de la série pour avoir un modèle additif, et de la dériver pour ôter la tendance linéaire pour la modélisation.

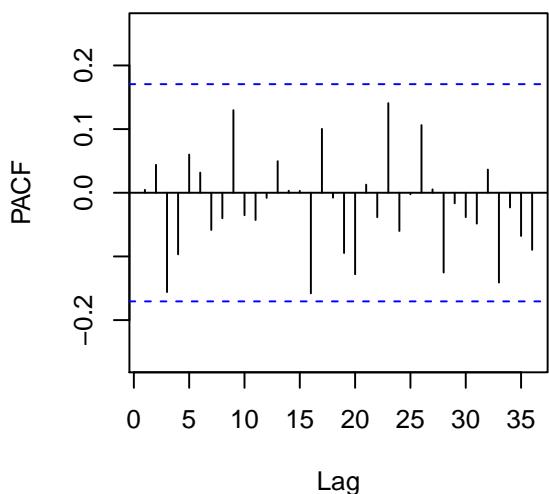
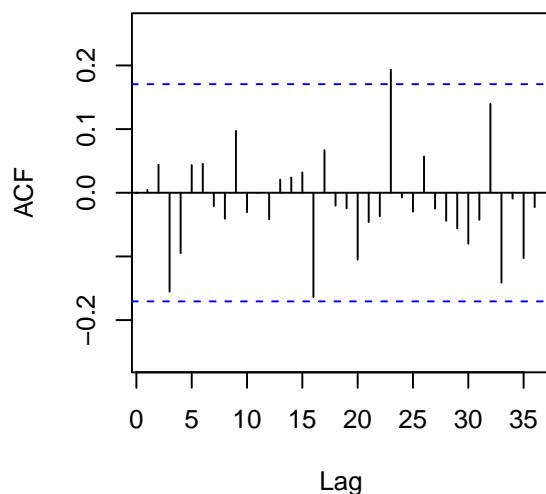
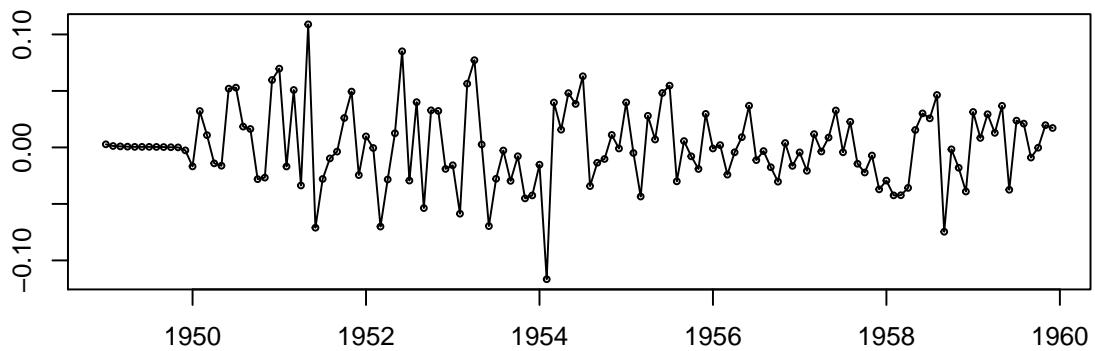
On modélise le logarithme de la chronique avec un processus SARIMA[0, 1, 1][0, 1, 1]12

```
AP <- window(AirPassengers, start=c(1949, 1), end=c(1959, 12))
#tsdisplay(AirPassengers)
analyse = Arima(AP, lambda = 0, order = c(0,1,1), seasonal = c(0,1,1))
```

Pour valider le modèle nous vérifions que les résidus sont des bruits blanc gaussiens. On regarde tout d'abord si on trouve des structures significatives dans les résidus, ce qui n'est pas le cas ici.

```
tsdisplay(residuals(analyse))
```

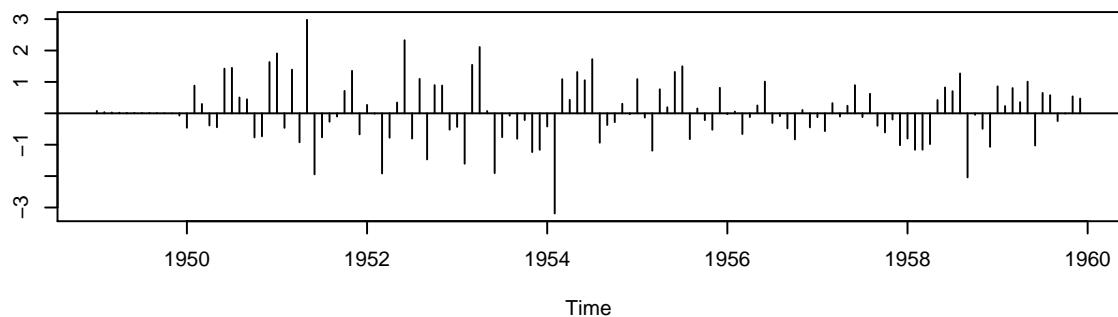
residuals(analyse)



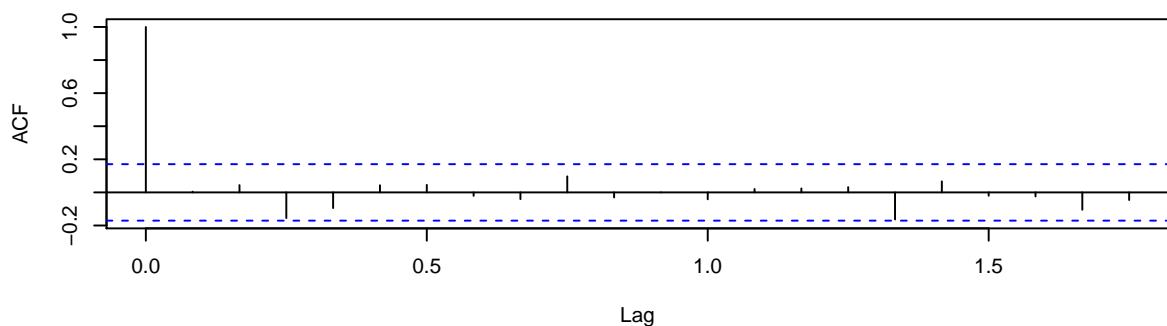
On effectue ensuite un test du porte-manteau (Ljung-Box). On vérifie que les p-values sont bien supérieures au seuil 5%.

```
tsdiag(analyse)
```

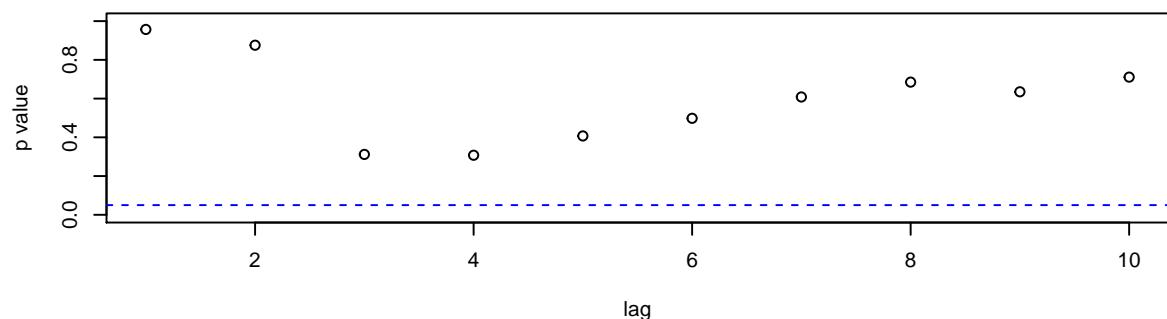
Standardized Residuals



ACF of Residuals



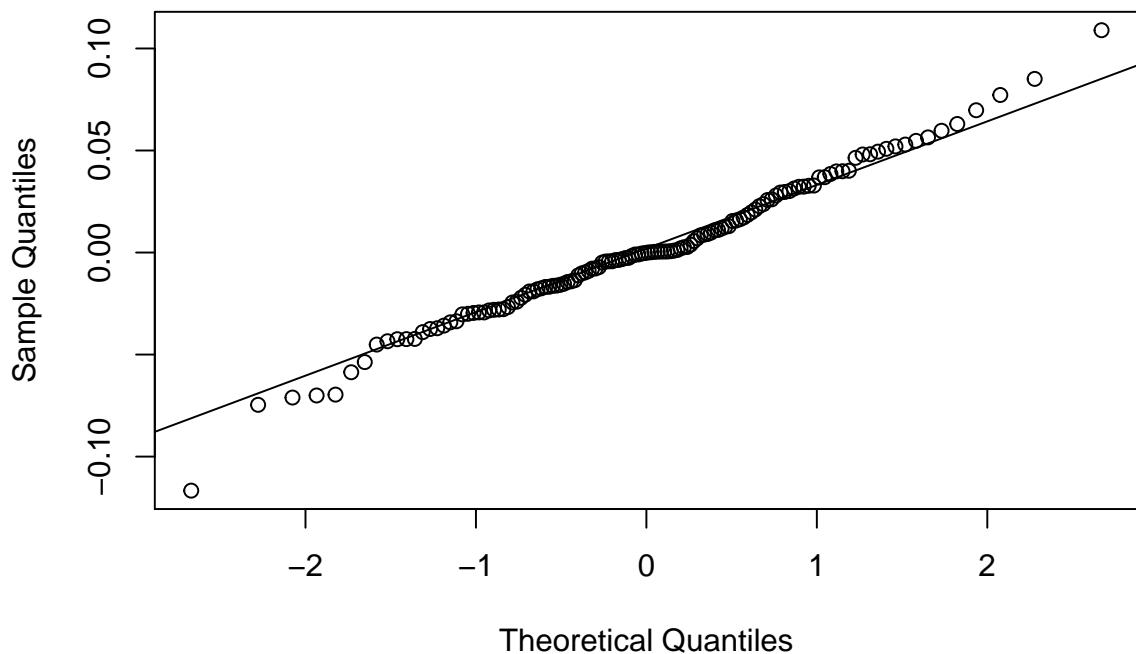
p values for Ljung–Box statistic



On peut également regarder la droite de Henry pour vérifier le caractère gaussien des résidus. Si les résidus sont parfaitement gaussiens ils collent à la droite de Henry.

```
qqnorm(residuals(analyse))
qqline(residuals(analyse))
```

Normal Q–Q Plot



On peut ensuite vérifier la significativité des coefficients du modèle par un test de Wald sous l'hypothèse H_0 que les coefficients sont nuls.

```
coeftest(analyse)
```

```
##  
## z test of coefficients:  
##  
##      Estimate Std. Error z value Pr(>|z|)  
## ma1 -0.348448  0.094280 -3.6959 0.0002191 ***  
## sma1 -0.562260  0.077416 -7.2629 3.79e-13 ***  
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

On peut finalement afficher l'estimation des coefficients du modèle avec les critère AIC et BIC ainsi que les pondérateurs pour mesurer l'erreur.

```
summary(analyse)
```

```
## Series: AP  
## ARIMA(0,1,1)(0,1,1)[12]  
## Box Cox transformation: lambda= 0  
##  
## Coefficients:  
##          ma1     sma1  
##        -0.3484  -0.5623  
## s.e.   0.0943  0.0774  
##  
## sigma^2 estimated as 0.001338: log likelihood=223.63  
## AIC=-441.26  AICc=-441.05  BIC=-432.92  
##
```

```

## Training set error measures:
##               ME      RMSE      MAE      MPE      MAPE      MASE
## Training set 0.2583509 9.000802 6.753838 0.04571618 2.58749 0.2218009
##               ACF1
## Training set 0.02125945

```

On peut finalement afficher la prédition pour l'année suivante avec l'intervalle de confiance à 95%:

```

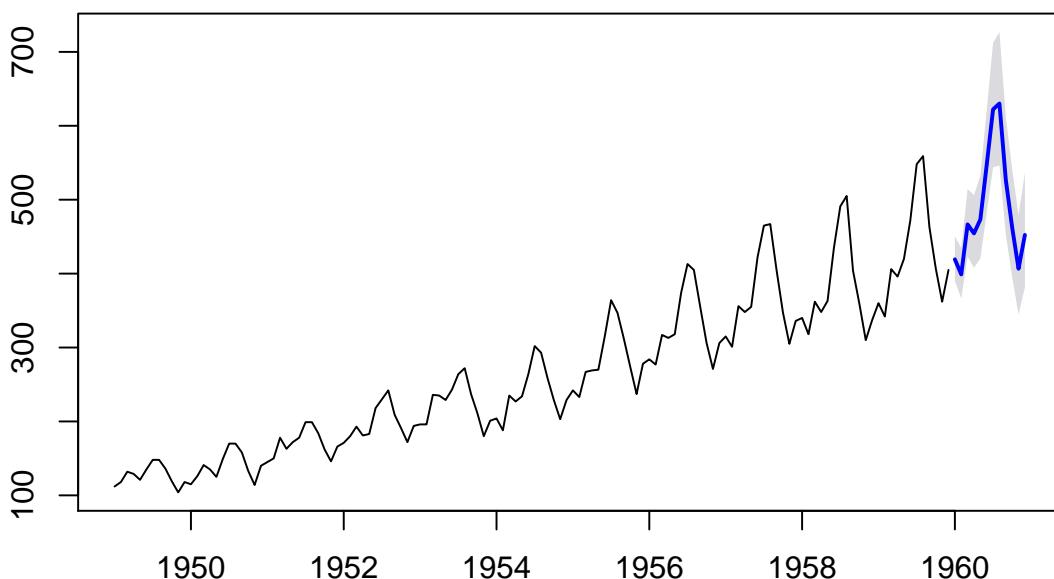
pred = forecast(analyse, h=12, level=0.95, lambda = 0)

## Warning in InvBoxCox(pred$pred, lambda, biasadj, var(residuals.Arima(object)), :
## biasadj information not found, defaulting to FALSE.

plot(pred)

```

Forecasts from ARIMA(0,1,1)(0,1,1)[12]



Pour mesurer l'erreur de prédition on utilise le pourcentage d'erreur relative MAPE

$$\text{MAPE} = \frac{100}{T} \sum_{t=1}^T \left| \frac{X_t - \hat{X}_{t-1}}{X_t} \right| \text{ avec } \hat{X}_t \text{ la prédition de } X_t \text{ à partir de } X_{t-1}$$

```

Y_true <- window(AirPassengers, start=c(1960, 1))
MAPE <- mean(abs(Y_true - pred$mean)/Y_true)
MAPE

## [1] 0.02904473

```

2 Approche Bayésienne

Nous allons dans cette partie adopter une approche bayésienne. Pour des observations X l'approche bayésienne considère pour déterminer les paramètres θ d'un modèle la distribution de probabilité des paramètres postérieure $p(\theta|X)$ en se basant sur la formule de Bayes

$$\begin{aligned} p(\theta|X) &= \frac{p(X|\theta)p(\theta)}{p(X)} \\ &= \frac{p(X|\theta)p(\theta)}{\int_{\Theta} p(X|\theta)p(\theta)d\theta} \end{aligned}$$

Avec $p(\theta)$ la probabilité à priori des paramètres. Dans l'approche fréquentiste précédente nous maximisions la vraisemblance $p(X|\theta)$.

Pour déterminer le θ optimal $\hat{\theta}$ dans l'approche bayésienne nous utilisons une fonction de coût $L(\cdot, \cdot)$ et le risque bayésien $R_B(X, \theta')$

$$\begin{aligned} R_B(y, \theta') &= \int_{\Theta} L(\theta, \theta') \pi(\theta|X) d\theta \\ \hat{\theta} &= \arg \min_{\theta' \in \Theta} R_B(y, \theta') \end{aligned} \tag{13}$$

On prend communément $L(\theta, \theta') = \|\theta - \theta'\|^2$ ce qui donne pour $\hat{\theta}$

$$\hat{\theta} = E[\theta|y]$$

Nous utilisons cette distribution à postériori pour établir une distribution pour la prédiction d'observations futures X^{new} .

$$p(X^{new}|X) = \int_{\Theta} p(X^{new}|\theta) \cdot p(\theta|X) d\theta \tag{14}$$

Dans la réalité la probabilité postérieure ne pouvant être exprimée simplement, nous allons voir une méthode permettant de simuler la probabilité postérieure.

2.1 Méthode de Monté-Carlo et Échantillonnage de Gibbs

La méthode MCMC *Markov Chain Monte Carlo* utilise les propriétés ergodiques des chaînes de Markov et la méthode de Monté-Carlo pour estimer une densité de probabilité jointe de variables aléatoires $\pi(U_1, \dots, U_K)$ que l'on ne peut obtenir analytiquement.

On se place dans le cas où nous pouvons définir les lois conditionnelles entre les variables U_i dont on souhaite connaître la loi jointe avec les lois conditionnelles entre ces variables $Loi(U_j|U_1, \dots, U_{j-1}, U_j, \dots, U_K)$.

En tirant successivement les variables selon leur loi conditionnelle on construit une ainsi une chaîne de Markov dont la loi stationnaire est la distribution jointe des variables.

Nous pouvons ainsi après des tirages successifs obtenir des tirages (U_1, \dots, U_K) selon la loi jointe ce qui nous permet de faire des estimations selon la méthode de Monté-Carlo.

Dans la suite nous rappelons les principes de la méthode de Monté-Carlo et des chaînes de Markov, pour présenter l'échantillonateur de Gibbs utilisé pour réaliser les tirages.

Méthode de Monté Carlo

La méthode de Monté-Carlo permet d'obtenir asymptotiquement l'espérance d'une variable aléatoire X en effectuant n tirages selon la loi de X .

$$\mathbb{E}[f(x)] = \int f(x)p(x)dx \approx \frac{1}{n} \sum_{i=1}^n f(x^i)$$

où les (x^i) sont des tirages indépendants selon la loi de X

Habituellement utilisée pour estimer des intégrales, nous utilisons la méthode de Monté-Carlo pour estimer des lois de variables aléatoires dont nous ne disposons pas d'expression analytique.

Chaîne de Markov

Soit $(X_n)_{n \in \mathbb{N}}$ une suite de variable aléatoire définies sur $(\Gamma, \mathcal{A}, \mathbb{P})$ à valeur dans (E, \mathcal{E}) .

La suite $(X_n)_{n \in \mathbb{N}}$ est une chaîne de Markov sur $(\Gamma, \mathcal{A}, \mathbb{P})$ si elle vérifie pour $\forall n \in \mathbb{N}$, $\forall (x_0, \dots, x_n) \in E^n | \mathbb{P}(X_n = x_n, \dots, X_0 = x_0) > 0$,

$$\forall n \in \mathbb{N} \quad \mathbb{P}(X_{n+1} = x_{n+1} | X_0 = x_1, \dots, X_n = x_n) = \mathbb{P}(X_{n+1} = x_{n+1} | X_n = x_n)$$

En particulier $(X_n)_{n \in \mathbb{N}}$ est une chaîne de Markov homogène si elle vérifie

$$\forall i, j \in I \quad \mathbb{P}(X_{n+1} = x_j | X_n = x_i) = P(x_i, x_j)$$

Avec P la probabilité de transition.

Une mesure positive v sur (E, \mathcal{E}) est invariante pour la chaîne X si

$$\forall y \in E \quad v(y) = \sum_{x \in E} P(x, y)v(x)$$

Si X est une chaîne irréductible récurrente elle admet une unique mesure invariante v avec notamment

$$\text{Pour } x \in E, \forall y \in E, \quad v(y) = \lim_{n \rightarrow +\infty} P^n(x, y)$$

De plus avec les théorèmes ergodiques nous avons le résultat suivant pour une mesure invariante v et pour une fonction f $-v$ mesurable

$$\lim_{n \rightarrow +\infty} \frac{1}{n} \sum_{i=1}^n f(X_i) = \sum_{x \in E} f(x)v(x) \tag{15}$$

Algorithme d'échantillonage de Gibbs

1. Prendre des valeurs initiales $U_k^{(0)}$, pour $k = 1, \dots, K$.
2. Pour $t = 1, 2, \dots$:
 - Pour $k = 1, 2, \dots, K$:
 - tirer $U_k^{(t+1)}$ depuis la distribution $\pi(U_k | U_1^{(t+1)}, \dots, U_{k-1}^{(t+1)}, U_{k+1}^{(t)}, \dots, U_K^{(t)})$
3. Continuer l'étape 2 jusqu'à convergence de la distribution jointe $\pi(U_1^{(t)}, U_2^{(t)}, \dots, U_K^{(t)})$

On note pour $x \in E$ avec $x = (x_1, \dots, x_K)$, $x_{-i} = (x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_K)$
Soit $x, y \in E$. On note $x \underset{j}{\sim} y \iff x_i = y_i \quad \forall i \neq j$

La loi conditionnelle entre les états dans l'échantillonnage de Gibbs est alors (le dénominateur ne dépend plus de x_i)

$$\pi(x_i|x_{-i}) = \frac{\pi(x)}{\pi(x_{-i})}$$

La probabilité de transition entre deux état x et y est

$$P(x, y) = \begin{cases} \pi(y_i|x_{-i}) & \text{si } \exists i \mid x \underset{i}{\sim} y \\ 0 & \text{sinon} \end{cases}$$

La chaîne de Markov ainsi définie est irréductible comme tous ces états sont communicants au bout d'un nombre finis d'étapes. Elle est de plus apériodique comme $P(x, x) > 0$. Cela nous assure l'unicité de la mesure stationnaire de masse finie [1].

On vérifie ainsi que π est une probabilité d'une chaîne réversible.

$$\begin{aligned} \text{Si } x \underset{i}{\sim} y \quad \pi(x)P(x, y) &= \pi(x_i, x_{-i}) \frac{\pi(y_i, x_{-i})}{\pi(x_{-i})} \\ &= \pi(y_i, x_{-i}) \frac{\pi(x_i, x_{-i})}{\pi(x_{-i})} \\ &= \pi(y)P(y, x) \end{aligned}$$

Ainsi la distribution jointe π est la probabilité stationnaire de la chaîne.

On peut ainsi estimer la probabilité marginale \hat{p} d'une variable en utilisant l'échantillonnage $(U_1^{(t)}, U_2^{(t)}, \dots, U_K^{(t)})$

$$\hat{p}_{U_i}(y) = \frac{1}{(M-m+1)} \sum_{t=m}^M \pi(U_i|U_1, \dots, U_{i-1}, U_{i+1}, \dots, U_K)$$

On prend en effet les échantillons qu'après m itérations pour que la distribution converge vers la probabilité stationnaire de la chaîne de Markov (*burn-in*). m dépend de la vitesse de convergence.

De plus les tirages successifs de l'échantillonneur de Gibbs sont corrélés. Dans la pratique nous ne prenons également que des tirages $(U_1^{(t)}, U_2^{(t)}, \dots, U_K^{(t)})$ espacés d'un *lag* pour obtenir des échantillons indépendants. On peut regarder le corrélogramme et le corrélograme partiel de l'échantillon pour déterminer le *lag*.

Plus les échantillons sont corrélés entre eux, plus nous devons faire de simulations.

On peut s'intéresser à la variance d'un tel estimateur. Dans [14] on montre que la variance de l'estimateur dépend des autocorrélations entre les échantillons de la série générée par l'échantillonneur. Ainsi pour un estimateur

$$\hat{\mu} = \frac{1}{N} \sum_{n=1}^N f(X_n) \tag{16}$$

$$\text{Var}(\hat{\mu}) \approx \frac{\sigma(f)^2 \tau(f)}{N} \tag{17}$$

Avec $\sigma(f)^2$ la variance de $f(X)$ pour la distribution stationnaire et

$$\tau = 1 + 2 \sum_{k=1}^{\infty} \rho(k)$$

le « temps d'autocorrélation ». On a donc intérêt à diminuer les corrélations entre les tirages pour réduire la variance de l'estimateur. On peut fournir un nombre optimal du nombre de tirages effectués en fonction de τ . Pour N échantillons on peut mesurer l'influence des autocorrections avec la taille effective d'échantillonnage $ESS = \frac{N}{\tau}$. Cela revient à estimer l'erreur de Monté-Carlo de la méthode par $\frac{\sigma(f)^2}{ESS}$ au lieu de $\frac{\sigma(f)^2}{N}$.

Par ailleurs la méthode MCMC présente comme risque de ne pas explorer toutes les valeurs de la loi que l'on souhaite estimer. En effet par l'aspect aléatoire certaines valeurs de la loi peuvent ne pas être explorées. Pour contrer cela on peut augmenter le nombre de simulations et lancer plusieurs simulations avec des initialisations différentes.

Nous allons utiliser la méthode MCMC dans la suite pour tirer des échantillons des lois postérieures $p(\theta|X)$ pour les paramètres que l'on souhaite estimer.

2.2 Le modèle d'espace d'état

Une généralisation des modèles employés pour la modélisation des séries temporelles s'écrit :

$$y_t = Z_t^T \alpha_t + \epsilon_t \quad \epsilon_t \sim \mathcal{N}(0, H_t) \quad (18)$$

$$\alpha_{t+1} = T_t \alpha_t + R_t \eta_t \quad \eta_t \sim \mathcal{N}(0, Q_t) \quad (19)$$

avec Z_t, H_t, T_t, R_t, Q_t des matrices à déterminer.

ϵ_t indépendants, η_t indépendants

(y_t, ϵ_t) indépendants, (α_t, η_t) indépendants

18 est l'équation dite d'*observation* car elle correspond aux états de la série temporelle observables. 19 est l'équation dite de *transition* car elle correspond à des états non visibles de façon similaire à un modèle de Markov caché. La combinaison de ces deux équations forme un modèle espace d'état (*state space form model*) sous sa forme gaussienne linéaire. On peut notamment dériver un modèle *ARIMA* de ce modèle général.

Le modèle bayésien classique utilisé dans [10] s'exprime de la manière suivante :

$$\begin{aligned} y_t &= \mu_t + \tau_t + \beta^T \mathbf{x}_t + \epsilon_t & \epsilon_t &\sim \mathcal{N}(0, \sigma_\epsilon^2) \\ \mu_t &= \mu_{t-1} + \delta_{t-1} + u_t & u_t &\sim \mathcal{N}(0, \sigma_u^2) \\ \delta_t &= \delta_{t-1} + v_t & v_t &\sim \mathcal{N}(0, \sigma_v^2) \\ \tau_t &= - \sum_{s=1}^{S-1} \tau_{t-s} + w_t & w_t &\sim \mathcal{N}(0, \sigma_w^2) \end{aligned} \quad (20)$$

On peut exprimer 20 avec 18 avec un terme de régression comme dans [7] (S.J. Koopman 1993)

On omet la composante saisonnière τ_t , traitée à part, qui ne sera pas pris en compte dans le modèle final car les données sont « désaisonnalisées ».

$$\begin{aligned} y_t &= Z_t \alpha_t + \varepsilon_t \\ \alpha_{t+1} &= T_t \alpha_t + \varepsilon_t \end{aligned} \tag{21}$$

Avec

$$\begin{aligned} Z_t^T &= (1 \ 0 \ \beta^T \mathbf{x}_t), \quad H_t = \sigma_\epsilon^2 \in \mathbb{R}^{+*} \\ \alpha_t &= \begin{pmatrix} \mu_t \\ \delta_t \\ 1 \end{pmatrix}, \quad T_t = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ Q_t &= \begin{pmatrix} \sigma_u^2 & 0 & 0 \\ 0 & \sigma_v^2 & 0 \\ 0 & 0 & \sigma_w^2 \end{pmatrix}, \quad R_t = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \eta_t = \begin{pmatrix} u_t \\ v_t \\ w_t \end{pmatrix} \end{aligned}$$

Si on inclut le terme saisonnal τ on a $\alpha_t^T = (\mu_t \ \delta_t \ \tau_t \ \tau_{t-1} \ \dots \ \tau_{t-S+1})$

Nous allons décrire les méthodes pour obtenir les paramètres du modèle ainsi définis. C'est à dire les méthodes dans une approche bayésienne nous permettant avec une distribution à priori des paramètres, obtenir la distribution à postériori. Tous d'abord nous allons présenter le filtre de Kalman afin d'obtenir les états cachés α_T puis l'implémentation de l'échantillonneur de gibbs afin d'obtenir la distribution postérieure pour β .

2.3 Filtre de Kapman

On veut déterminer dans cette partie les états « cachés » α_t dit aussi latents. Les α_t peuvent être vus comme les états cachés d'une chaîne de Markov caché dont les états observables sont les y_i . On détermine les α_t par l'utilisation du filtrage et du lissage de Kalman [5].

On veut déterminer α_t , H_t et Q_t (H_t, Q_t constant en pratique) à partir des observations $y_{1:t} = y_1, y_2, \dots, y_t$. On suppose connu β .

Le principe de filtrage est obtenu par la linéarité des variables gaussiennes dans 18 et 19 $p(\alpha_t|y_{1:t}) \sim \mathcal{N}(\hat{\alpha}_t, P_t)$. Avec ainsi $\hat{\alpha}_t = E(\alpha_t|y_{1:T})$ et $P_t = E[(\alpha_t - \hat{\alpha}_t)(\alpha_t - \hat{\alpha}_t)^T]$

Pour effectuer un tirage selon la loi $p(\alpha_t|y_{1:t})$ on effectue un tirage α_t^+ selon la probabilité à priori $p(\alpha)$ et on prend $\hat{\alpha}_t^+ = E(\alpha_t^+|y_{1:t})$. Ainsi $\tilde{\alpha}_t = \hat{\alpha}_t - \hat{\alpha}_t^+ + \alpha_t^+$ est un tirage selon $p(\alpha_t|y_{1:t})$

Le filtre de base pour estimer le bruit gaussien des $\begin{pmatrix} \varepsilon_t \\ \eta_t \end{pmatrix}$ en itérant sur $t = 1, \dots, T$ est donné par [5] :

$$\begin{aligned} a_{t+1} &= E(\alpha_{t+1}|y_{1:t}) = T_t a_t + K_t v_t \\ v_t &= y_t - E(y_t|y_{1:t-1}) = y_t - Z_t a_t \\ P_{t+1} &= T_t P_t L_t^T + R_t Q_t R_t^T \\ F_t &= \text{Var}(v_t) = Z_t P_t Z_t^T + H_t \\ K_t &= T_t P_t Z_t^T F_t^{-1} \\ L_t &= T_t - K_t Z_t \\ \begin{pmatrix} \hat{\varepsilon}_t \\ \hat{\eta}_t \end{pmatrix} &= \begin{bmatrix} H_t F_t^{-1} & -H_t K_t^T \\ 0 & Q_t R_t^T \end{bmatrix} \begin{pmatrix} v_t \\ r_t \end{pmatrix} \end{aligned} \tag{22}$$

avec r_t évalué rétrospectivement sur $t = T, \dots, 1$:

$$\begin{aligned} r_{t-1} &= Z_t F_t^{-1} v_t + L_t^T r_t \text{ pour } t = T, \dots, 1 \\ r_T &= 0 \end{aligned} \tag{23}$$

Ainsi on peut itérer les $\hat{\alpha}_{t+1}$ sur $t = 1, \dots, T$

$$\begin{aligned} \hat{\alpha}_{t+1} &= E(\alpha_t | y_{1:T}) \\ &= T_t \hat{\alpha}_t + R_t \hat{\eta}_t \\ &= T_t \hat{\alpha}_t + R_t Q_t R_t^T r_t \\ \text{avec } \hat{\alpha}_1 &= a_1 + P_1 r_0 \end{aligned} \tag{24}$$

L'algorithme pour obtenir le tirage $\tilde{\alpha}$ est le suivant :

- On tire le vecteur des perturbations $\begin{pmatrix} \varepsilon_t^+ \\ \eta_{t+1}^+ \end{pmatrix}$ selon la distribution à priori avec H_t et Q_t . On tire $\alpha_1^+ \sim N(a_1, P_1)$ et ainsi itérativement sur 18 en prenant la moyenne on obtient α^+ et y^+ .
- Avec 24 et 23 on obtient $\hat{\alpha}$ et en prenant la moyenne $\hat{\alpha}^+$.
- On obtient ainsi $\tilde{\alpha} = \hat{\alpha} - \hat{\alpha}^+ + \alpha^+$

On peut séparer la procédure précédente en deux éléments : le filtre de Kalman et le lissage de Kalman.

Filtre de Kalman

- On itère en avant de 1 à T .
- On combine $p(\alpha_t | y_{1:t})$ et y_t pour avoir $p(\alpha_{t+1} | y_{1:t})$

Lissage de Kalman

- On itère en arrière de T à 1
- On combine r_t et $p(\alpha_{t+1} | y_{1:t})$ pour obtenir $p(\alpha_{t+1} | y_{1:T})$

2.4 Utilisation d'une loi à priori *spike and slab*

Nous mettons en place une méthode de sélection de variable similaire à la regression ridge pour « mettre à zéro » certains paramètres de la régression. Ceci permet notamment de prendre en compte les corrélations entre les variables explicatives du modèle et de réduire la taille du modèle.

Cette sélection se fait sur la probabilité à priori de β en utilisant une loi variable aléatoire γ qui suit une loi de Bernoulli.

Les distributions à priori

On définit ainsi la *spike and slab prior* :

$$p(\beta, \gamma, \sigma_\varepsilon^2) = p(\beta_\gamma | \gamma, \sigma_\varepsilon^2) p(\sigma_\varepsilon^2 | \gamma) p(\gamma) \tag{25}$$

γ ayant pour distribution à priori avec K le nombre de variables explicatives.

$$p(\gamma) = \prod_{k=1}^K \pi_k^{\gamma_k} (1 - \pi_k)^{1-\gamma_k} \tag{26}$$

Les π_k peuvent être pris constants $\forall k$, à 0 ou 1 si on veut exclure ou inclure certaines variables. On peut aussi si on souhaite avoir un modèle de taille p poser $\pi = \frac{p}{K}$

On a également les probabilités à priori

$$\beta_\gamma \mid \sigma_\epsilon^2, \gamma \sim \mathcal{N} \left(b_\gamma, \sigma_\epsilon^2 (\Omega_\gamma^{-1})^{-1} \right) \quad \sigma_\epsilon^2 \mid \gamma \sim IG \left(\frac{\nu}{2}, \frac{ss}{2} \right)$$

Avec $IG(\cdot, \cdot)$ la distribution Gamma inverse, $s\Omega^{-1} \propto X^T X$ similairement à la méthode des moindres carrés. Ω_γ est la matrice Ω pour les rangs et colonnes k tel que $\gamma_k = 1$

$$\frac{ss}{df} = (1 - R^2) s_y^2 \quad (27)$$

Avec R^2 le coefficient de détermination attendu de la régression, et s_y l'écart quadratique moyen de la régression.

Les distributions à postériori

On prend y_t de 18 en omettant la partie temporelle avec y_t^* pour ne garder que la partie de régression

$$y_t^* = y_t - Z_t^{*T} \alpha_t = y_t - Z_t^T |_{\beta \mathbf{x}_t=0} \alpha_t = y_t - \mu_t$$

On prend

$$\mathbf{y}^* = y_{1:n}^*$$

On a alors les résultats suivants pour les lois à postériori

$$\beta_\gamma \mid \sigma_\epsilon, \gamma, \mathbf{y}^* \sim \mathcal{N} \left(\tilde{\beta}_\gamma, \sigma_\epsilon^2 (V_\gamma^{-1})^{-1} \right) \quad \epsilon^2 \mid \gamma, \mathbf{y}^* \sim IG \left(\frac{\nu + n}{2}, \frac{SS_\gamma}{2} \right) \quad (28)$$

Avec

$$\begin{aligned} V_\gamma^{-1} &= (\mathbf{X}^T \mathbf{X})_\gamma + \Omega_\gamma^{-1} \\ \tilde{\beta}_\gamma &= (V_\gamma^{-1})^{-1} (\mathbf{X}_\gamma^T \mathbf{y}^* + \Omega_\gamma^{-1} b_\gamma) \\ SS_\gamma &= ss + \mathbf{y}^{*T} \mathbf{y}^* + b_\gamma^T \Omega_\gamma^{-1} b_\gamma - \tilde{\beta}_\gamma^T V_\gamma^{-1} \tilde{\beta}_\gamma \end{aligned} \quad (29)$$

En utilisant les distributions postérieures et en marginalisant on trouve la distribution postérieure :

$$\gamma \mid \mathbf{y}^* \sim C(\mathbf{y}^*) \frac{|\Omega_\gamma^{-1}|^{\frac{1}{2}}}{|V_\gamma^{-1}|^{\frac{1}{2}}} \frac{p(\gamma)}{SS_\gamma^{\frac{N}{2}-1}} \quad (30)$$

Avec $C(\mathbf{y}^*)$ une constante de normalisation.

2.5 Entrainement des paramètres

Pour obtenir la distribution postérieure des paramètres on utilise la méthode MCMC présenté précédemment. A chaque itération les paramètres tirés des lois à priori permettent de tirer des paramètres selon la loi postérieure.

Sur $1, \dots, M$ On commence par tirer $\gamma, \beta, \sigma_\epsilon^2, \sigma_v^2, \sigma_u^2$ de leur distribution à priori.

1. Avec le filtre de Kalman on simule les états latents α depuis $p(\alpha | y, \gamma, \beta, \sigma_\epsilon^2, \sigma_v^2, \sigma_u^2)$
2. On simule σ_u^2 et σ_v^2 avec la distribution postérieure $p\left(\frac{1}{\sigma_u^2}, \frac{1}{\sigma_v^2} | y, \alpha, \beta, \sigma_\epsilon^2\right)$
3. On simule β et σ_ϵ^2 avec la distribution postérieure $p(\beta, \sigma_\epsilon^2 | y, \alpha, \sigma_u^2, \sigma_v^2)$
4. Retour à la première étape.

$\sigma_u^2, \sigma_v^2, \sigma_w^2$ sont tiré selon la loi $.\mid \gamma \sim IG\left(\frac{\nu}{2}, \frac{ss}{2}\right)$

Le modèle final est la moyenne des modèles $(\gamma, \beta, \sigma_\varepsilon^2, \sigma_v^2, \sigma_u^2)_t$ ainsi tirés.

3 Utilisation du modèle sur la chronique *AirPassenger*

Nous utilisons l'approche bayesienne pour effectuer une prédiction sur la série temporelle *AirPassenger*. Nous utilisons le package `bsts`.

$$y_t = \mu_t + \tau_t$$
$$\mu_{t+1} = \mu_t + \eta_t$$

Nous effectuons 1000 simulations MCMC avec l'année 1960 comme année de validation. On ne considère pas les 100 premières simulations qui correspondent au temps de convergence de la chaîne de Monté-Carlo.

```
library(lubridate)
library(bsts)
library(dplyr)
library(ggplot2)
library(Boom)

data("AirPassengers")
Y <- window(AirPassengers, start=c(1949, 1), end=c(1959,12))
y <- log10(Y)

# ajout d'une marche aléatoire au modèle mu_t = mu_{t-1} + N(0,sigma.level).^2)
ss <- AddLocalLevel(list(), y)
# ajout d'une composante stationnaire annuelle au modèle
ss <- AddSeasonal(ss, y, nseasons = 12)

bsts.model <- bsts(y, state.specification = ss, niter = 1000, ping=0, seed=2016)
```

Nous pouvons regarder les paramètres par défaut des distributions à priori.

```
bsts.model$prior

## $prior.guess
## [1] 0.1807634
##
## $prior.df
## [1] 0.01
##
## $initial.value
## [1] 0.1807634
##
## $fixed
## [1] FALSE
##
## $upper.limit
## [1] 0.2169161
##
## attr(,"class")
## [1] "SdPrior"          "DiffDoubleModel"   "DoubleModel"      "Prior"

help(SdPrior)
# paramètres pour la loi Gamma inverse:
# prior.guess estimation deviation standard SdPrior
# df degree of freedom (df) pour calculer le R^2 à priori
# upper.limit limite haute pour l'écart-type des résidus
# initial.value : valeur initiale de la chronique
# fixed : si sigma est fixé
```

```

#### Get a suggested number of burn-ins
burn <- SuggestBurn(0.1, bsts.model)

#### Predict
p <- predict.bsts(bsts.model, horizon = 12, burn = burn, quantiles = c(.025, .975))

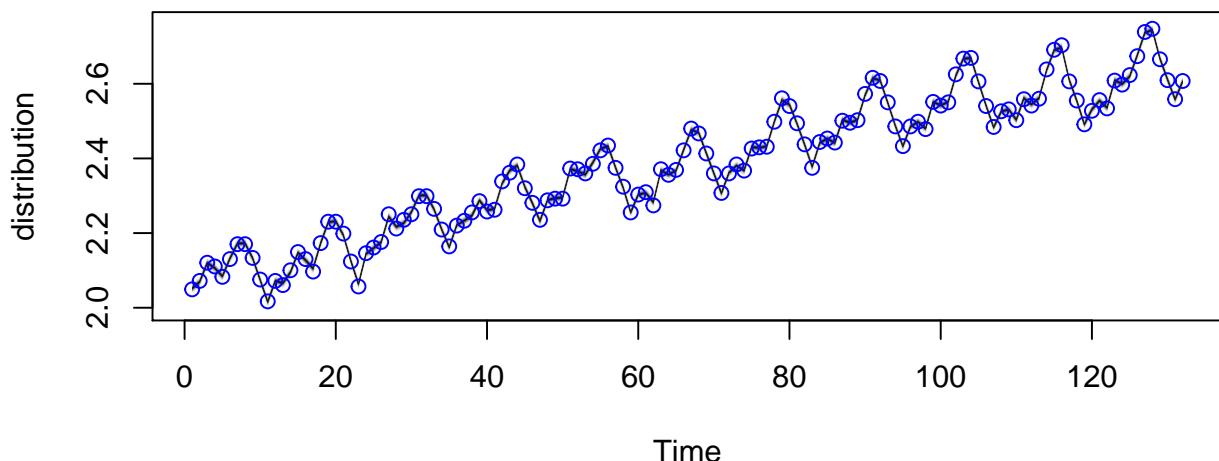
#### Actual versus predicted
d2 <- data.frame(
  # fitted values and predictions
  c(10^as.numeric(-colMeans(bsts.model$one.step.prediction.errors[-(1:burn),]))+y),
  10^as.numeric(p$mean)),
  # actual data and dates
  as.numeric(AirPassengers),
  as.Date(time(AirPassengers)))
names(d2) <- c("Fitted", "Actual", "Date")

#### 95% forecast credible interval
posterior.interval <- cbind.data.frame(
  10^as.numeric(p$interval[1,]),
  10^as.numeric(p$interval[2,]),
  subset(d2, year(Date)>1959)$Date)
names(posterior.interval) <- c("LL", "UL", "Date")
d3 <- left_join(d2, posterior.interval, by="Date")

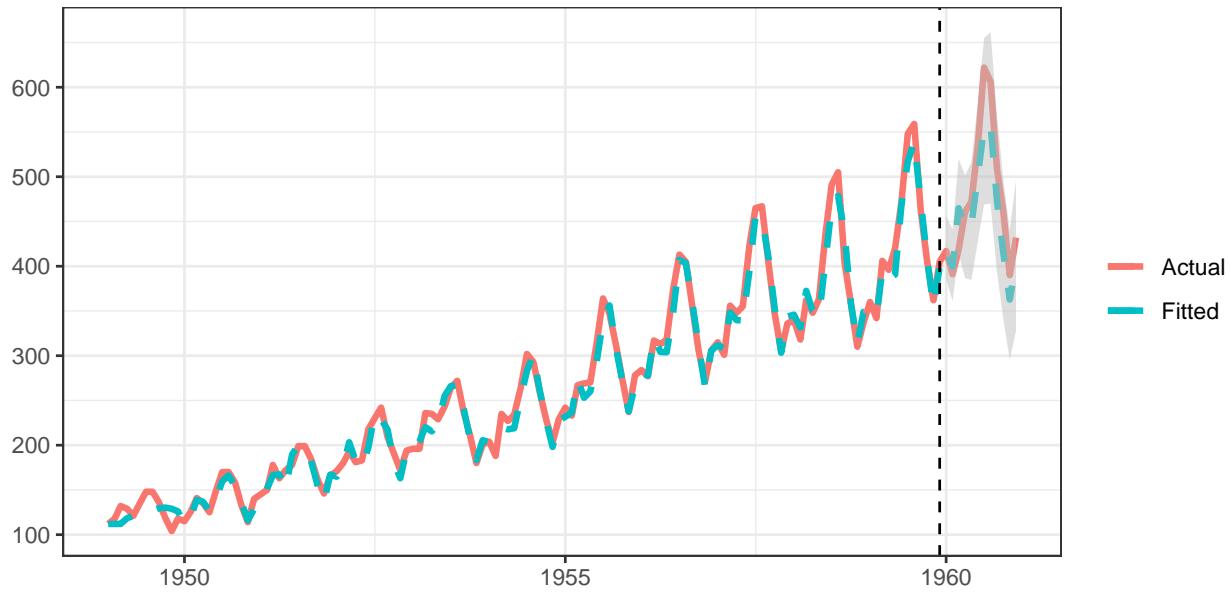
#### MAPE
MAPE <- filter(d2, year(Date)>1959) %>% summarise(MAPE=mean(abs(Actual-Fitted)/Actual))
MAPE
##          MAPE
## 1 0.06530951

```

Les prédictions du modèle pour les valeurs un temps en avant :

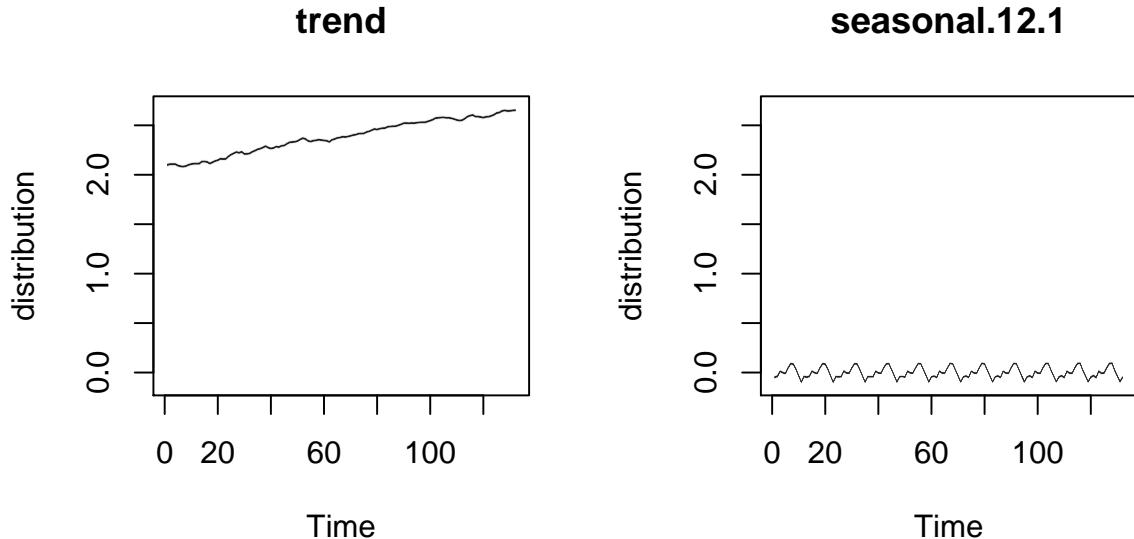


La prédition du modèle pour l'année 1960:



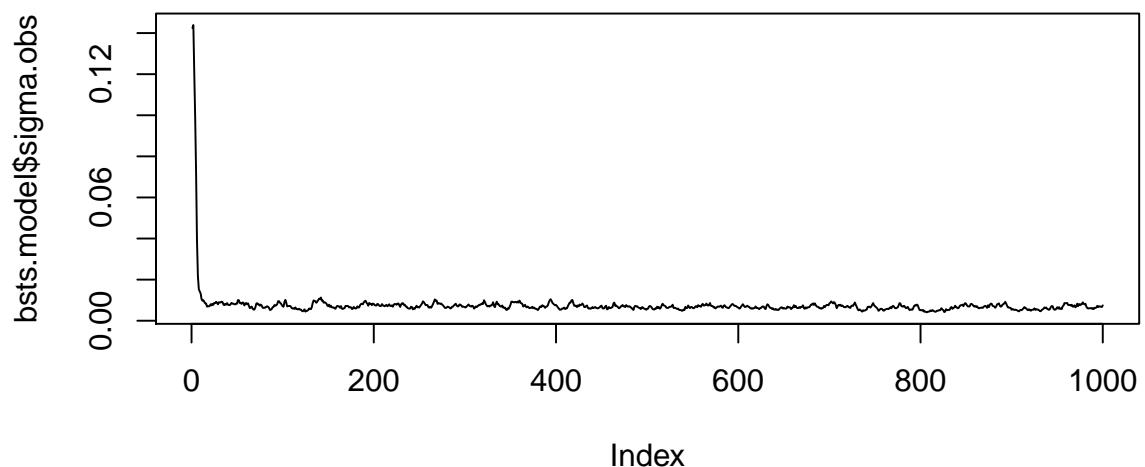
Nous pouvons extraire du modèle la partie saisonnière et la tendance.

```
plot(bsts.model, "components")
```



Nous pouvons regarder l'évolution des valeurs échantillonées pour des paramètres afin d'évaluer la convergence de la simulation.

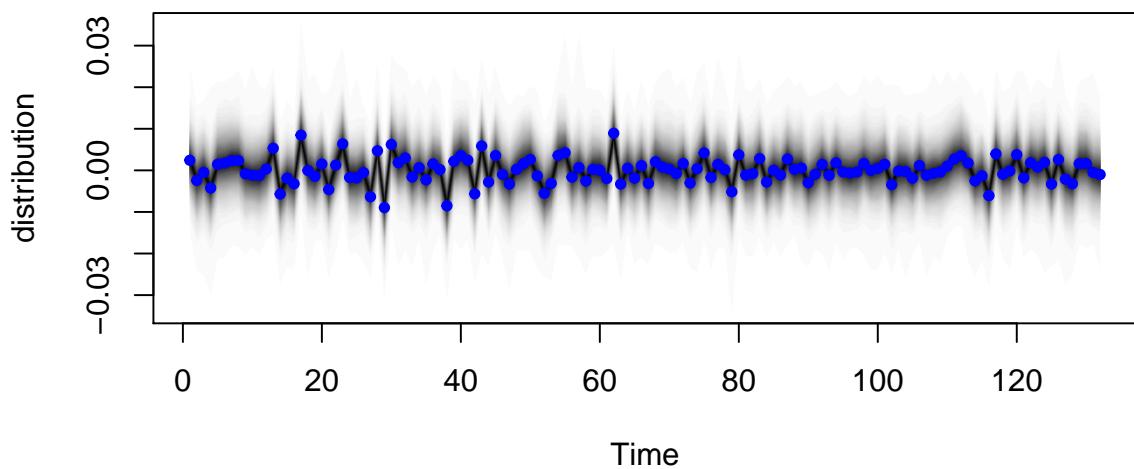
```
plot(bsts.model$sigma.obs, type='l')
```



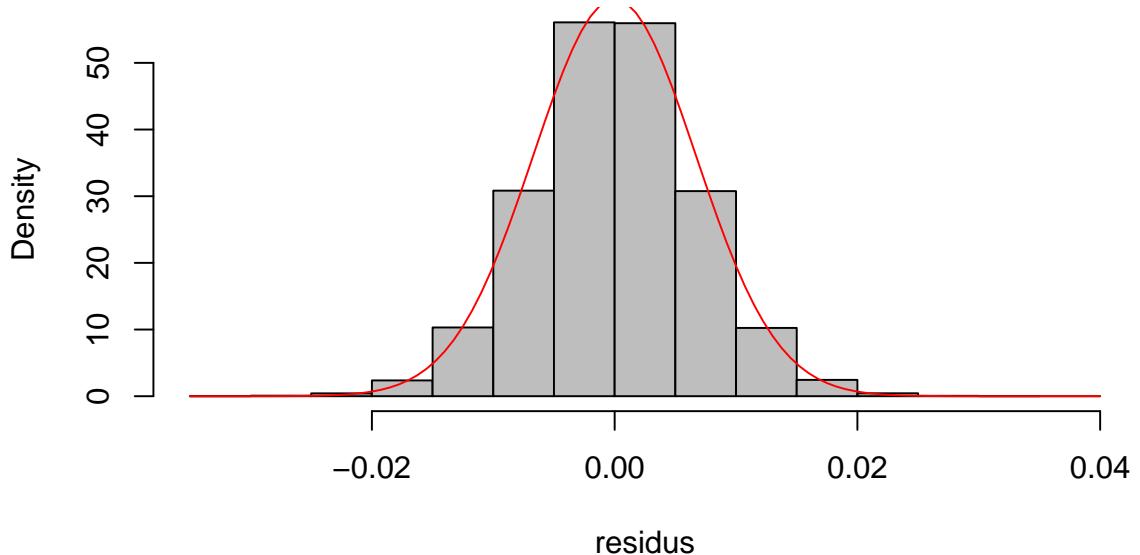
Cela justifie de prendre un burn-in de 10% des tirages, soit 100 tirages.

Nous pouvons également nous intéresser aux résidus du modèle et vérifier qu'ils correspondent bien à un bruit blanc

```
# affichage de la distribution à postériori des résidus du modèle
PlotBstsResiduals(bsts.model, burn = SuggestBurn(0.1, bsts.model), style = "dynamic" )
```

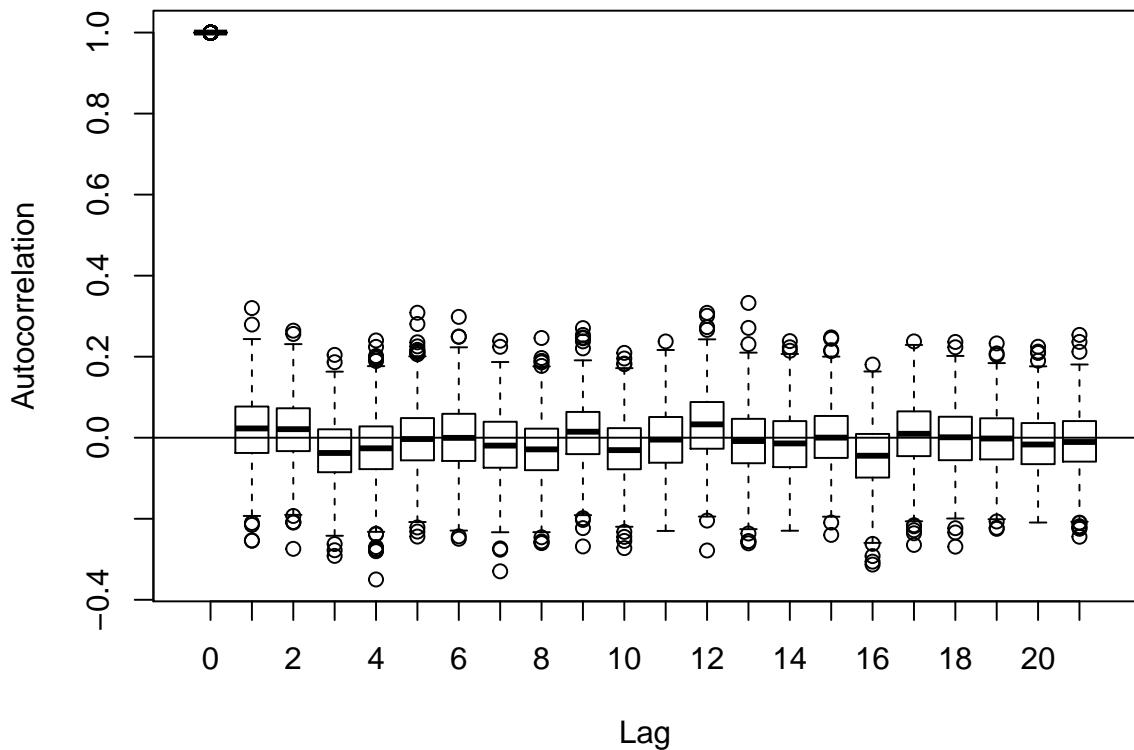


Histogramme de la distribution postérieure des résidus



On affiche le corrélogramme de la distribution postérieure des résidus avec des boîtes à moustache pour chaque lag.

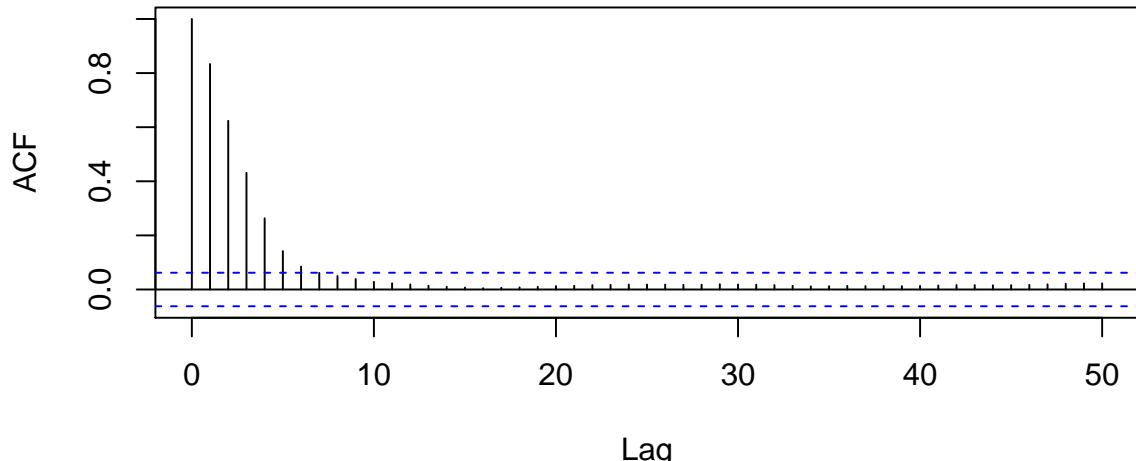
```
# qqline(r)
AcfDist(r)
```



Pour finir nous pouvons observer les autocorrélations entre les étapes successives de l'échantillonneur pour une variable.

```
acf(bsts.model$sigma.obs,lag.max=50)
```

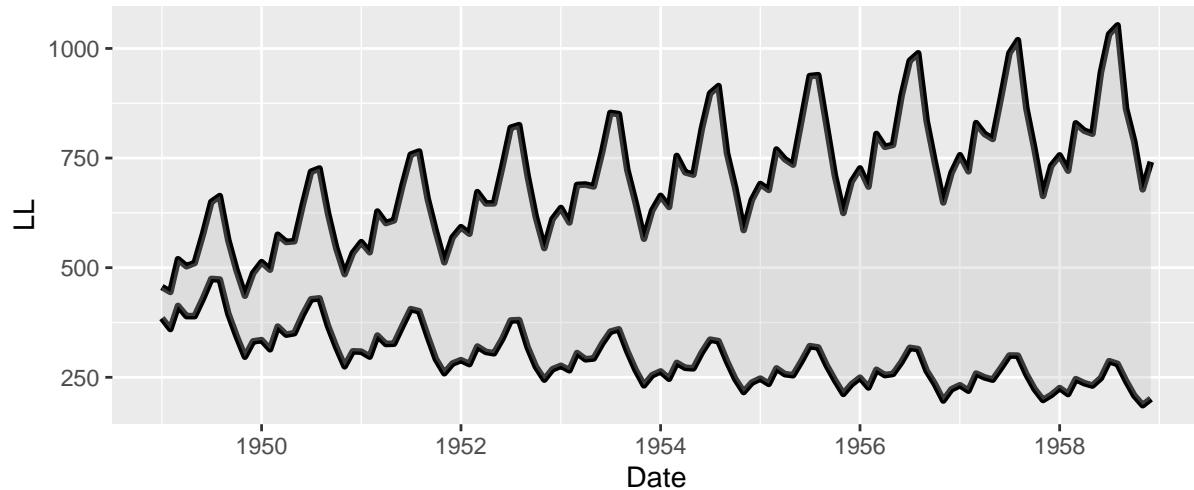
Series bststs.model\$sigma.obs



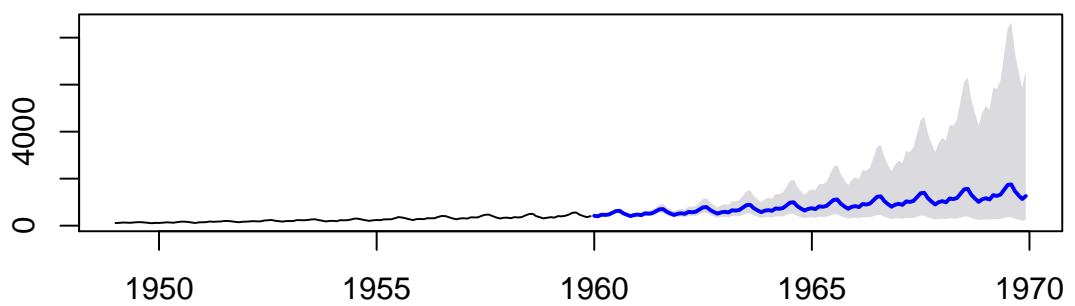
L'erreur relative de prédiction est plus importante au final pour l'approche bayésienne pour cette chronique ce qui nous fait dire que pour cette chronique, l'approche bayésienne est moins efficace.

Néanmoins si on compare les intervalles de confiance des prédictions à 95% entre les deux approches pour une prédiction à long terme, nous pouvons voir que l'intervalle de confiance pour l'approche fréquentiste diverge beaucoup plus que celui correspondant à l'approche bayésienne.

Intervalle de confiance de la prédiction pour l'approche bayésienne



Intervalle de confiance de la prediction pour l'approche fréquentiste



4 Prédition des données du chômage avec la *spike and slab prior*

On cherche à prédire la chronique des demandes d'indémnité du chômage 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 appliquons ainsi le modèle complet tel que décrit précédemment.

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

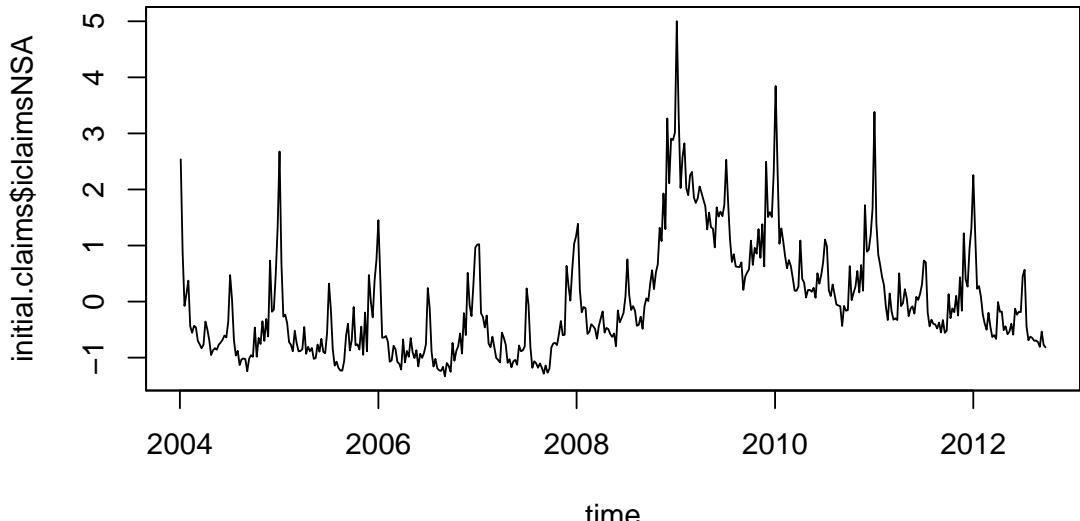
nous tirons 2000 échantillons avec un burn-in de 400 tirages. Pour la *spicke and slab prior* nous prenons une taille de modèle à priori de 5, soit $\pi_t = \pi = 5/10$

```
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")
```

US initial claims for unemployment per week



```
# ajout au modèle de  $\mu_t = \mu_{t-1} + \delta_{t-1} N(0, \sigma^2)$ 
#        $\delta_t = \delta_{t-1} + N(0, \sigma^2)$ 
ss <- AddLocalLinearTrend(list(), initial.claims$iclaimsNSA)
# ajout au modèle d'une composante stationnaire annuelle
ss <- AddSeasonal(ss, initial.claims$iclaimsNSA, nseasons = 52)

# modèle avec une régression avec une distribution à priori spike and slab
# avec les paramètres par défaut ici
# comme les recherches google sont similaires, on attend une taille de
# modèle inférieure à 10, on prend 5.
```

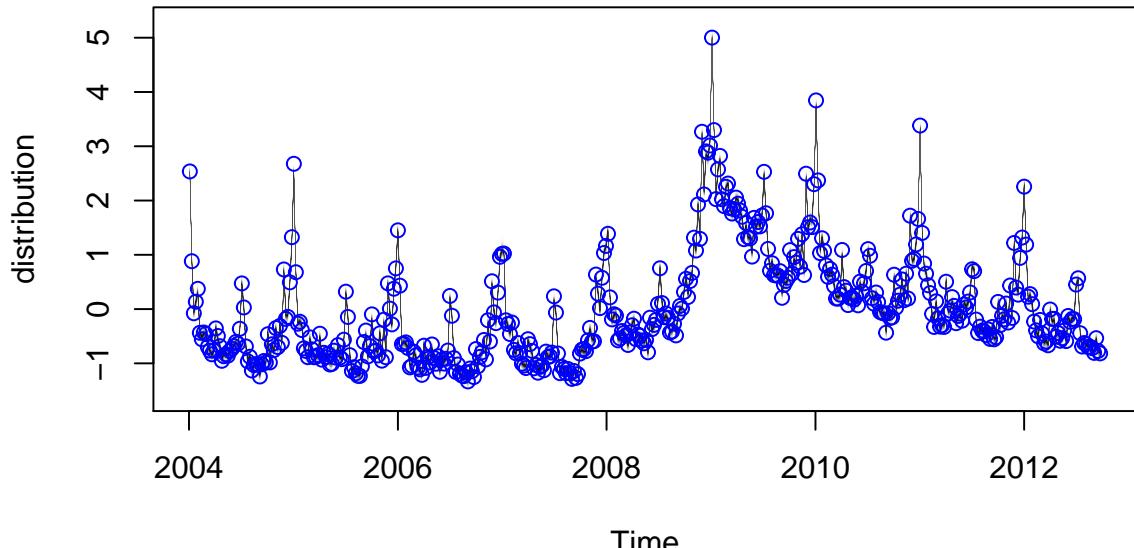
```

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

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

```

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 vérifie ainsi que le modèle correspond bien à la série à prédire.

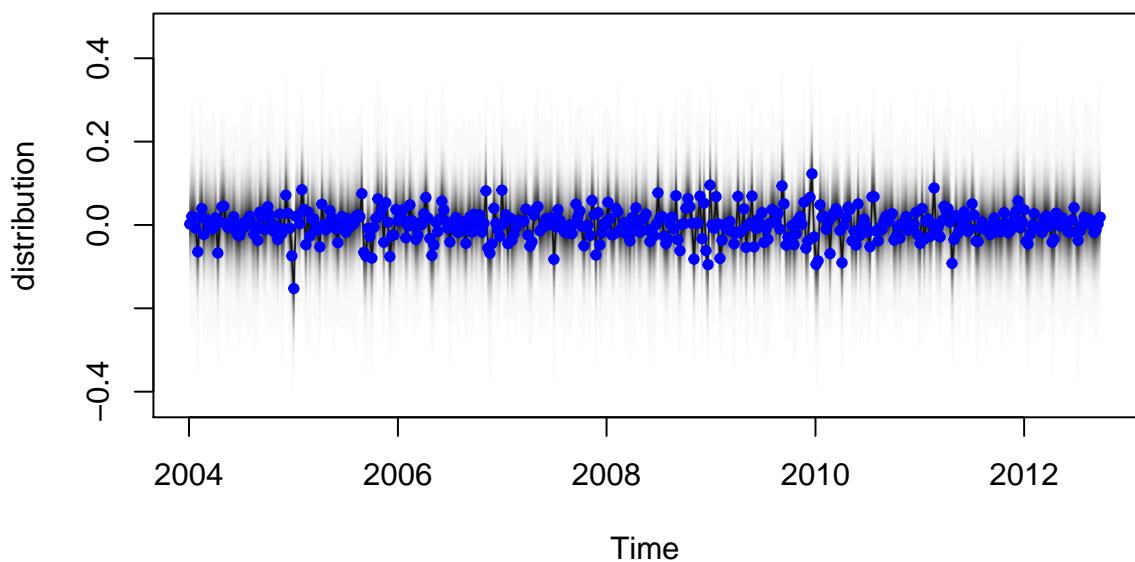
```

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

```

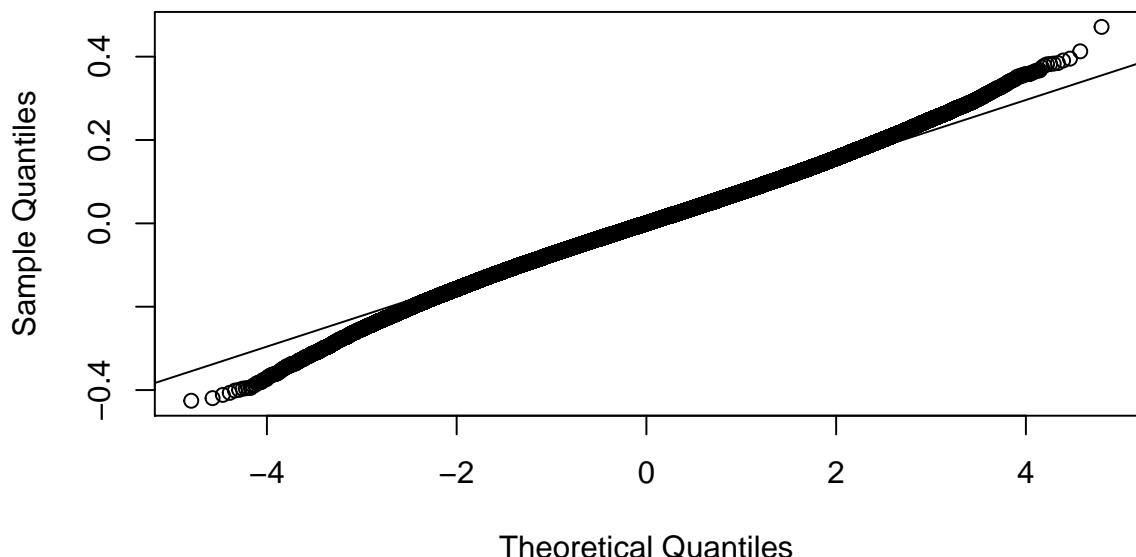
r <- residuals(bsts.reg,burn)
PlotBstsResiduals(bsts.reg,burn)

```



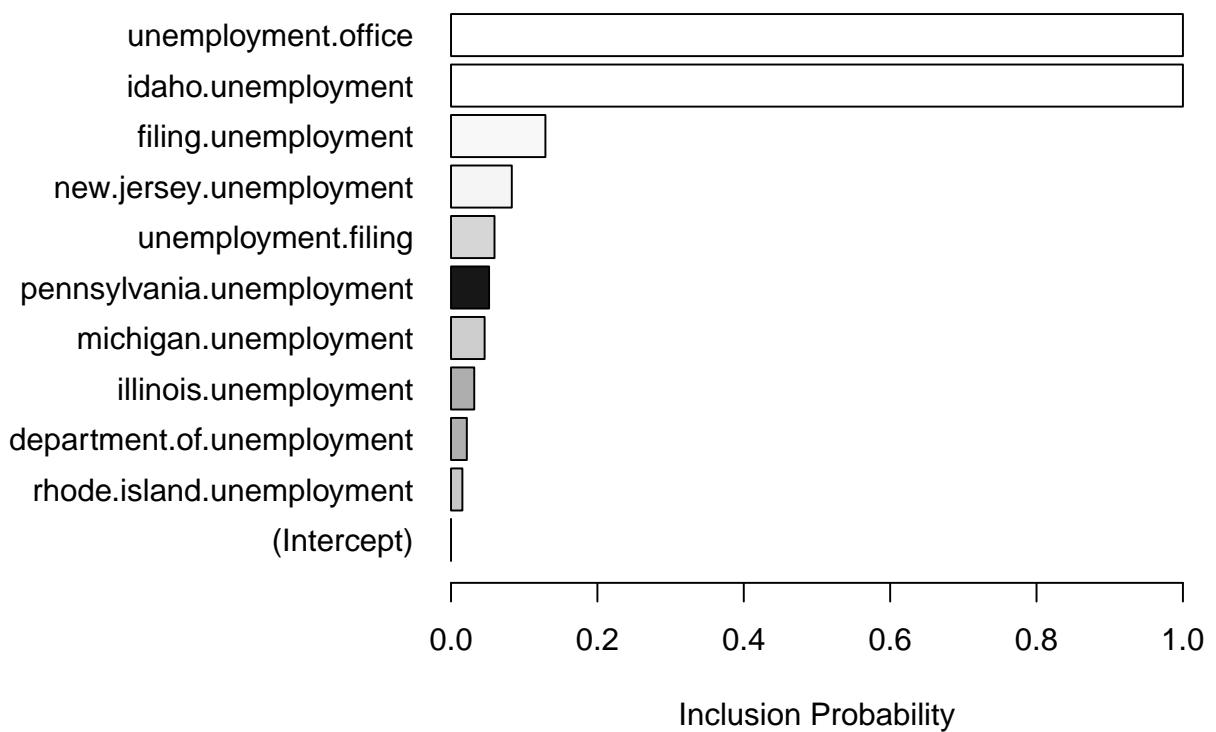
```
qqnorm(r)  
qqline(r)
```

Normal Q-Q Plot

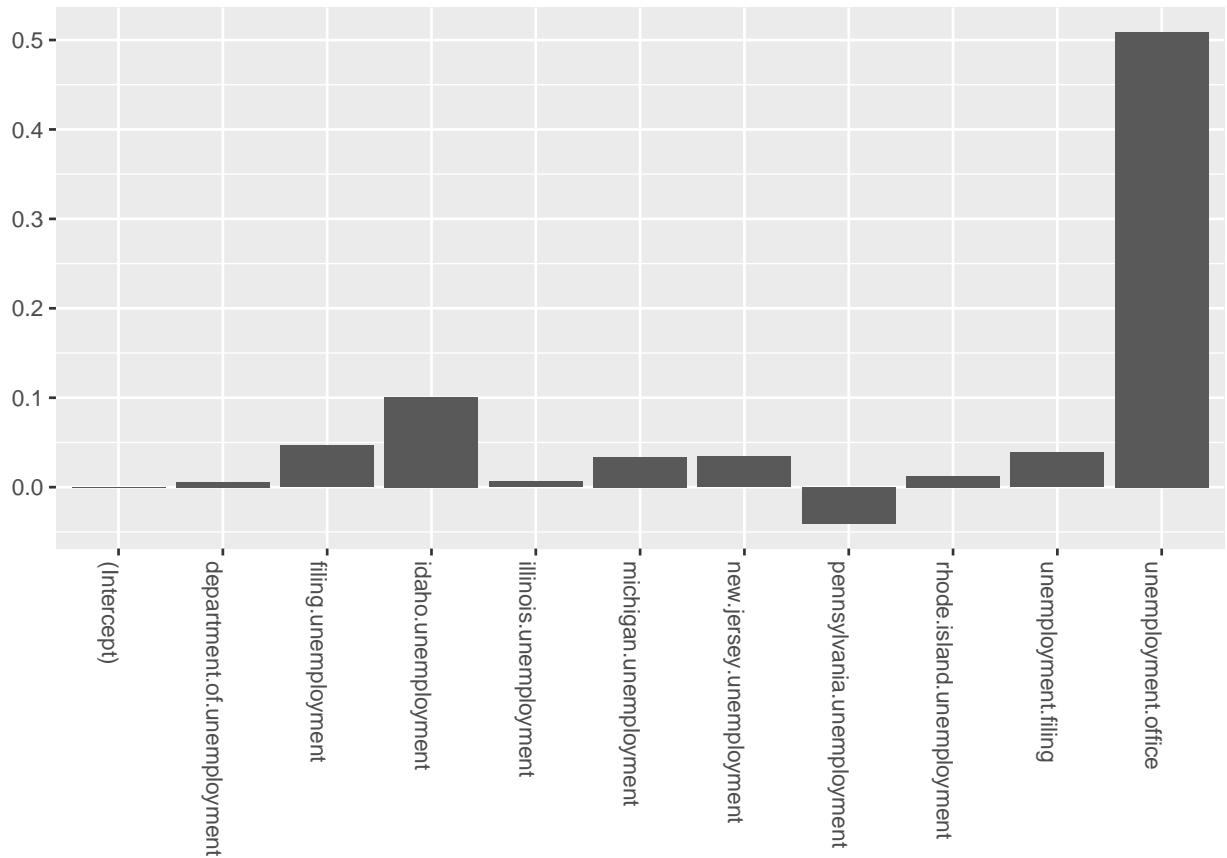


Après les tirages successifs, on peut estimer la probabilité à postérieur des variables explicatives dans le modèle.

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



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



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 distribution à priori de β de prendre en compte ces chroniques.

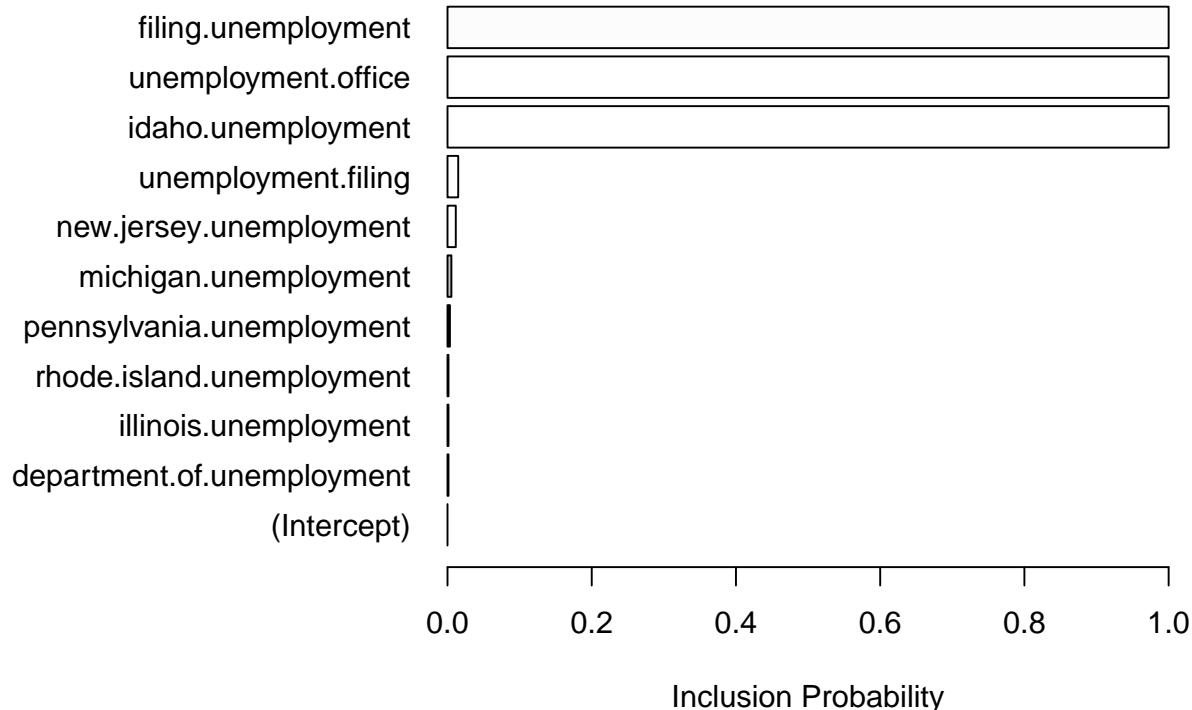
```
prior.spikes <- rep(0.1,11)
prior.spikes[3] <- 1
prior.spikes[11] <- 1

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

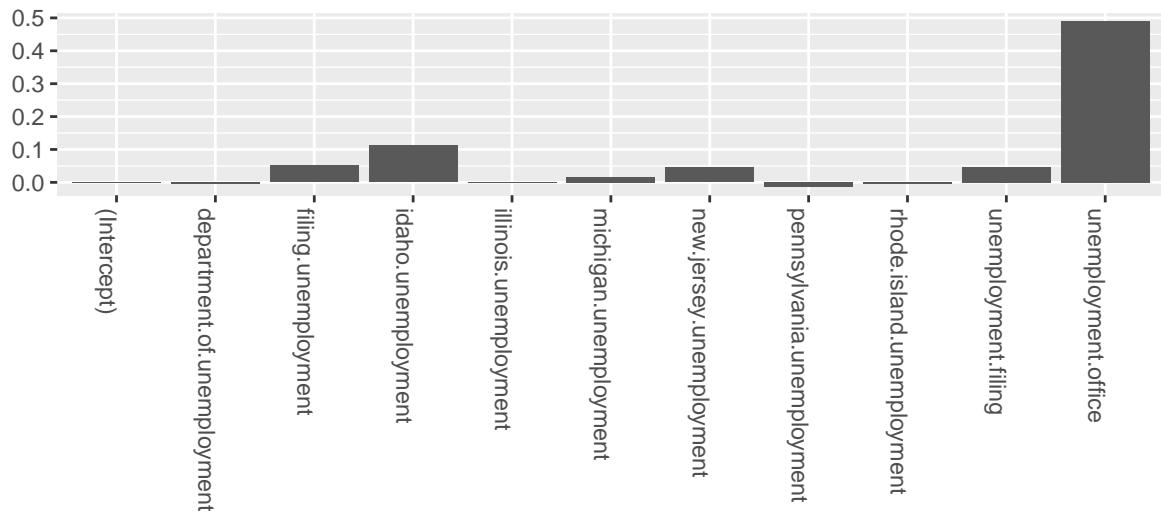
# probabilité à postériori d'inclusion des variables
plot(bststs.reg.priors, "coef", burn = SuggestBurn(0.3, bststs.reg),
      main="probabilité à postériori des variables explicatives")
```

probabilité à postériori des variables explicatives



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

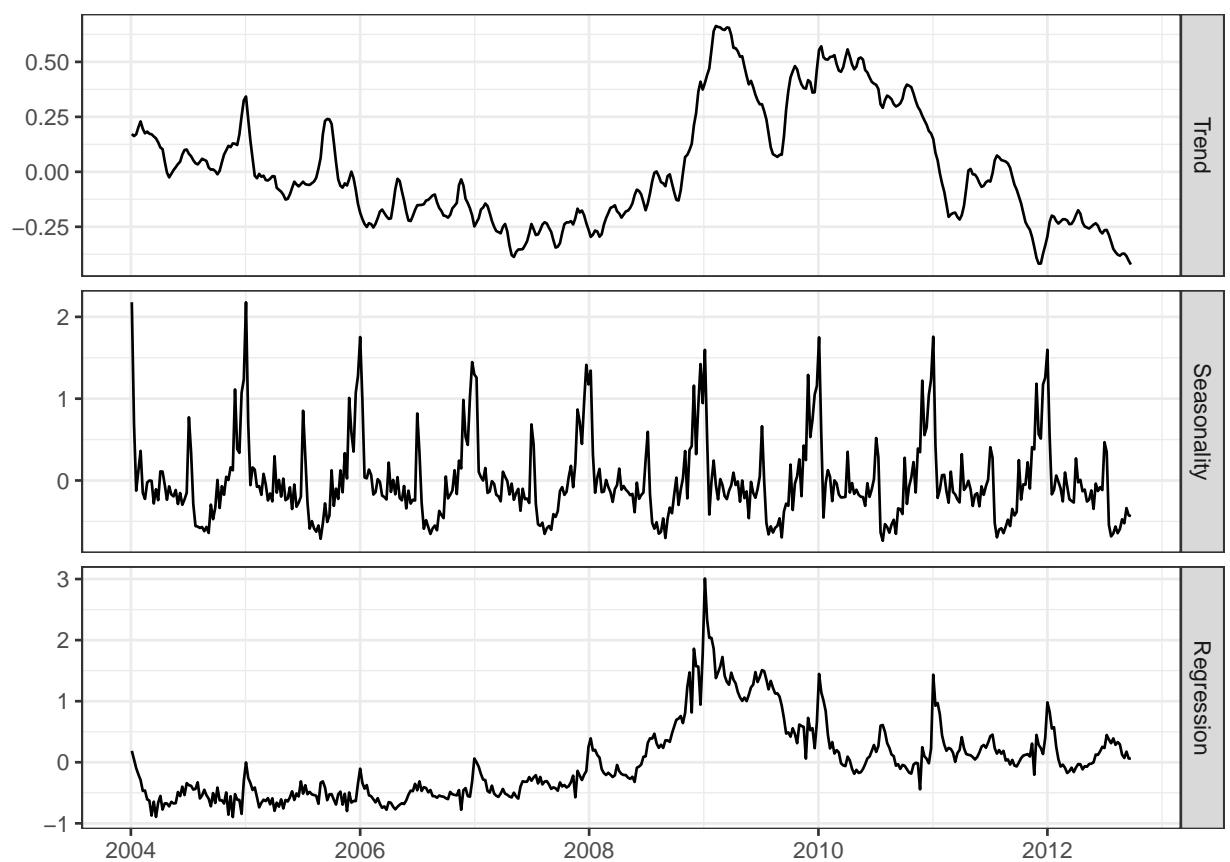
```
# coefficients
coeff <- data.frame(melt(apply(bsts.reg.priors$coefficients[-(1:burn),], 2, PositiveMean)))
coeff$Variable <- as.character(row.names(coeff))
ggplot(data=coeff, aes(x=Variable, y=value)) +
  geom_bar(stat="identity", position="identity") +
  theme(axis.text.x=element_text(angle = -90,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



Nous pouvons afficher les composantes du modèle.

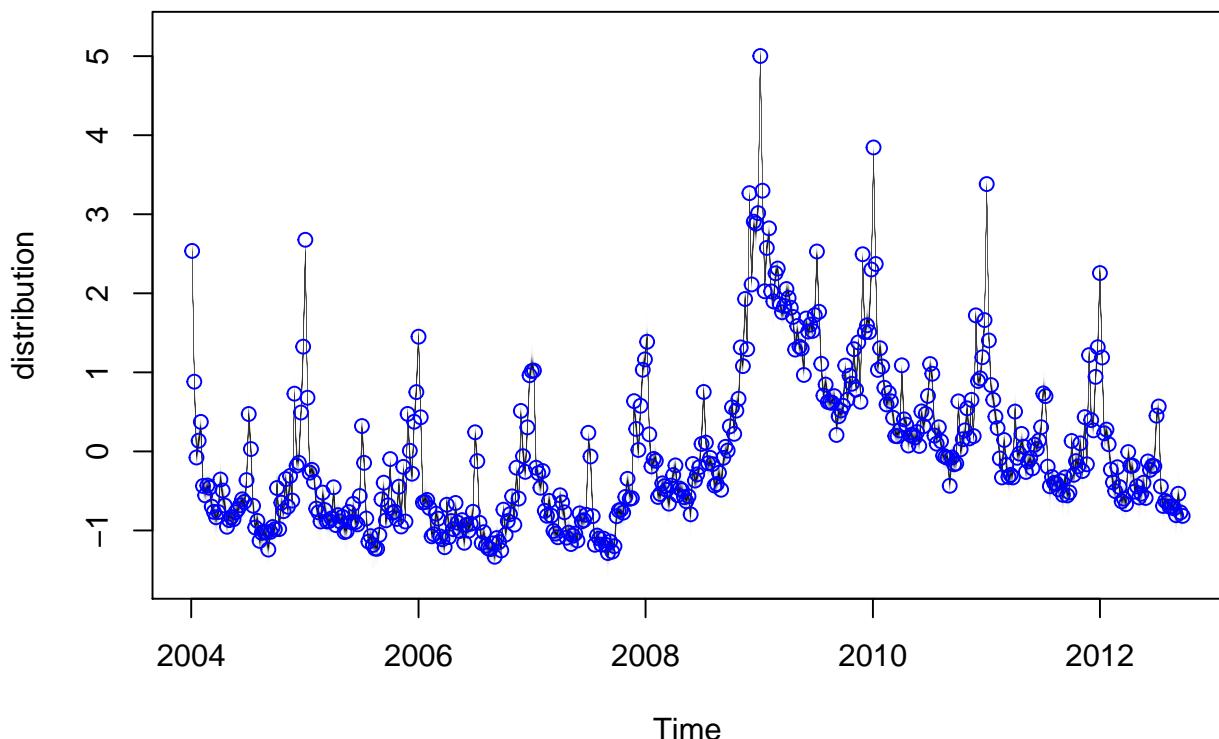
```
# composantes du modèle
components.withreg <- cbind.data.frame(
  colMeans(bsts.reg.priors$state.contributions[-(1:burn), "trend",]),
  colMeans(bsts.reg.priors$state.contributions[-(1:burn), "seasonal.52.1",]),
  colMeans(bsts.reg.priors$state.contributions[-(1:burn), "regression",]),
  as.Date(time(initial.claims)))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

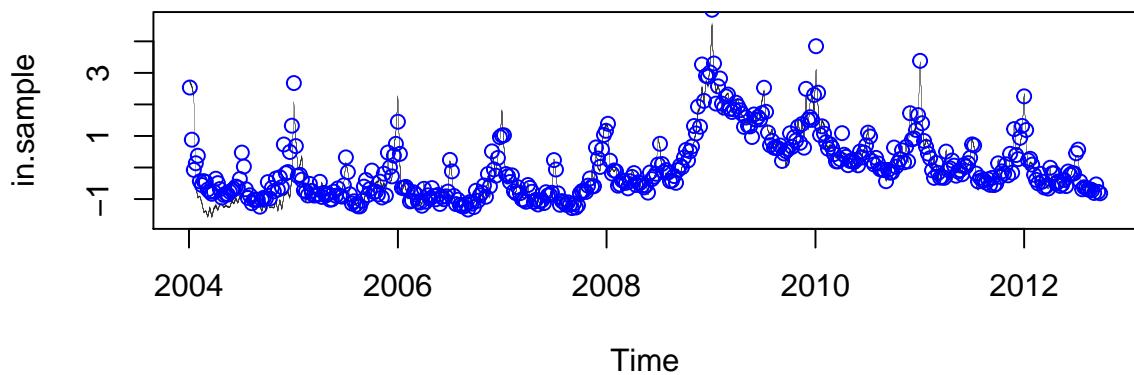
Observations et valeurs obtenues par le modèle bststs.reg



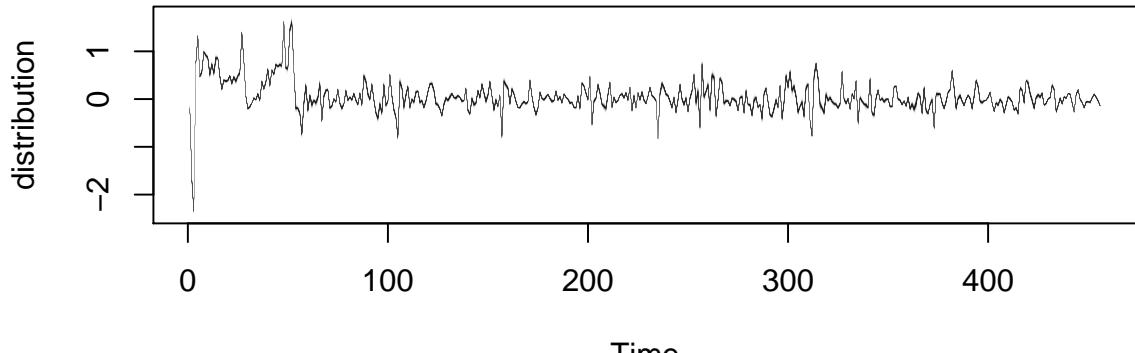
On voit que les données subissent une rupture de tendance durant l'année 2009, probablement dues à la crise économique. Le modèle réussit s'adapter à cette rupture de tendance.

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)
```



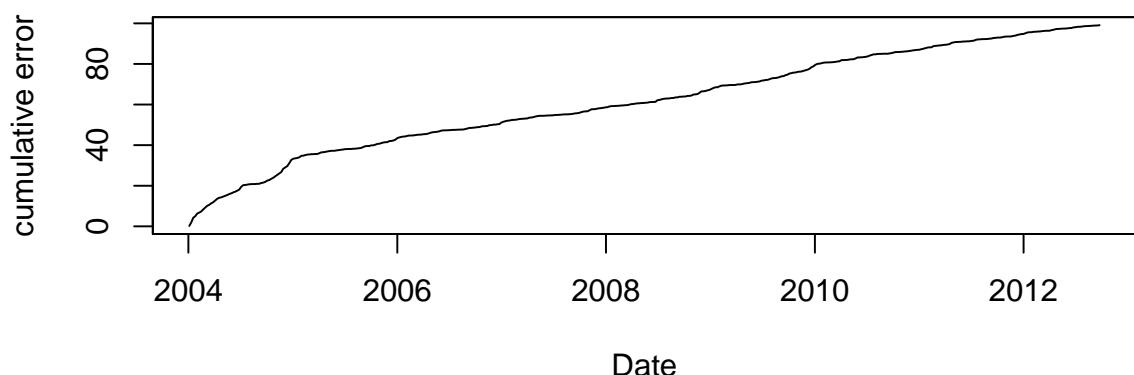
```
# evolution de l'erreur au cours du temps
# renvoie la distribution postérieure de l'erreur de prediction pour t sachant t-1
errors <- bststs.prediction.errors(bsts.reg.priors, burn = burn, standardize = TRUE)$in.sample
PlotDynamicDistribution(errors)
```



```
## NULL
```

L'erreur de prédition pour les premier t est importante. nous pouvons nous intéresser également aux erreur de prédition cumulé au temps suivant.

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



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

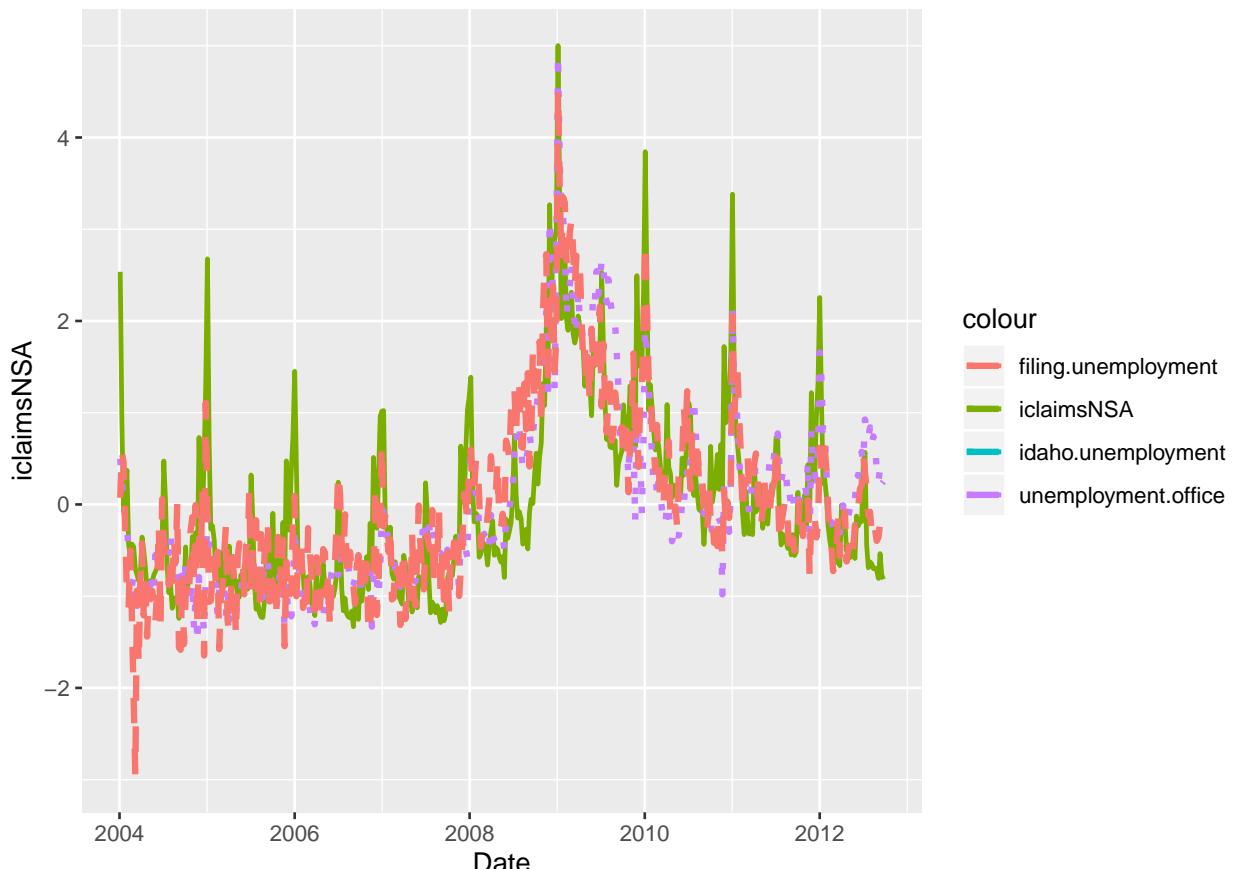
La chronique des demandes de droit au chômage et les chronique des recherches Google suivent le même comportement. Nous pouvons essayer d'adopter une approche fréquentiste et essayer de modéliser la série `iclaimsNSA` par un modèle ARIMA avec une régression linéaire sur les recherches Google `unemployment.office,idaho.unemployment`, et `filing.unemployment`. Un tel modèle s'écrit

$$(1 - B)^d X_t = (1 - B)^d f(t) + \frac{\Theta(B)}{\Phi(B)} \varepsilon_t$$

On peut voir cela comme une approche fréquentiste par rapport à la méthode précédente.

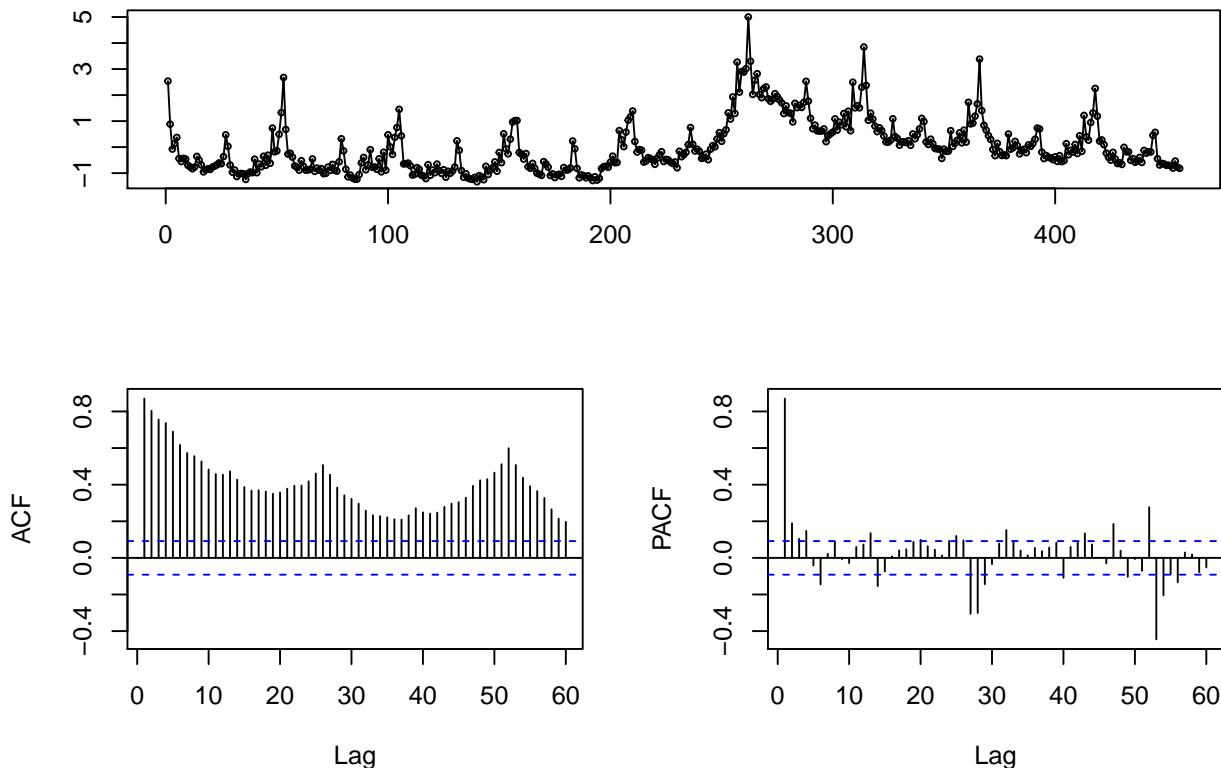
```
data(iclaims)
Y <- initial.claims
Plot <- cbind.data.frame(
  as.numeric(initial.claims$iclaimsNSA),
  as.numeric(initial.claims$idaho.unemployment),
  as.numeric(initial.claims$unemployment.office),
  as.numeric(initial.claims$filing.unemployment),
  as.Date(time(initial.claims)))

names(Plot) <- c("iclaimsNSA", "idaho.unemployment","unemployment.office","filing.unemployment","Date")
ggplot(data=Plot, aes(x=Date)) +
  geom_line(aes(y=iclaimsNSA, colour = "iclaimsNSA"), size=0.9,linetype = "solid") +
  geom_line(aes(y=idaho.unemployment, colour = "idaho.unemployment"), size=1.2,linetype = "blank") +
  geom_line(aes(y=unemployment.office, colour = "unemployment.office"), size=1.2,linetype = "dotted") +
  geom_line(aes(y=filing.unemployment, colour = "filing.unemployment"), size=1.2,linetype = "dashed")
```



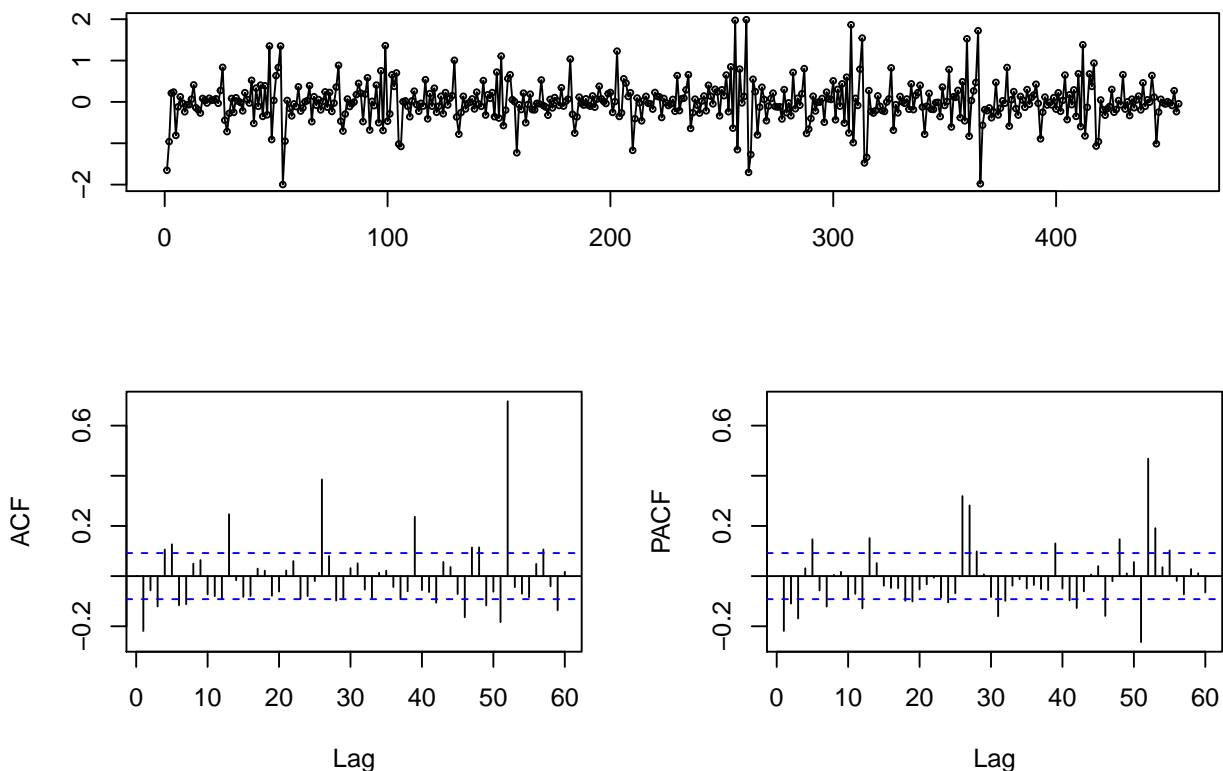
```
# corrélogramme de la série  
tsdisplay(initial.claims$iclaimsNSA,lag.max = 60)
```

initial.claims\$iclaimsNSA



```
# corrélogramme de la série différenciée  
tsdisplay(diff(initial.claims$iclaimsNSA),lag.max = 60)
```

diff(initial.claims\$iclaimsNSA)



```
# le kurtosis est supérieur à 3
kurtosis(diff(initial.claims$iclaimsNSA))

## [1] 3.451918

# le test de box_Pierce nous permet de rejeter l'hypothèse d'un bruit blanc gaussien
Box.test(initial.claims$iclaimsNSA)

##
## Box-Pierce test
##
## data: initial.claims$iclaimsNSA
## X-squared = 345.64, df = 1, p-value < 2.2e-16
```

Ainsi nous ne pouvons pas appliquer ce modèle sur cette chronique. Nous pourrions par ailleur essayer d'appliquer un modèle de régression multiple classique dans, mais les résidus qui en découlent ne composent pas un bruit blanc, ce qui montre l'utilité de l'approche bayésienne.

5 Comparaison de l'approche bayésienne et de l'approche fréquentiste

Nous pouvons comparer les deux approches que nous avons mit en place : une approche fréquentiste tout d'abord pour estimer les paramètres du modèle en maximisant la vraisemblance des données. Ensuite l'approche bayésienne où nous posont une estimation à priori qui est actualisée par les données.

Dans le premier cas nous obtenons une estimation directe du paramètre optimal à appliquer, avec une variance déterminée par l'information de fisher. Dans le second cas nous obtenons une distribution de probabilité sur laquelle nous effectuons un choix.

Le principale reproche fait à l'approche bayésienne est la subjectivité introduite dans le modèle, mais d'une part des modèles de lois à priori non-informatives ont été développés et cette subjectivité peut permettre d'éviter des phénomènes d'overfitting ou de biais dans l'échantillon des données.

D'autre part les intervalles de confiance à postériori semblent plus naturels à utiliser, et permettent de se renseigner sur l'influence des données sur les hypothèse à priori. Nous avons pu voir comme autre avantage majeure de la méthode est qu'elle n'est pas impacté par des ruptures de tendances où dans une approche fréquentiste il aurait fallu utiliser des modèles non linéaires plus complexes. De plus les intervalles de confiances pour une prédiction à long terme sont beaucoup plus resserrés .

Cependant dans la converge de la méthode MCMC , nous pouvont être induit en erreur sur la convergence du modèle, il est possible que certaines valeurs de la distribution à postériori ne soient pas explorées pour cause de probabilités trop faible dans la chaîne de Markov. Il peut également être perturbant de considérer que plusieurs simulations ne renverrons pas les mêmes estimations pour les paramètres du modèle de part l'aspect aléatoire de la méthode. De plus l'approche fréquentiste se montre utilise pour rejeter des hypothèses sur les données avec des statistiques de test.

Conclusion

Nous avons ainsi pu étudier une nouvelle approche statistique pour paramétriser nos modèles. Les méthodes de Monté-Carlo par chaîne de Markov possèdent de nombreuses autres implémentations que l'algorithme d'échantillonnage de Gibbs que nous avons présenté, pour une réduction des corrélations entre les tirages successifs et une meilleure exploration de toutes les valeurs prises par la chaîne. Ces méthodes sont le pilier de nombreuses applications d'analyse de données comme l'algorithme d'espérance-maximisation où l'allocation de Dirichlet latente. Ces méthodes de classification peuvent notamment être implémentées de manière non paramétriques.

Références

- [1] Wikipédia Markov Chain.
- [2] anhdanggit. Google Queries for Nowcasting New Housing Sales.
- [3] Isabelle Drouet. *Le bayesianisme aujourd’hui*. Edition Matérilogiques, 2016.
- [4] Edward I. George and Robert E. McCulloch. Variable Selection Via Gibbs Sampling. *Journal of the American Statistical Association*, 88(423) :881–889, Sep 1993.
- [5] S.H. Koopman J. Durbin. A simple and efficient simulation smoother for state space time series analysis. October 2011.
- [6] Adrian E. Raftery Jennifer A. Hoeting, David Madigan and Chris T. Volinsky. Bayesian Model Averaging : A Tutorial. *Statistical Science*, 14(4) :382–417, 1999.
- [7] Siem Jan Koopman. Disturbance Smoother for State Space Models. *Biometrika*, 80(1) :117–126, Mar 1993.
- [8] Kim Larsen. Sorry ARIMA, but I’m Going Bayesian, 2016.
- [9] Chrisitian Robert. *L’analyse statistique bayesienne*. Economica, 1995.
- [10] Steven L. Scott and Hal Varian. Predicting the Present with Bayesian Structural Time Series . *International Journal of Mathematical Modelling and Numerical Optimisation*, 5 :4–23, 2014.
- [11] Cyrill Stachniss. Slam course - 03 - kalman filter - cyrill stachniss.
- [12] Frédéric Sur. Cours de deuxième année : Modélisation des séries temporelles, 2018.
- [13] Denis Villemonais. Probabilités Cours SG 231, 2017.
- [14] B. Walsh. Markov Chain Monte Carlo and Gibbs Sampling, Lecture Notes for EEB , 2002.