Rapport II

Olivier Berthier

2024-05-31

L'objet de ce projet est d'essayer des méthodes différentes de celles présentées dans le premier rapport, soit le modèle ARIMAX et l'approche bayésienne. Dans ce but, nous avons délibérément choisi deux jeux de données très corrélés. Le premier est le taux de chômage dans les Pays de la Loire de 1990 à 2020, le deuxième est le taux de chômage en France sur la même période, que nous utiliserons comme variable exogène. Les deux jeux proviennent du site officiel de l'INSEE. Nous avons bien conscience que les statistiques du chômage des Pays de la Loire sont elles-mêmes "intégrées" à celles de la France, ce qui assure une corrélation forte "par définition". Notre intérêt étant ici les méthodes statistiques en elles-mêmes et non l'analyse du phénomène chômage, nous acceptons cette limite théorique.

```
library(zoo)
library(ggseas)
library(caschrono)
library(forecast)
library(tseries)
library(readr)
library(bayesforecast)
library(ggplot2)
library(gridExtra)
library(bsts)
```

Importation des données du taux de chômage des Pays de la Loire:

```
chomage_pdll <- read_delim("C:/Users/olivi/OneDrive - Université Paris-Dauphine/Documents ODD
gram/R pojects/Jonathan/data rappport II/chomage_pdll.csv",
    delim = ";", escape_double = FALSE, trim_ws = TRUE)
```

```
## Rows: 120 Columns: 2
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ";"
## chr (1): Trimestre
## dbl (1): tx_chomage
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Exploration des données

```
head (chomage_pdll)
```

```
## # A tibble: 6 × 2
##   Trimestre tx_chomage
##   <chr>          <dbl>
## 1 1990-T1          8.1
## 2 1990-T2          8
## 3 1990-T3          7.9
## 4 1990-T4          7.7
## 5 1991-T1          7.7
## 6 1991-T2          7.9
```

```
summary(chomage_pdll)
```

```
##   Trimestre           tx_chomage
##  Length:120         Min.   : 5.700
##  Class :character   1st Qu.: 7.200
##  Mode  :character   Median : 7.950
##                     Mean   : 7.953
##                     3rd Qu.: 8.800
##                     Max.   :10.100
```

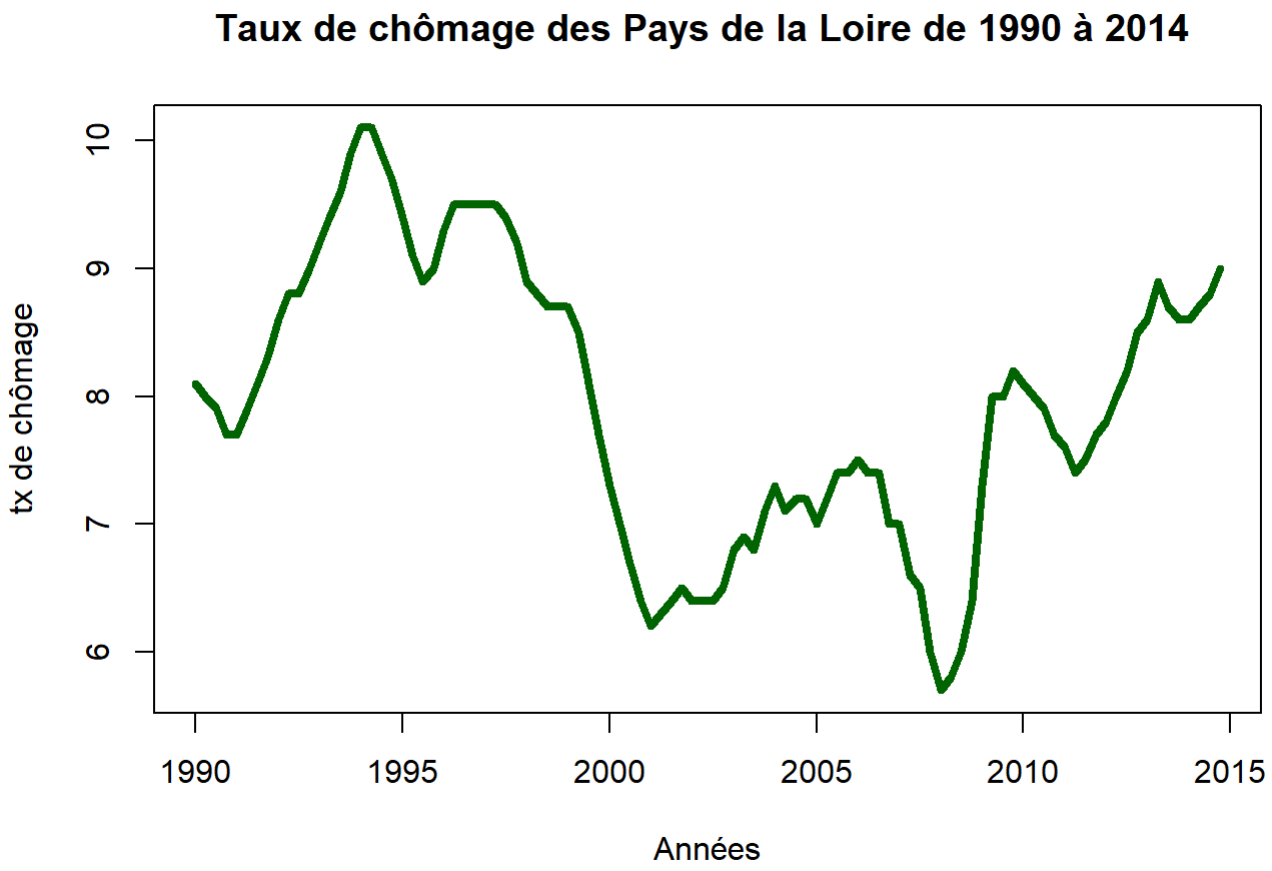
Le jeu de données contient 120 trimestres, les chiffres sont en pourcentage de la population active. Le taux de chômage minimal sur cette période est de 5.7%, le maximum est de 10.1%.

Transformation des données en “ts”:

```
vec=as.vector(t(as.matrix(chomage_pdll[,2])))
d=ts(vec, start = c(1990, 1), frequency = 4)
```

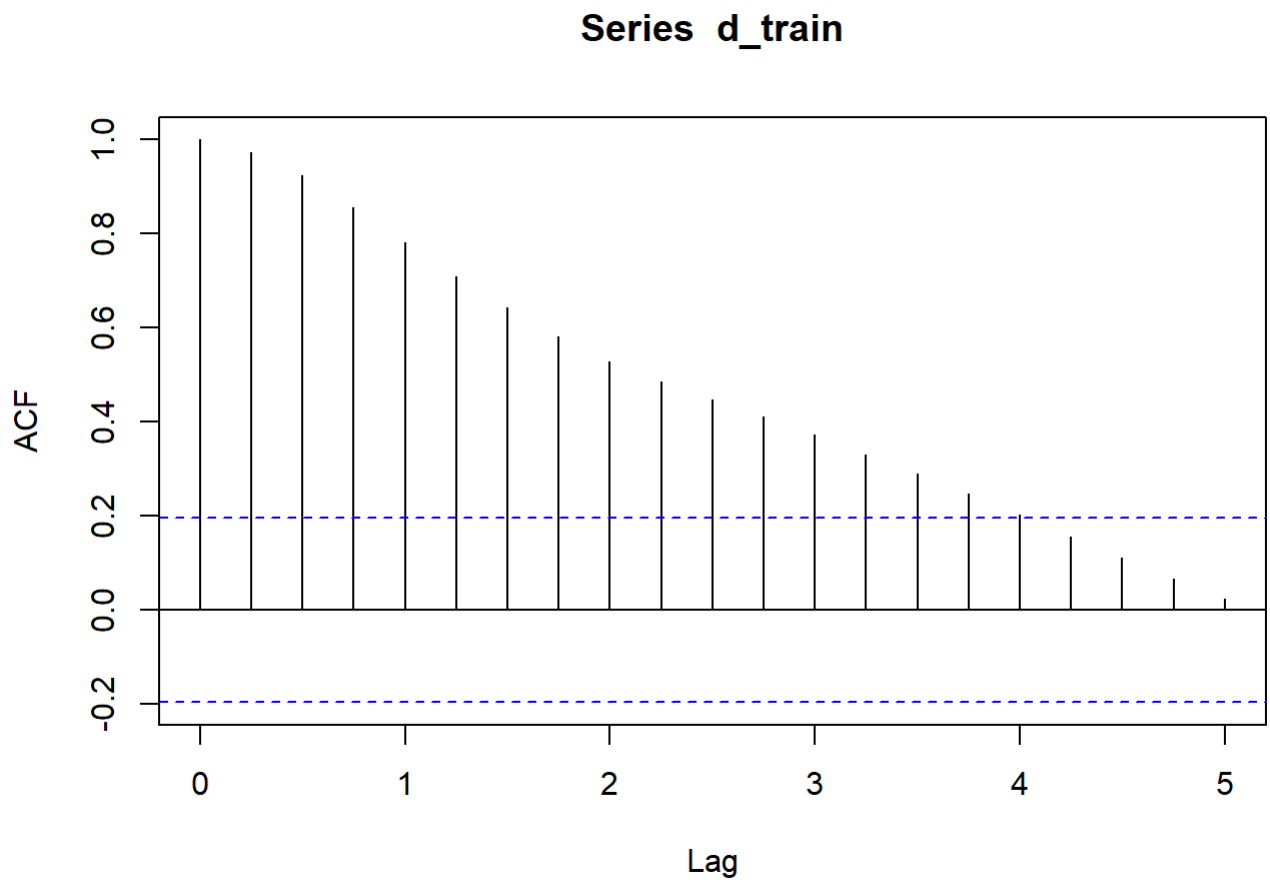
Nous séparons les données en 2 groupes, celui d’entraînement (**d_train**) et celui de test (**d_test**). Nous isolons 5 années, soit 20 données. Le jeu d’entraînement comporte donc 100 données (25 ans).

```
d_train <- window(d, end = c(2014, 4))
d_test <- window(d, start = c(2015, 1))
plot (d_train, main = "Taux de chômage des Pays de la Loire de 1990 à 2014 ", col = "darkgree
n", ylab="tx de chômage", xlab = "Années",lwd= 4)
```

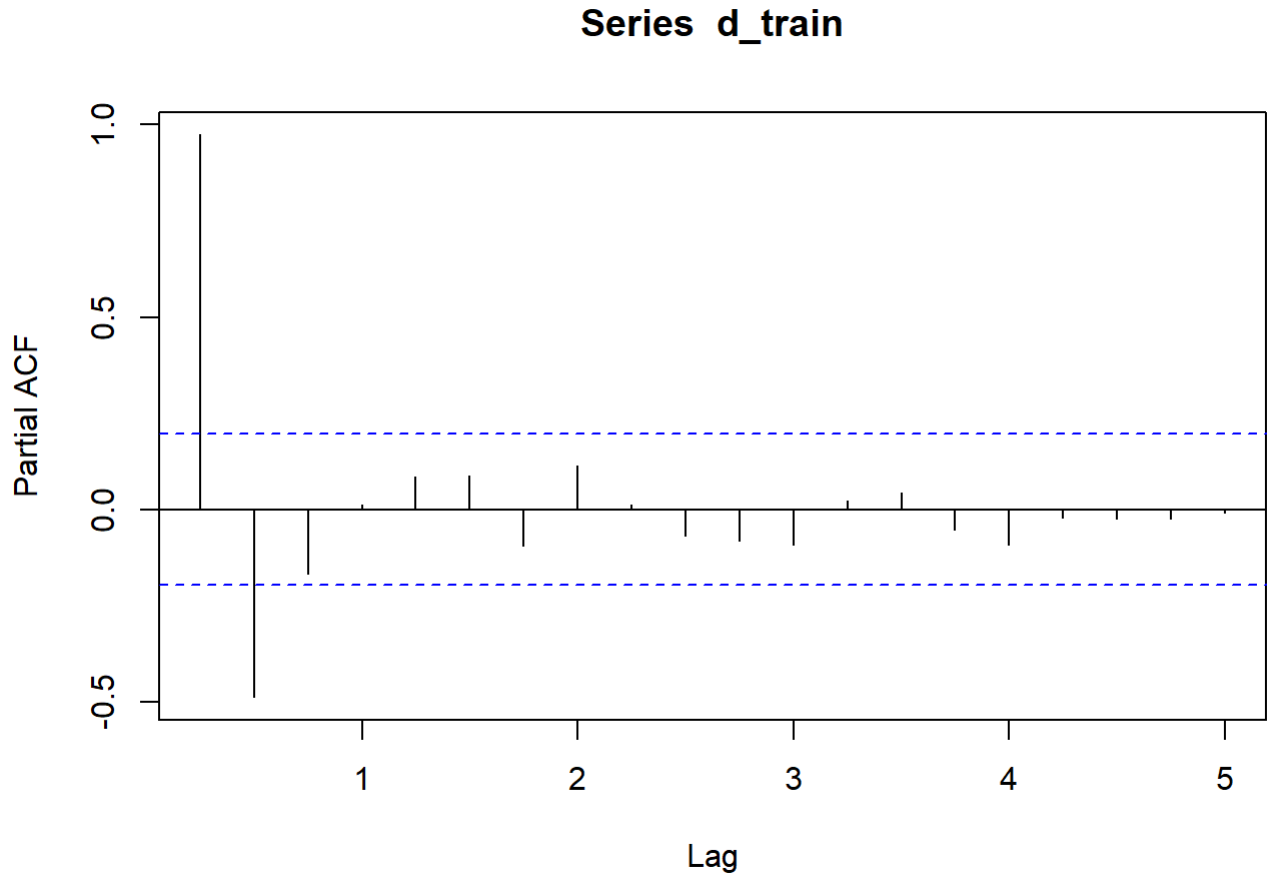


La série **d_train** présente un comportement assez erratique, mais pas totalement aléatoire. Des périodes émergent: forte croissance du chômage de 1990 à 1994, suivie d'une baisse importante jusqu'en 2000, puis une augmentation à nouveau jusqu'en 2015, avec un creux important en 2007, où la série atteint son minimum, une remontée brutale (sans doute liée à la crise économique de 2008), et un creux moins important en 2011. Il n'y a pas de tendance globale. Cependant, graphiquement, on peut aisément constater que les valeurs oscillent autour d'un taux de chômage moyen d'environ 8%. Aucune saisonnalité n'est évidente.

```
acf(d_train)
```


Series d_train
ACF
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
0
1
2
3
4
5
Lag

```
pacf(d_train)
```


Series d_train
Partial ACF
1.0
0.5
0.0
-0.5
1
2
3
4
5
Lag

```
Box.test(d_train,lag=20,type="Box-Pierce")
```

```
##
##  Box-Pierce test
##
## data:  d_train
## X-squared = 575.93, df = 20, p-value < 2.2e-16
```

Voyons si une stationnarisation est nécessaire, étant donné qu'il n'y a pas de tendance ni de saisonnalité évidente. Faisons un test de Dickey-Fuller sur la série brute:

```
adf.test(d_train)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  d_train
## Dickey-Fuller = -1.4626, Lag order = 4, p-value = 0.7992
## alternative hypothesis: stationary
```

On ne peut pas rejeter la non-stationnarité de la série. Testons la série avec différentiation d'ordre 1:

```
adf.test(diff(d_train,lag=1,difference=1))
```

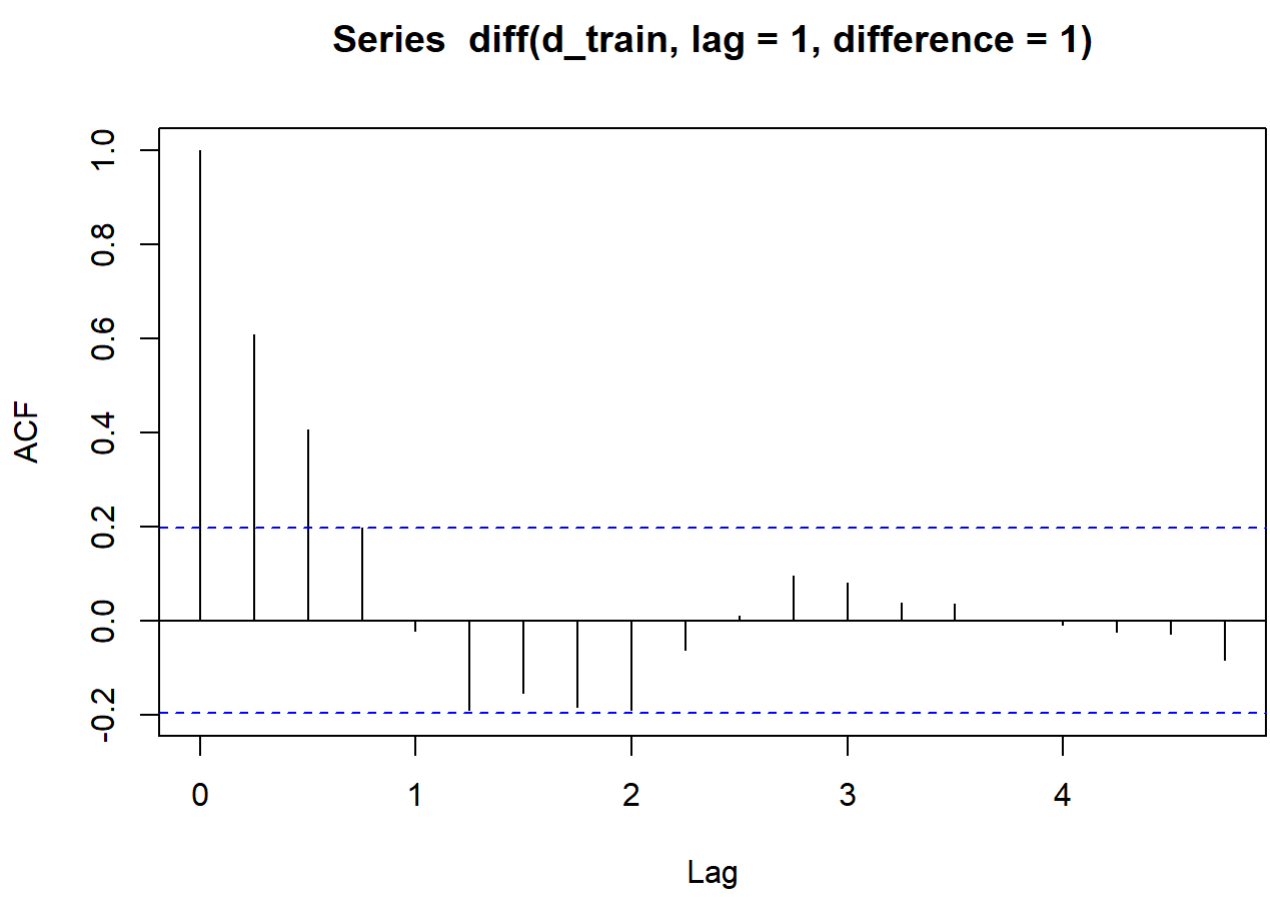
```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(d_train, lag = 1, difference = 1)
## Dickey-Fuller = -4.7918, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

On ne peut pas rejeter la stationnarité de la série différenciée.

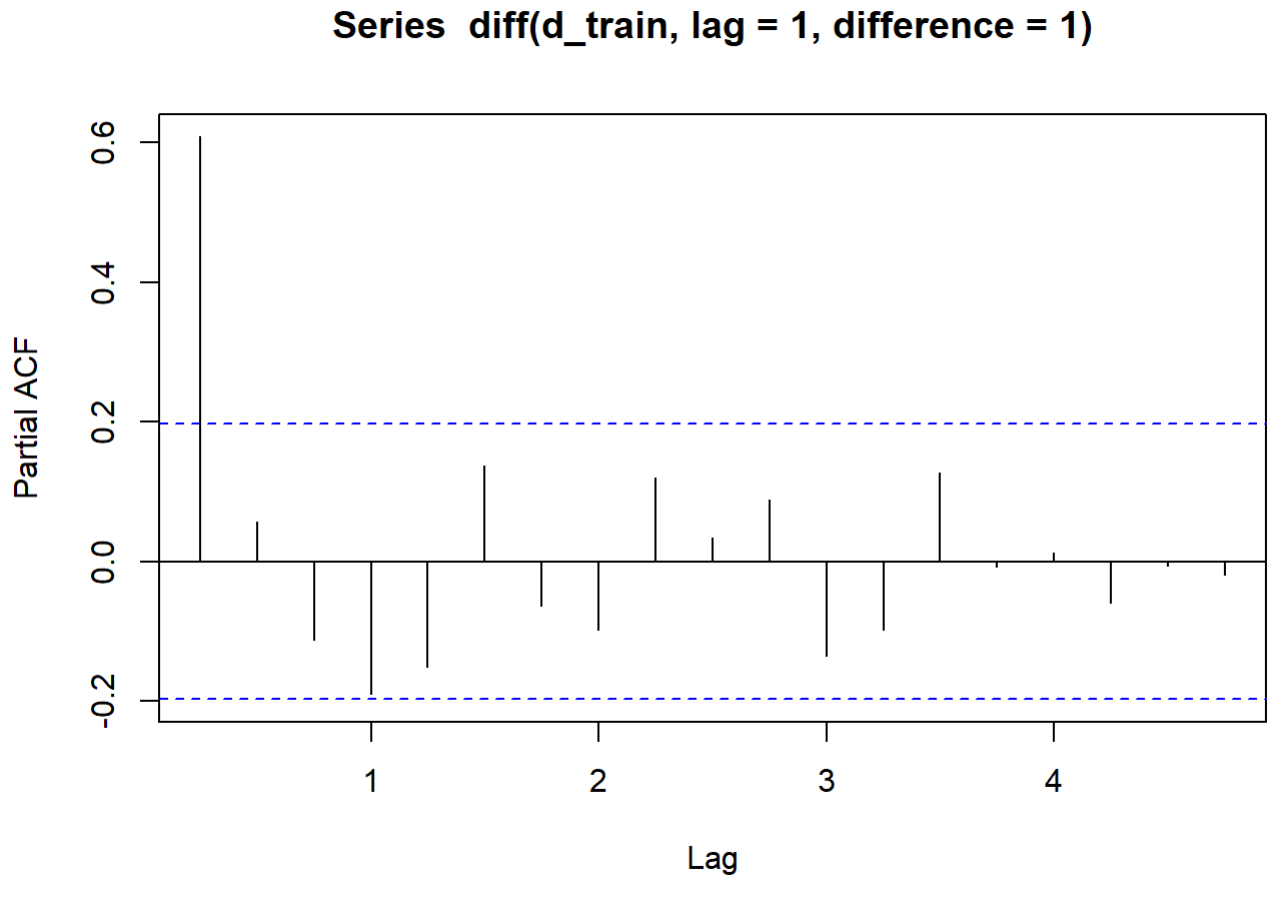
Élaboration du modèle

Étudions l'autocorrélation de la série différenciée:

```
acf(diff(d_train,lag=1,difference=1))
```




```
pacf(diff(d_train,lag=1,difference=1))
```



Oscillations de l'ACF, PACF presque nul à partir du rang 2. Apparemment, le modèle ressemble à un AR 1.

```
#arima_110<- Arima(diff(d_train,lag=1,difference=1),order=c(1,0,0))
arima_110<- Arima(d_train,order=c(1,1,0))
summary (arima_110)
```

```
## Series: d_train
## ARIMA(1,1,0)
##
## Coefficients:
##          ar1
##       0.6083
## s.e.  0.0790
##
## sigma^2 = 0.03379:  log likelihood = 27.48
## AIC=-50.96   AICc=-50.84   BIC=-45.77
##
## Training set error measures:
##                       ME     RMSE       MAE        MPE     MAPE      MASE
## Training set 0.005028885 0.181977 0.1404496 0.07389482 1.846746 0.2460431
##                     ACF1
## Training set -0.03044099
```

```
Box.test(arima_110$residuals,lag=20,type="Box-Pierce")
```

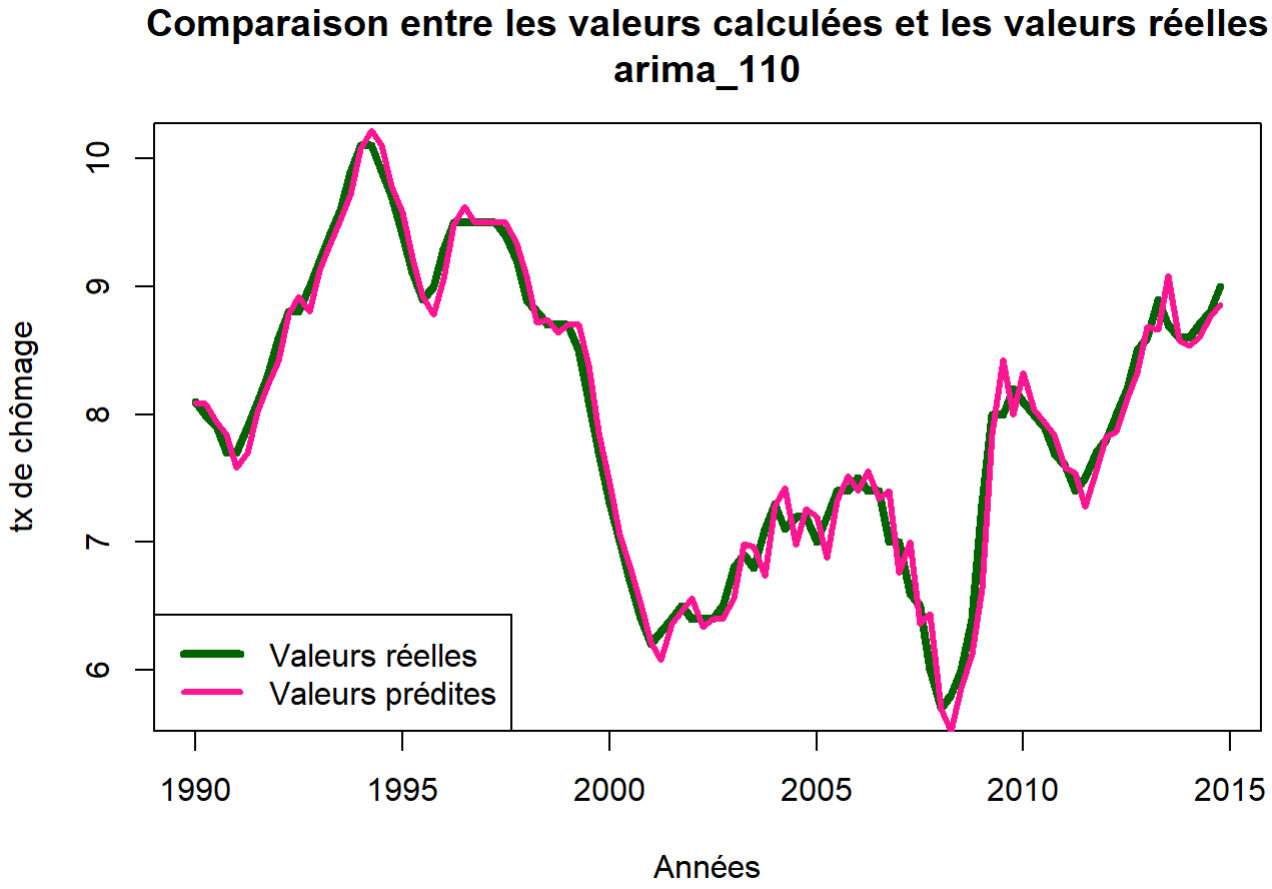
```
##
##  Box-Pierce test
##
## data:  arima_110$residuals
## X-squared = 15.609, df = 20, p-value = 0.7406
```

```
t_stat(arima_110)
```

```
##             ar1
## t.stat 7.701755
## p.val  0.000000
```

La blancheur des résidus ne peut pas être rejetée (p-valeur de 0.7406). Coefficient significatif. Pas d'éventuelle colinéarité à tester: un seul coefficient présent dans notre modèle. Notons que la fonction *auto.arima()* (non représenté ici) propose aussi un modèle AR 1.

```
# Tracer les valeurs prédites et les valeurs réelles
plot(d_train, type = "l", col = "darkgreen", main = "Comparaison entre les valeurs calculées
et les valeurs réelles\n arima_110 ", lwd= 4, ylab="tx de chômage", xlab = "Années")
lines(fitted(arima_110), col = "deeppink", lwd= 3)
legend("bottomleft", legend = c("Valeurs réelles", "Valeurs prédites"), col = c("darkgreen",
"deeppink"), lwd = c(4,3))
```



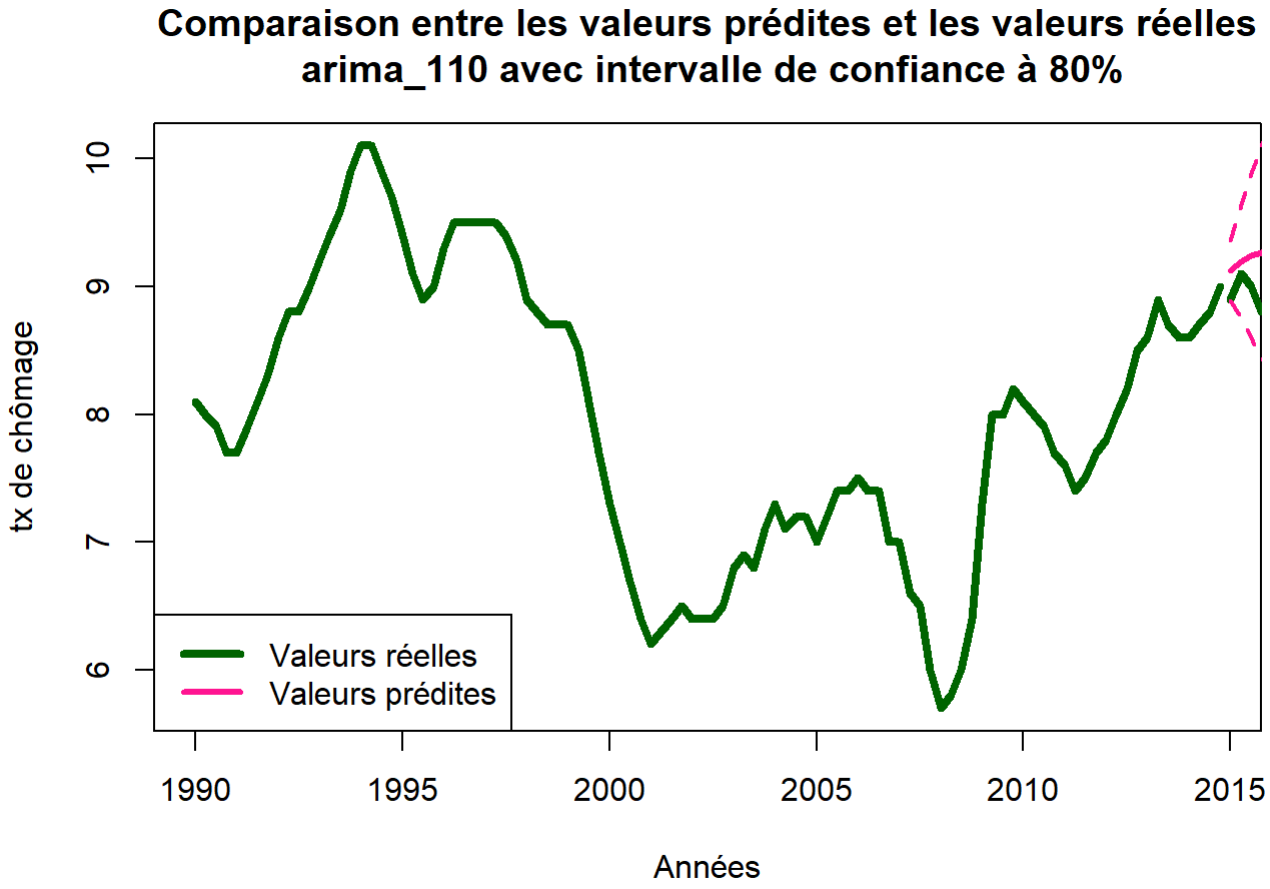
Sans surprise, notre modèle présente un peu de retard lors des changements brutaux d'évolution des valeurs réelles, c'est particulièrement visible lors des pics.

Prédictions

```
# Calculer les prédictions et les intervalles de confiance
forecast_results <- forecast(arima_110, h = 20, level = 80)

# Extraire les valeurs prédites et les intervalles de confiance inférieurs et supérieurs
predicted_values <- forecast_results$mean
lower_ci <- forecast_results$lower[, "80%"]
upper_ci <- forecast_results$upper[, "80%"]

# Tracer les valeurs prédites avec intervalle de confiance
plot(d_train, col = "darkgreen", main = "Comparaison entre les valeurs prédites et les valeur
s réelles\n arima_110 avec intervalle de confiance à 80%",lwd= 4, ylab="tx de chômage", xlab
= "Années")
lines (d_test, col = "darkgreen",lwd= 4)
lines(predicted_values, col = "deeppink",lwd= 3)
lines(lower_ci, col = "deeppink", lty = 2,lwd= 2)  # Intervalle de confiance inférieur
lines(upper_ci, col = "deeppink", lty = 2,lwd= 2)  # Intervalle de confiance supérieur
legend("bottomleft", legend = c("Valeurs réelles", "Valeurs prédites"),
       col = c("darkgreen", "deeppink"), lwd= c(4,3))
```



Deux remarques sur nos prédictions:

- Sans surprise, le modèle ne prévoit pas la baisse brutale du chômage qui s'amorce en 2015, mais poursuit au contraire l'augmentation de la période qui précède la fin des données d'entraînement (2012 à 2015).
- Les valeurs réelles sont dans notre intervalle de confiance à 80%.

Notre modèle **arima_110** peut être utile pour encadrer les valeurs futures du taux de chômage mais n'est pas pertinent s'il s'agit d'en prédire les valeurs précises. Voyons si nous parvenons à faire mieux en incorporant des données exogènes.

ARIMAX

Exploration des données du taux de chômage de la France entière.

```
chomage_fr <- read_delim("C:/Users/olivi/OneDrive - Université Paris-Dauphine/Documents ODD g
ram/R pojects/Jonathan/data rappport II/chomage_france.csv",
    delim = ";", escape_double = FALSE, trim_ws = TRUE)
```

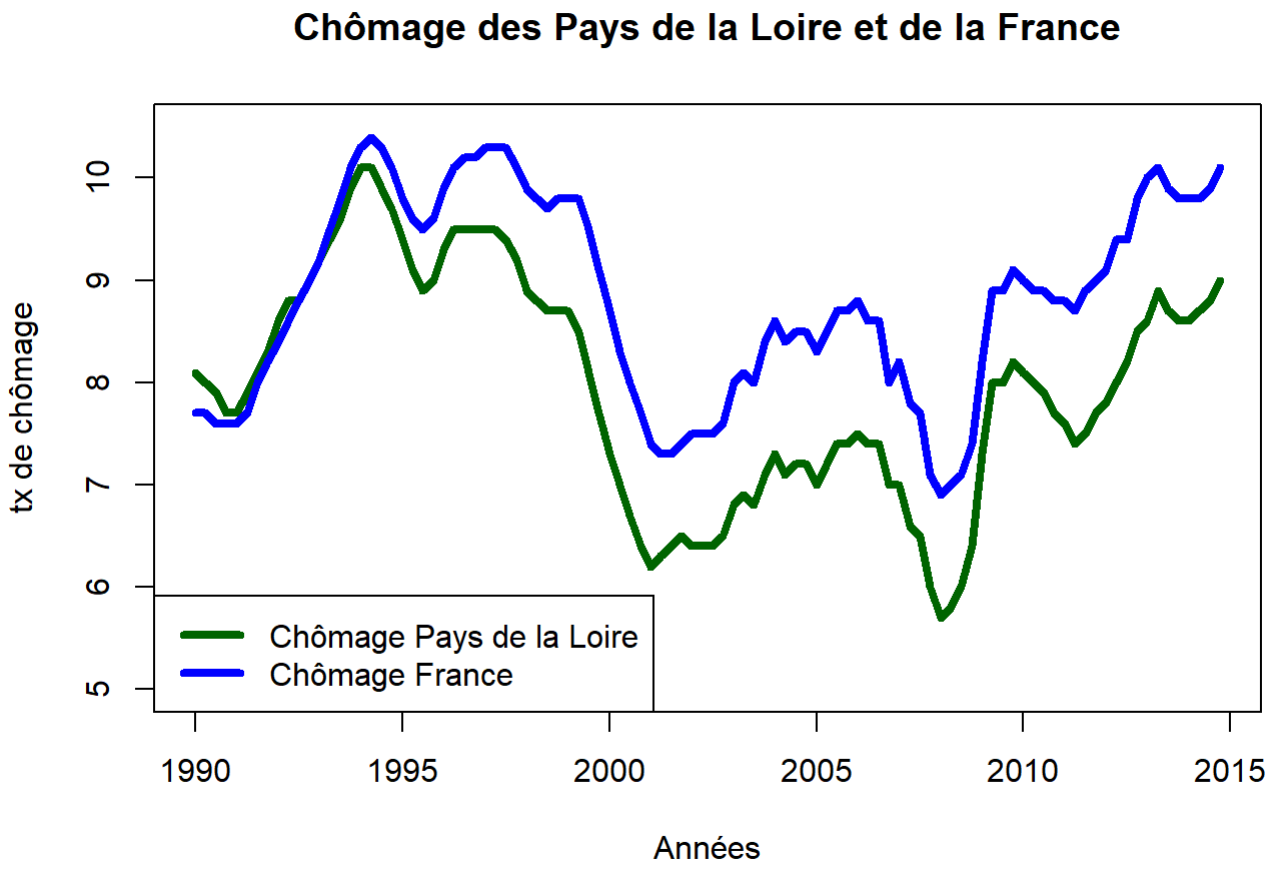
```
## Rows: 120 Columns: 2
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ";"
## chr (1): Trimestre
## dbl (1): tx_chomage
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Transformation des données en “ts” et séparation en deux groupes **df_train** et **df_test** (et deux vecteurs **vecf_train** et **vecf_test** pour une utilisation en données exogènes dans notre modèle).

```
vecf=as.vector(t(as.matrix(chomage_fr[,2])))
# Créer vecf_train avec les 100 premières valeurs de vecf
vecf_train <- head(vecf, 100)
# Créer vecf_test avec les 20 dernières valeurs de vecf
vecf_test <- tail(vecf, 20)

# Diviser les données en un ensemble d'entraînement et un ensemble de test
df=ts(vecf,start = c(1990, 1), frequency = 4)
df_train <- window(df, end = c(2014, 4))
df_test <- window(df, start = c(2015, 1))

# Tracer les deux séries temporelles sur le même graphique
plot(d_train, main = "Chômage des Pays de la Loire et de la France", col = "darkgreen",lwd=
4, ylab="tx de chômage", xlab = "Années", ylim=c(5,10.5))
lines(df_train, col = "blue",lwd= 4)
legend("bottomleft", legend = c("Chômage Pays de la Loire", "Chômage France"), col = c("darkg
reen", "blue"),lwd= c(4,4))
```



Les deux courbes suivent à peu près la même évolution. Notons cependant que l’écart est presque nul au début des années 90 pour se creuser ensuite et rester assez stable à partir des années 2000, au bénéfice des Pays de la Loire qui ont environ 1% de chômage de moins que l’ensemble de la France.

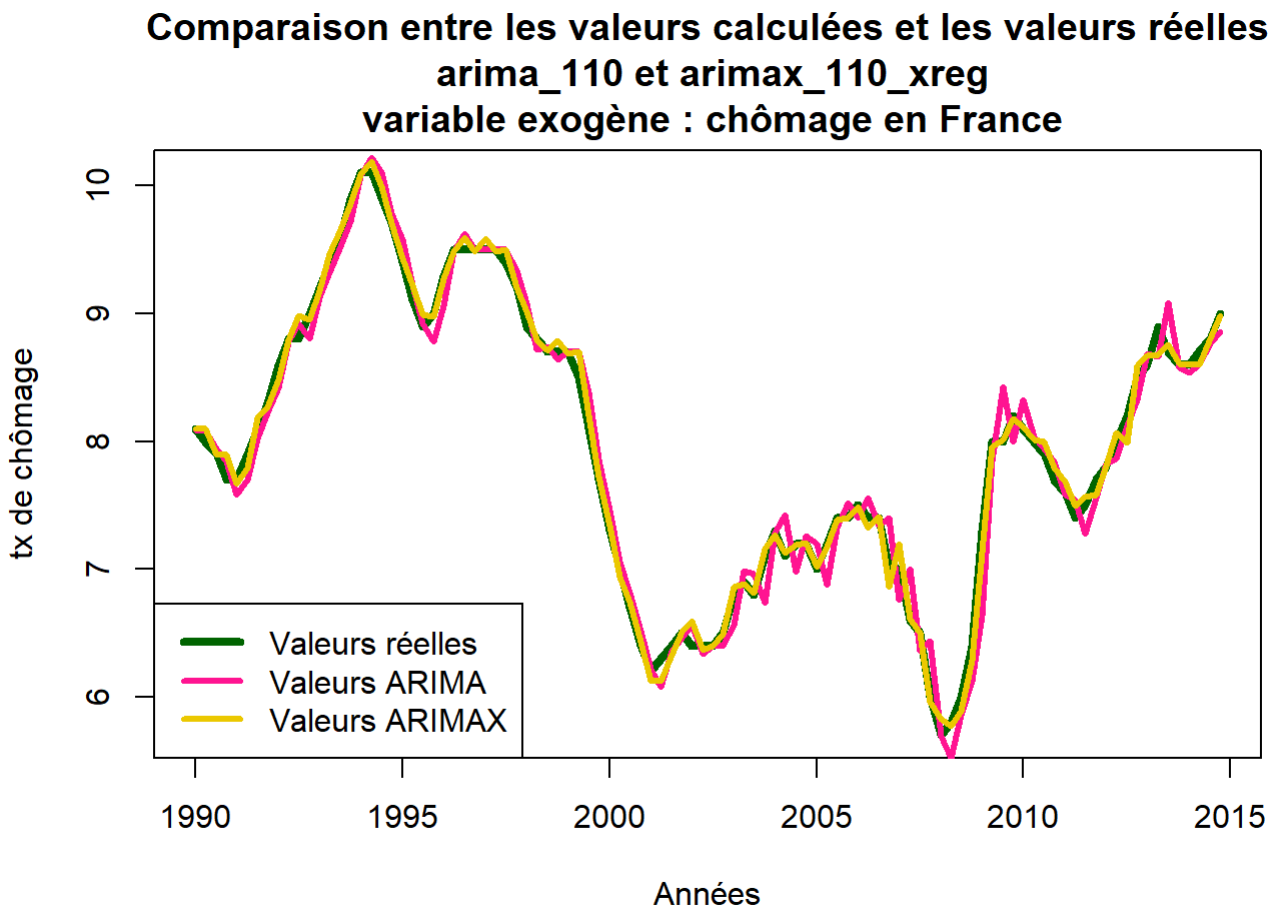
```
arimax_110_xreg <- Arima(d_train, order=c(1,1,0), xreg = vecf[1:100])
summary(arimax_110_xreg)
```

```
## Series: d_train
## Regression with ARIMA(1,1,0) errors
##
## Coefficients:
##          ar1    xreg
##       0.1539  0.8964
## s.e.  0.1188  0.0462
##
## sigma^2 = 0.007092:  log likelihood = 105.49
## AIC=-204.98   AICc=-204.73   BIC=-197.2
##
## Training set error measures:
##                       ME       RMSE        MAE        MPE      MAPE      MASE
## Training set -0.01053145 0.08293887 0.06048731 -0.1112879 0.7791547 0.1059632
##                      ACF1
## Training set -0.01825254
```

Remarque: ici nous ajoutons la variable exogène au modèle **arima_110** développé précédemment. Considérant que le modèle devait être repris du début, nous avons essayé d'autres modèles (non représenté ici) avec différents ordres p et q. Aucun ne fait mieux qu'**arima_110**, qui a aussi l'avantage de la parcimonie.

Notons que l'AICc de **arimax_110_xreg** est nettement meilleur que celui de **arima_110**, respectivement -204.73 et -50.84 (les autres indices de qualité vont dans le même sens). Notons aussi que le coefficient du régresseur exogène est nettement supérieur au coefficient d'autorégression. On peut déjà supposer que nos prédicitons seront de meilleure qualité.

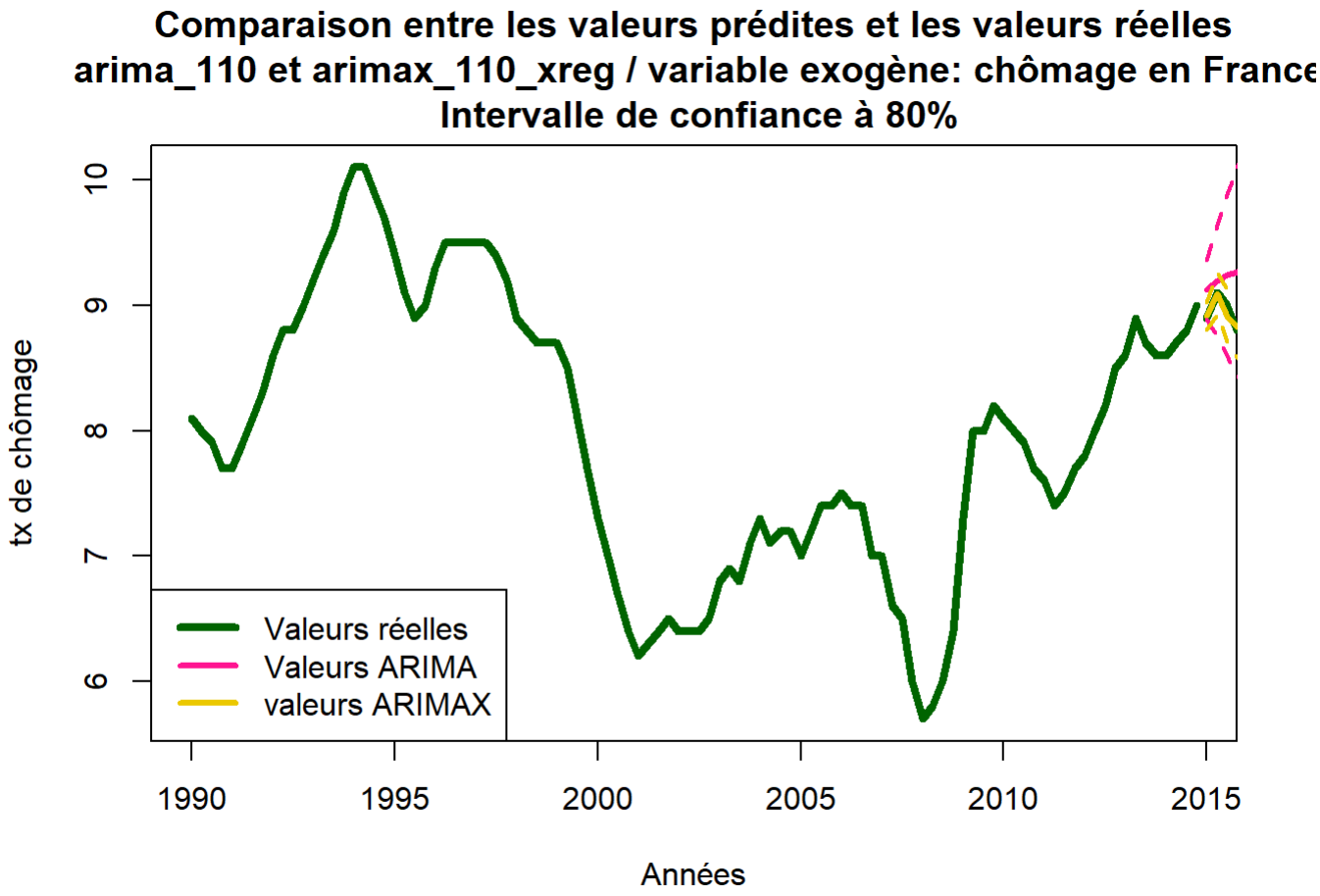
```
# Tracer les valeurs prédites et les valeurs réelles
plot(d_train, type = "l", col = "darkgreen",lwd= 4, main = "Comparaison entre les valeurs cal
culées et les valeurs réelles\n arima_110 et arimax_110_xreg\n variable exogène : chômage en
France", ylab="tx de chômage", xlab = "Années")
lines(fitted(arima_110), col = "deeppink",lwd= 3)
lines(fitted(arimax_110_xreg), col = "gold2",lwd= 3)
legend("bottomleft", legend = c("Valeurs réelles", "Valeurs ARIMA", "Valeurs ARIMAX"), col =
c("darkgreen", "deeppink", "gold2"), lwd= c(4,3,3))
```



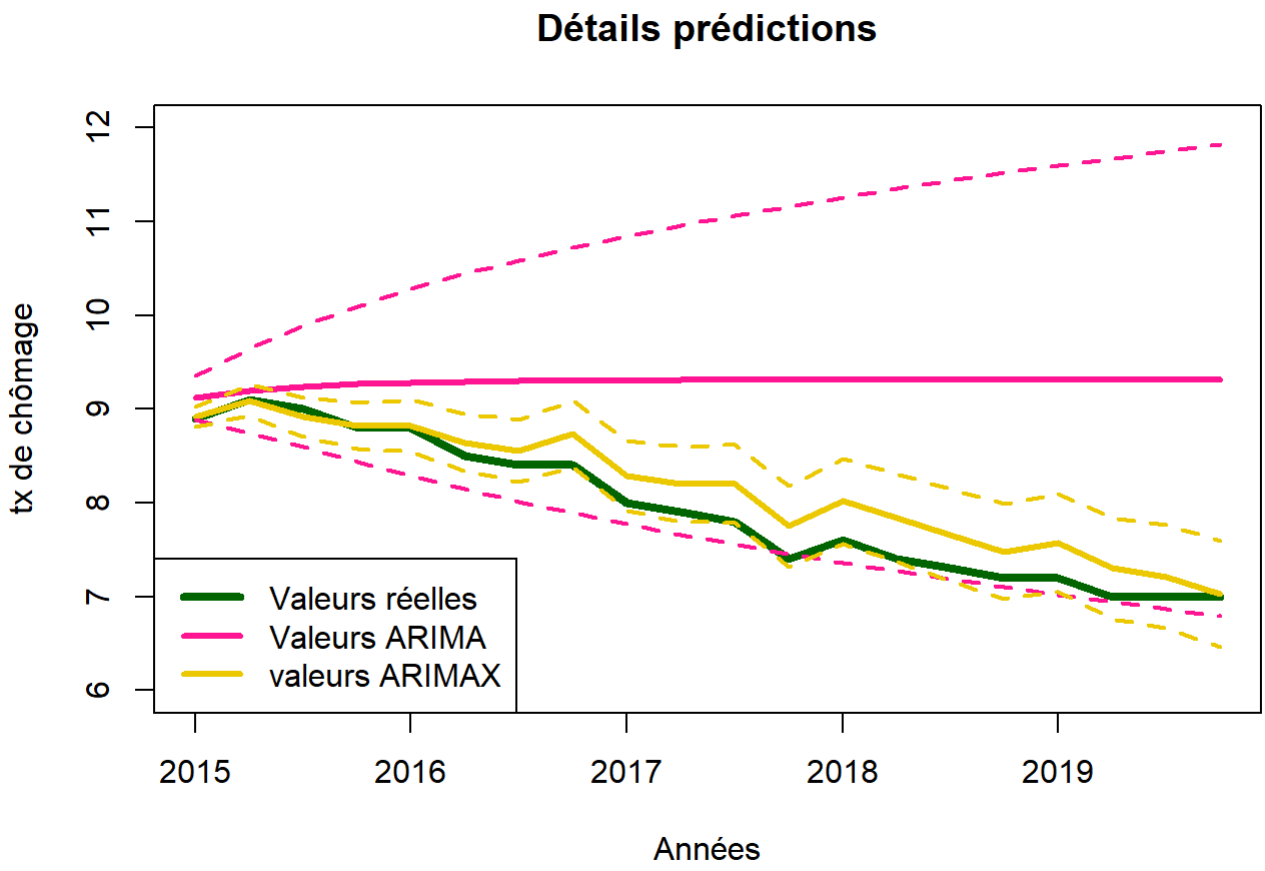
Où l'on voit que le modèle ARIMAX suit mieux les valeurs réelles. Il anticipe mieux les changements de tendances et les pics.

```
# Calculer les prédictions et les intervalles de confiance
forecast_result <- forecast(arima_110, h = 20, level = 80)
forecast_resultx <- forecast(arimax_110_xreg, xreg=(vecf[101:120])) # pas besoin de h , wreg=
fixe le nombre de prédictions
# Extraire les valeurs prédites et les intervalles de confiance inférieurs et supérieurs
mean <- forecast_result$mean
meanx <- forecast_resultx$mean
lower_ci <- forecast_result$lower[, "80%"]
upper_ci <- forecast_result$upper[, "80%"]
lower_cix <- forecast_resultx$lower[, "80%"]
upper_cix <- forecast_resultx$upper[, "80%"]

# Tracer les valeurs prédites et les valeurs réelles
plot(d_train, col= "darkgreen", lwd= 4, main = "Comparaison entre les valeurs prédites et les
valeurs réelles\n arima_110 et arimax_110_xreg / variable exogène: chômage en France\n Interv
alle de confiance à 80%", ylab="tx de chômage", xlab = "Années")
lines(d_test, , col= "darkgreen", lwd= 4)
lines(mean, col = "deeppink", lwd=3) # prédictions arima
lines(meanx, col = "gold2", lwd=3) # prédictions arimax
lines(lower_ci, col = "deeppink", lty = 2, lwd=2)  # Intervalle de confiance inférieur arima
lines(upper_ci, col = "deeppink", lty = 2, lwd=2)  # Intervalle de confiance supérieur arima
lines(lower_cix, col = "gold2", lty = 2, lwd=2)  # Intervalle de confiance inférieur arimax
lines(upper_cix, col = "gold2", lty = 2, lwd=2)  # Intervalle de confiance supérieur arimax
legend("bottomleft", legend = c("Valeurs réelles", "Valeurs ARIMA", "valeurs ARIMAX"), col =
c("darkgreen", "deeppink", "gold2"), lwd= c(4,3,3))
```




```
# Tracer les valeurs prédites avec intervalle de confiance
plot(d_test, type = "l", main = "Détails prédictions",ylim = c(6,12), col= "darkgreen", lwd=
4, ylab="tx de chômage", xlab = "Années")
lines(mean, col = "deeppink", lwd=3)
lines(meanx, col = "gold2", lwd=3)
lines(lower_ci, col = "deeppink", lty = 2, lwd=2)  # Intervalle de confiance inférieur
lines(upper_ci, col = "deeppink", lty = 2, lwd=2)  # Intervalle de confiance supérieur
lines(lower_cix, col = "gold2", lty = 2, lwd=2)  # Intervalle de confiance inférieur
lines(upper_cix, col = "gold2", lty = 2, lwd=2)  # Intervalle de confiance supérieur
legend("bottomleft", legend = c("Valeurs réelles", "Valeurs ARIMA", "valeurs ARIMAX"), col =
c("darkgreen", "deeppink", "gold2"), lwd= c(4,3,3))
```



La différence de qualité des prédictions est impressionnante. Remarque: le modèle ARIMA sans variable exogène ne démérite pas. L'évolution de la courbe du chômage est tout simplement "trop difficile" à prédire, trop erratique (on pourrait dire trop aléatoire). Ajoutons que les 5 années mises à l'écart pour notre jeu de test correspondent à une période d'inflexion assez brutale (à la baisse) du taux de chômage, et donc ne suivent pas l'évolution globalement croissante des trimestres précédents. Rappelons également que la courbe réelle se trouve bien dans l'intervalle de confiance de l'ARIMA classique (pour être précis, elle en sort ponctuellement au dernier trimestre 2017), mais ceci au prix d'un intervalle très large (ce qui signifie que le modèle prend bien en compte l'aspect erratique de la série).

Concernant le modèle ARIMAX, les prédictions sont très proches de la courbe réelle et cette dernière se situe toujours dans l'intervalle de confiance qui reste assez resserré sur les 5 années de prédiction, ce qui est sans doute le résultat le plus impressionnant.

Limite de notre approche ARIMAX: nous avons utilisé des valeurs réelles pour la variable exogène (la série test du chômage de l'ensemble de la France). Nous avons donc en quelque sorte triché, i.e.: une partie de l'avenir était connu! Evidemment, dans la plupart des cas, pour de vraies prédictions (autrement que pour des données test disponibles), nous ne pourrions pas agir de la sorte.

Commentaire sur la limite exposée ci-dessus: on pourrait choisir la variable exogène de manière à ce qu'elle soit déterministe. Ici dans le cas du chômage, ce pourrait être par exemple l'investissement public (une variable fixée par l'action politique). Ou bien, cas intermédiaire, la variable exogène pourrait être "peu aléatoire" (ici, ce pourrait être la démographie, qui évolue de manière déterministe à court terme).

Revenons à notre modèle. Le modèle ARIMAX peut-il, si on ne connait pas les valeurs de la variable exogène, faire mieux que l'ARIMA classique? Nous essaierons deux méthodes: la première, naïve, est de considérer la variable exogène comme une constante pour nos prédictions. Nous prendrons la dernière valeur connue de la période d'entraînement (soit ici 10.1%, le taux de chômage de la France au 4e trimestre 2014). La deuxième consiste à utiliser les prédictions d'un modèle de séries temporelles comme variables exogènes.

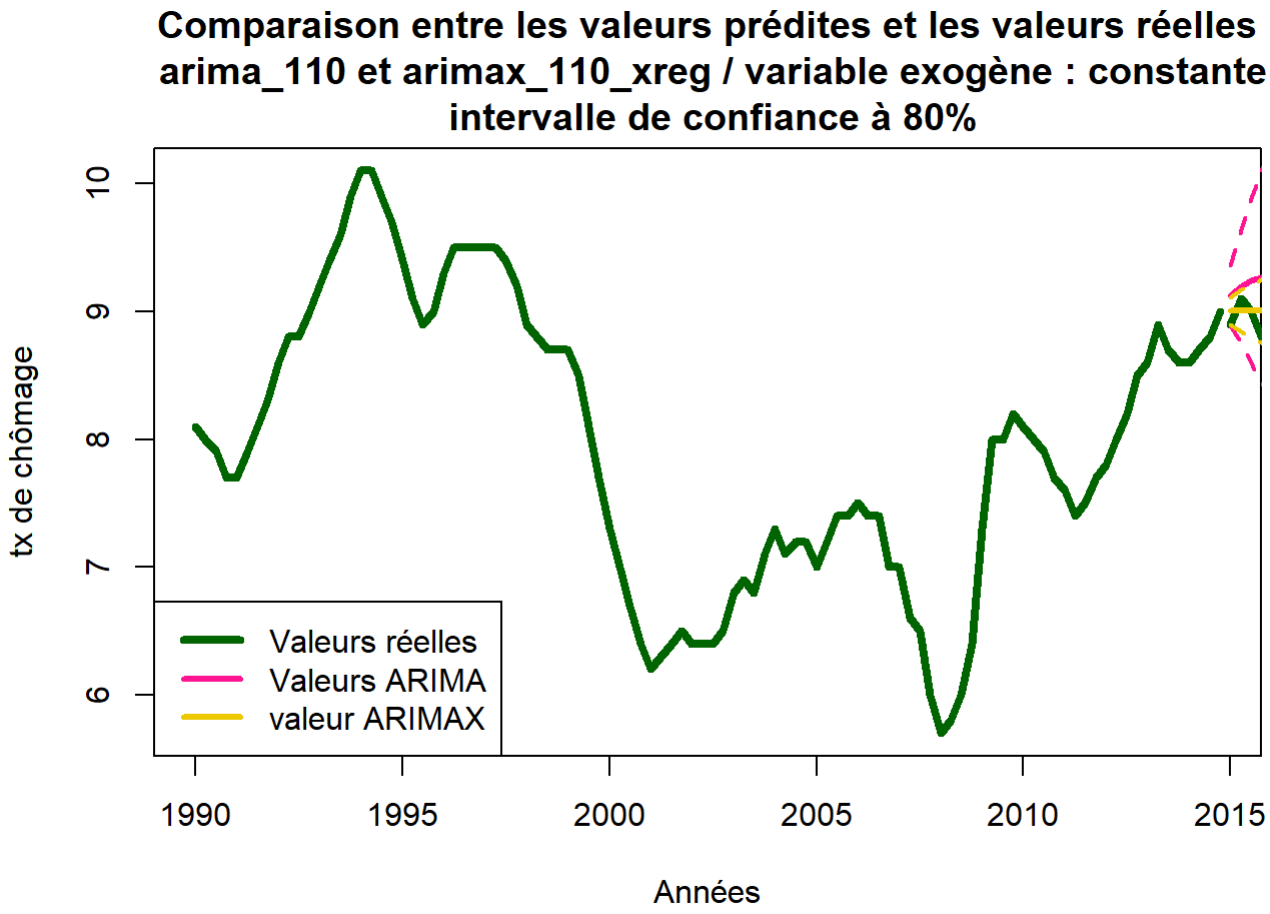
```
last_value <- tail(vecf_train, n = 1) # extraction de la valeur du dernier trimestre de la sé
rie d'entraînement de la France
cst <- rep(10.1, times = 20) # création vecteur de 20 valeurs à 10.1
```

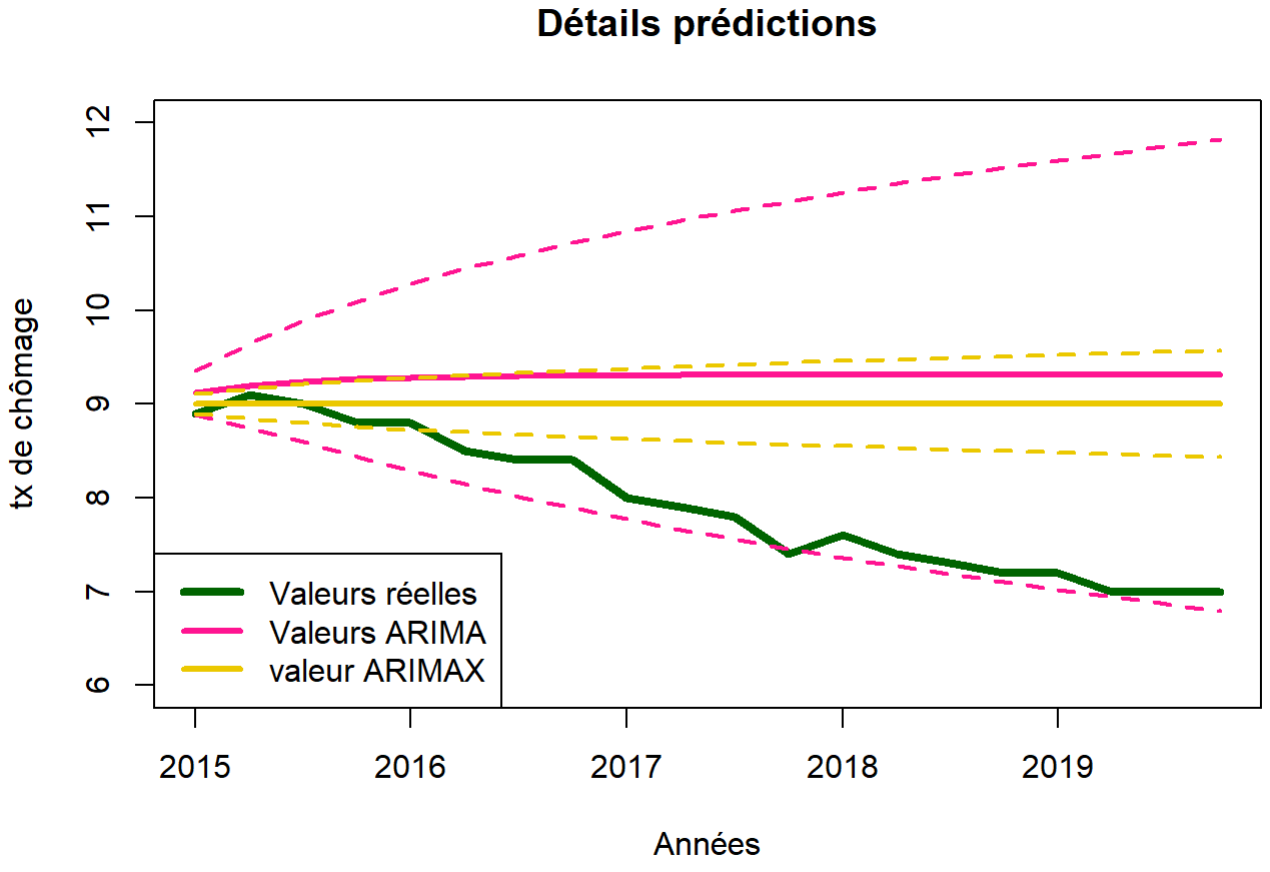
Afin de ne pas surcharger inutilement le rapport, nous n'affichons pas ici le code des graphiques suivants, car il est identique au précédent, à l'exception de la ligne suivante:

- *forecast_resultx <- forecast(arimax_110_xreg, xreg=(cst))*

qui remplace

- *forecast_resultx <- forecast(arimax_110_xreg, xreg=(vecf[101:120]))*





Avec une variable exogène constante, l'ARIMAX fait à peine mieux que l'ARIMA classique, et surtout, son intervalle de confiance est inexploitable. En quelque sorte, la constante brouille les cartes, son coefficient étant prédominant dans le modèle, les prédictions sont elles-mêmes proches d'une constante. La raison d'une telle

différence dans les intervalles de confiance des deux modèles pourrez se résumer ainsi: ARIMA ne sait pas, alors qu'ARIMAX croit savoir.

L'autre solution proposée consiste à créer un modèle pour la série "taux de chômage en France" et à utiliser les prédictions comme variables exogènes de notre ARIMAX du taux de chômage des Pays de la Loire. Nous ne détaillons pas l'élaboration du modèle lui-même, nous aboutissons à un modèle AR(1) (**arima_fr_110**), identique dans sa structure au modèle utilisé pour les Pays de la Loire (seule la valeur des coefficients change).

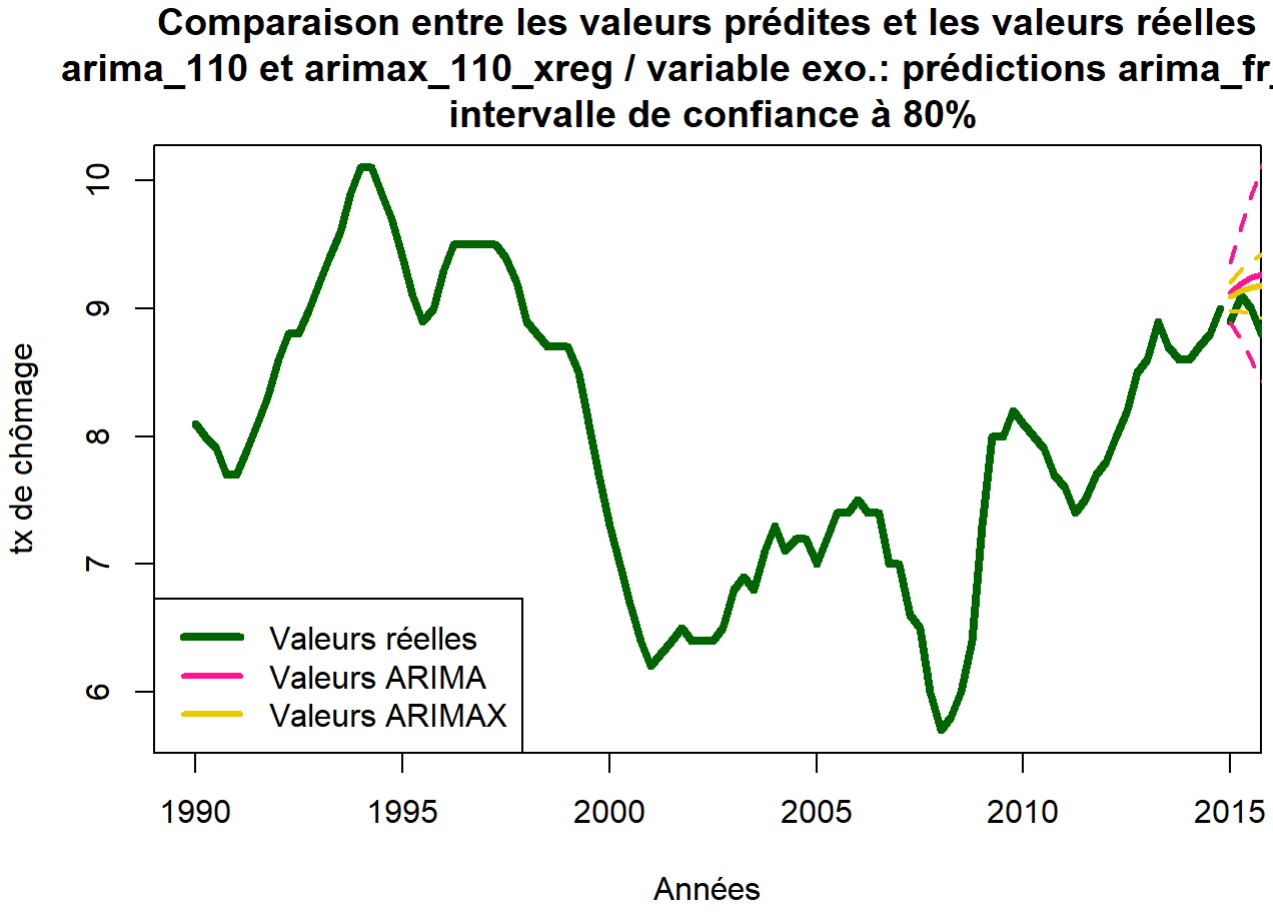
```
arima_fr_110<- Arima(df_train,order=c(1,1,0)) # modèle ARIMA sur la série entrainement France entière
summary (arima_fr_110)
```

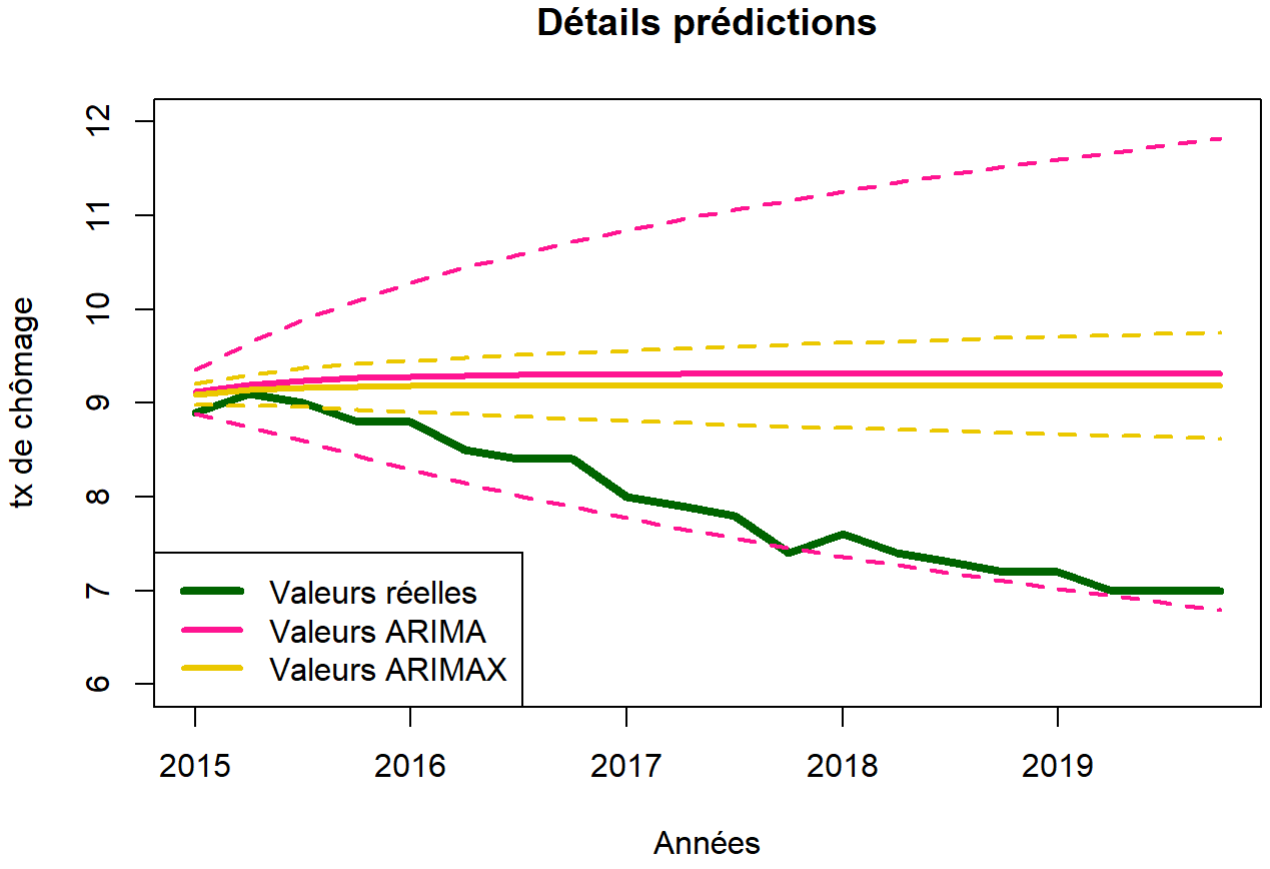
```
## Series: df_train
## ARIMA(1,1,0)
##
## Coefficients:
##          ar1
##       0.5061
## s.e.  0.0860
##
## sigma^2 = 0.04051:  log likelihood = 18.59
## AIC=-33.19   AICc=-33.06   BIC=-28
##
## Training set error measures:
##                       ME      RMSE       MAE       MPE     MAPE     MASE
## Training set 0.01294166 0.1992366 0.1441262 0.1451218 1.683553 0.255279
##                     ACF1
## Training set -0.1151032
```

Calculons les prédictions:

```
# Calculer les prédictions et les intervalles de confiance
forecast_result_fr <- forecast(arima_fr_110, h = 20, level = 80)
mean_fr <- forecast_result_fr$mean # valeurs prédites moyennes ARIMA série France
lower_ci_fr <- forecast_result_fr$lower[, "80%"] # valeurs prédites IC 80% inférieur ARIMA série France
upper_ci_fr <- forecast_result_fr$upper[, "80%"] # valeurs prédites IC 80% supérieur ARIMA série France
```

Nous utilisons les prédictions d'**arima_fr_110** comme variable exogène, ce qui donne:





Le modèle utilisant des prédictions comme variable exogène ne fait pas mieux que celui avec valeur constante.

Non représenté ici: nous avons essayé de pousser cette idée en utilisant les valeurs des intervalles de confiance de l'ARIMA sur la série d'entraînement du chômage en France calculées plus haut (ici) comme variable exogène, afin d'encadrer plus largement nos prédictions. Par exemple, l'intervalle inférieur de nos prédictions ARIMAX devient l'intervalle inférieur de l'ARIMAX qui a comme variable exogène l'intervalle inférieur des prédictions de l'ARIMA sur les valeurs du chômage de la France entière. Et vice-versa pour l'intervalle supérieur.

Le résultat est décevant: la courbe moyenne des prédictions reste bien sûr inchangée, mais l'intervalle de confiance "total" est encore plus large que celui du modèle ARIMA classique sans variable exogène (comme explication, nous pouvons avancer que la même incertitude est en quelque sorte comptée plusieurs fois). Extrait du code:

- forecast_resultx <- forecast(arimax_110_xreg, xreg=(mean_fr), level = 80) # Inchangé
- forecast_resultx_u <- forecast(arimax_110_xreg, xreg=(upper_ci_fr), level = 80) # encadre notre moyenne par le haut
- forecast_resultx_l <- forecast(arimax_110_xreg, xreg=(lower_ci_fr), level = 80) # encadre notre moyenne par le bas

Approche bayésienne

Nous allons utiliser le package **bayesforecast** pour recalculer notre modèle ARIMAX.

Nous utilisons la fonction *stan_sarima()* du package. Nous ne précisons pas les lois des **priors** pour l'estimation des paramètres. Par défaut, la fonction utilise des priors *"non informative"*. Nous définissons le nombre d'itérations MCMC à 3000 (la période de **warmup** est définie par défaut comme le nombre d'itérations divisé par 2).

Nous devons transformer nos données en matrice pour la variable exogène de la fonction *stan_sarima()*.

Calcul du modèle **arimax_110_bayes**:

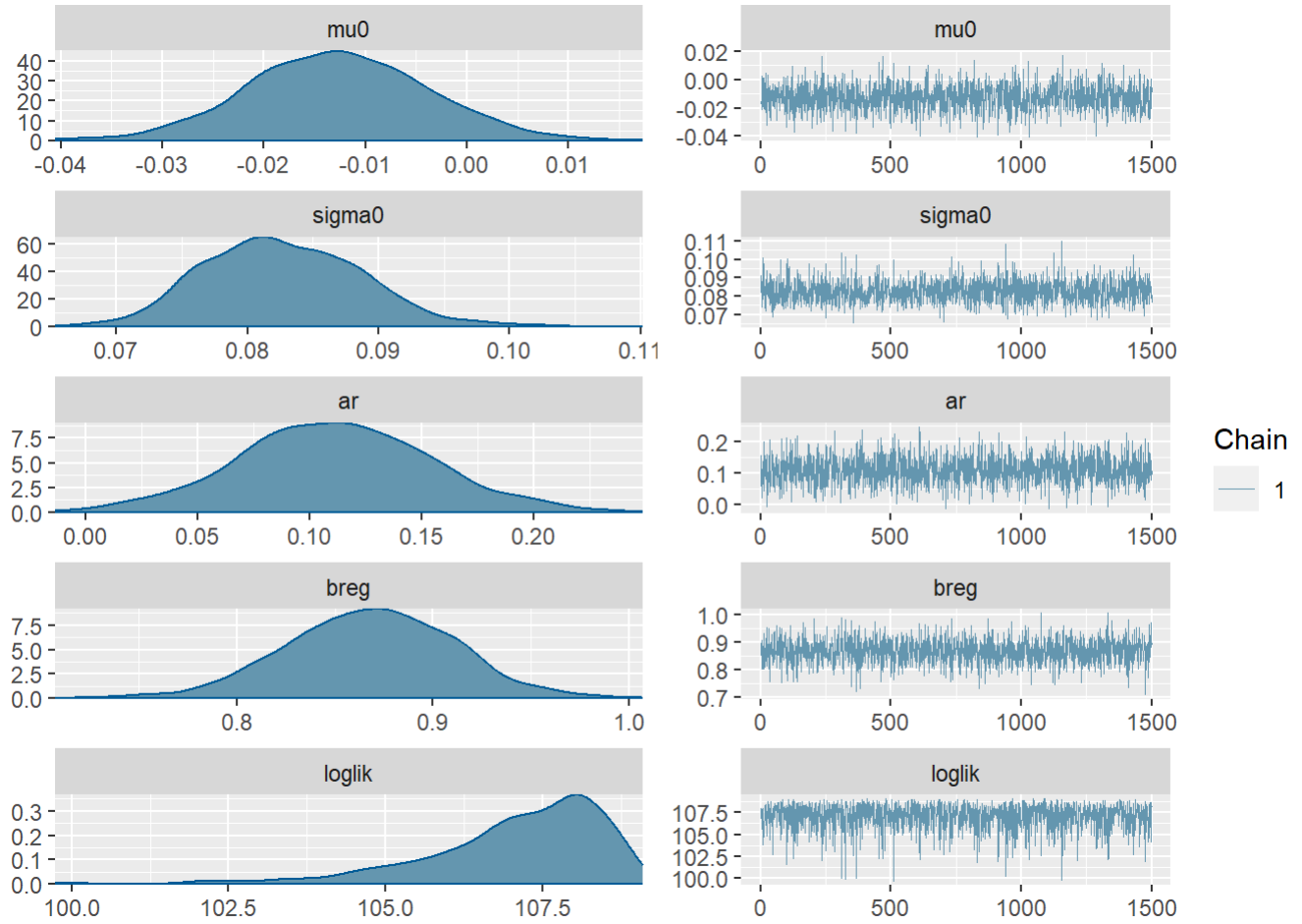
```
mat_train <- matrix(vecf[1:100], nrow = 100, byrow = TRUE)  # création matrice nécessaire pou
r fonction stan_sarima() série entraïnement
mat_test <- matrix(vecf[101:120], nrow = 20, byrow = TRUE) # création matrice nécessaire pour
fonction stan_sarima() série test
arimax_110_bayes = stan_sarima(ts = d_train,order = c(1,1,0),seasonal = c(0,0,0),iter=3000,ch
ains = 1, xreg = mat_train)
```

```
summary(arimax_110_bayes)
```

```
##            mean     se       5%      95%      ess   Rhat
## mu0     -0.0127 0.0002  -0.0272   0.0025 1503.011 1.0018
## sigma0   0.0827 0.0002   0.0737   0.0926 1559.970 1.0051
## ar       0.1112 0.0011   0.0396   0.1848 1554.270 0.9999
## breg     0.8678 0.0011   0.7970   0.9348 1599.150 0.9998
## loglik 107.0570 0.0377 104.3115 108.7343 1580.669 1.0011
```

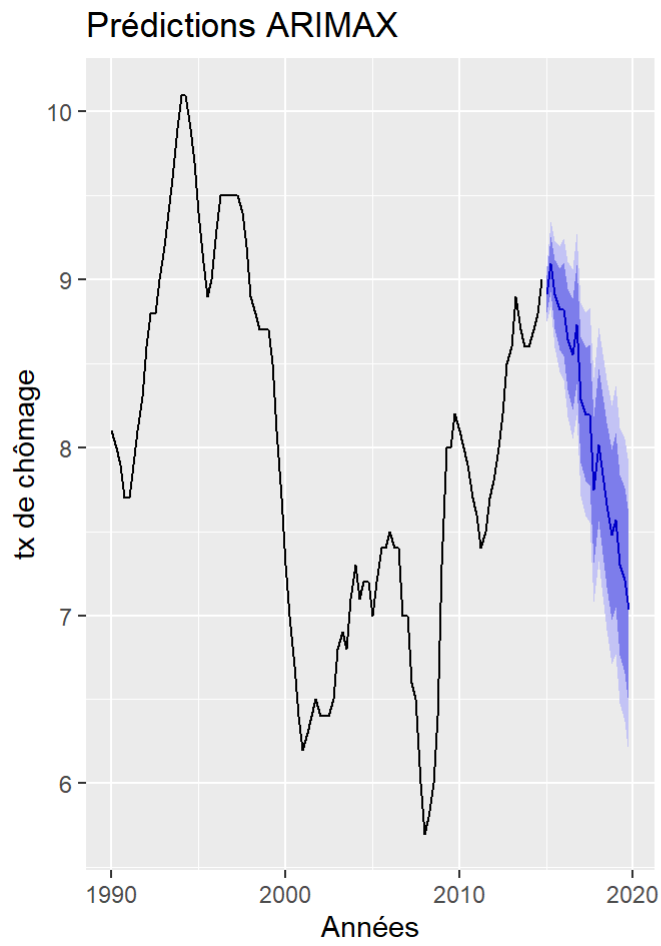
Nous constatons que les coefficients sont très proches de ceux calculés par l'approche non bayésienne (ici). Le package **bayesforecast** permet d'afficher la distribution de chaque paramètre et non plus seulement une valeur brute:

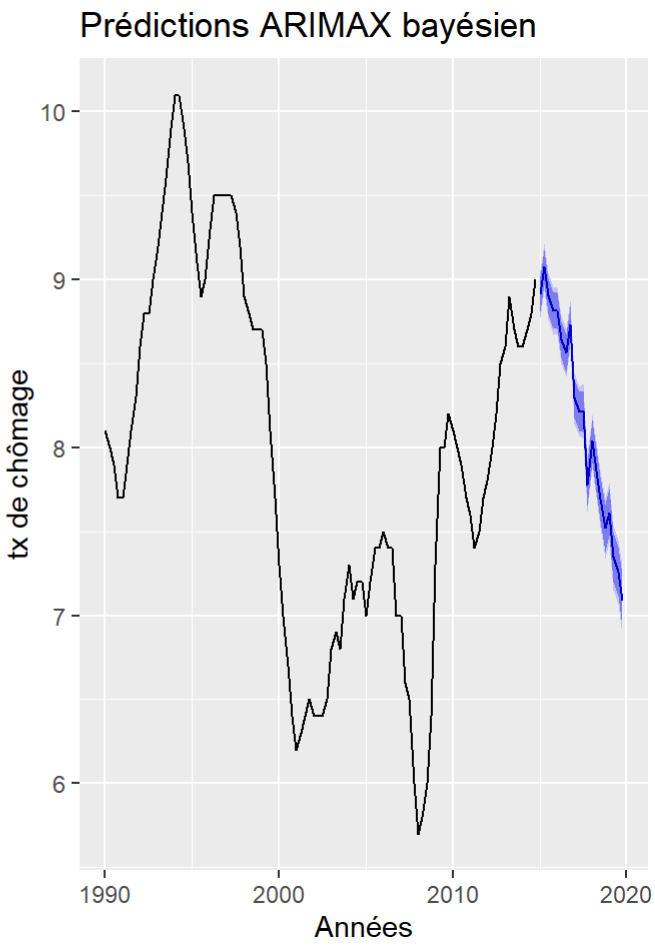
```
mcmc_plot(object = arimax_110_bayes)
```



Comparaison des prédictions ARIMAX classique et ARIMAX bayésien:

```
par(mfrow = c(1, 2))
p1 <- autoplot(object = forecast(arimax_110_xreg,h = 20, xreg=mat_test), main = "Prédictions
ARIMAX", xlab= "Années", ylab="tx de chômage")
p2 <- autoplot(object = forecast(arimax_110_bayes,h = 20, xreg=mat_test),main = "Prédictions
ARIMAX bayésien", xlab= "Années", ylab="tx de chômage")
grid.arrange(p1, p2, ncol = 2)
```





Deux remarques:

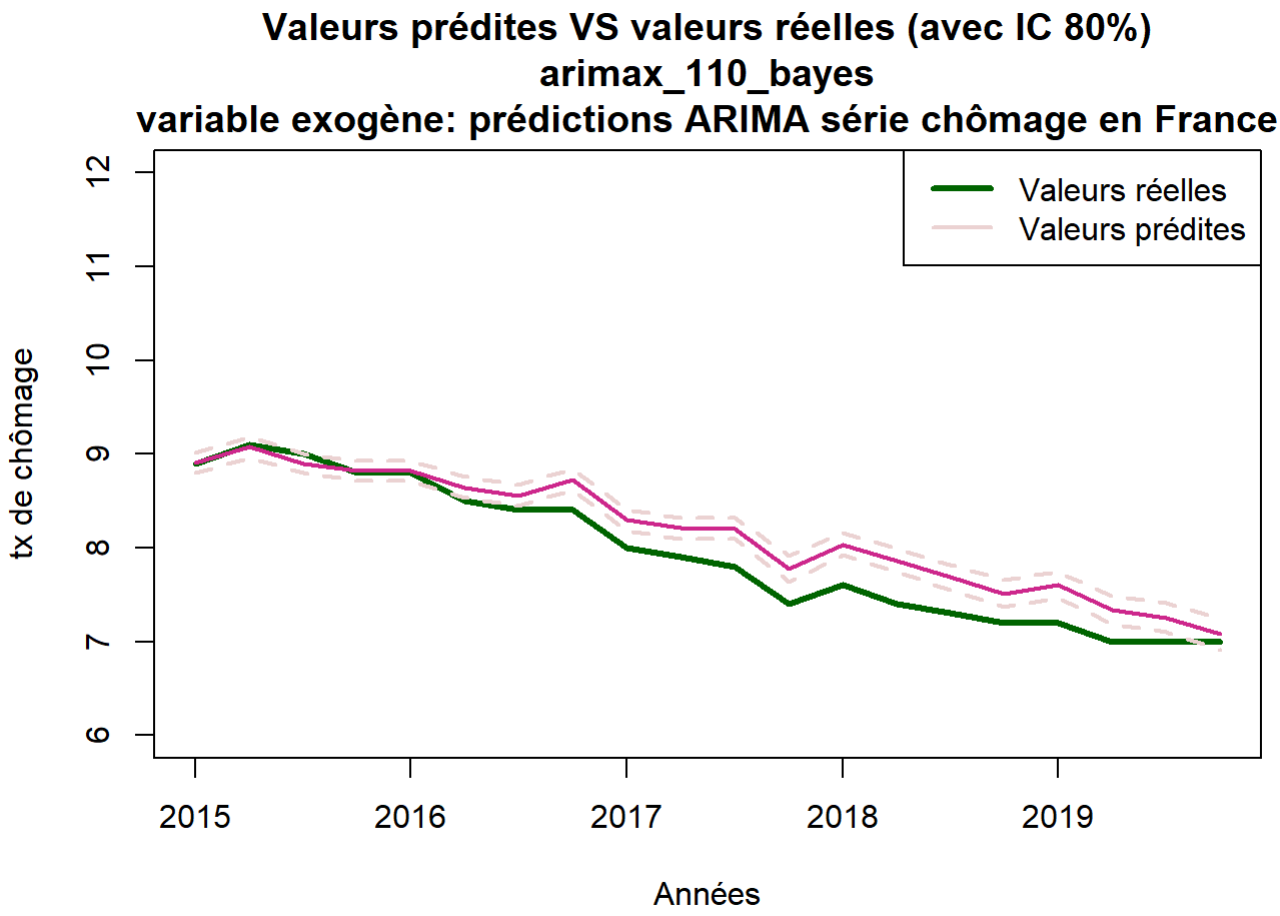
- Les valeurs moyennes des prédictions sont similaires (sans surprise, les coefficients étant très proches pour les deux modèles);
- L'intervalle de confiance de l'ARIMAX bayésien est nettement plus resserré que celui de l'ARIMAX classique. (intervalle de confiance à 80% en bleu foncé et 90% en bleu clair.)

Comparons les prédictions et les valeurs réelles:

```
# Calculer les prédictions et les intervalles de confiance
forecast_bayes <- forecast(arimax_110_bayes,h = 20, xreg=mat_test)
# Extraire les valeurs prédites et les intervalles de confiance inférieurs et supérieurs
mean <- forecast_bayes$mean
lower_ci <- forecast_bayes$lower[, "80%"]
upper_ci <- forecast_bayes$upper[, "80%"]

# Tracer les valeurs prédites avec l'intervalle de confiance
plot(d_test, type = "l", main = "Valeurs prédites VS valeurs réelles (avec IC 80%)\n arimax_1
10_bayes \nvariable exogène: prédictions ARIMA série chômage en France", ylim = c(6, 12), col
= "darkgreen", lwd=3, ylab="tx de chômage", xlab = "Années")
lines(mean, col = "maroon3", lwd=2)
lines(lower_ci, col = "mistyrose2", lty = 2, lwd=2)  # Intervalle de confiance inférieur
lines(upper_ci, col = "mistyrose2", lty = 2, lwd=2)  # Intervalle de confiance supérieur

legend("topright", legend = c("Valeurs réelles", "Valeurs prédites"), col = c("darkgreen", "m
istyrose2"), lwd= c(3, 2))
```



On peut constater que les valeurs réelles sont la plupart du temps en dehors de l'intervalle de confiance à 80%.

Modèle espace-état bayésien

En utilisant le package **BSTS**, nous allons proposer un modèle espace-état, en utilisant la série du chômage en France comme variable exogène.

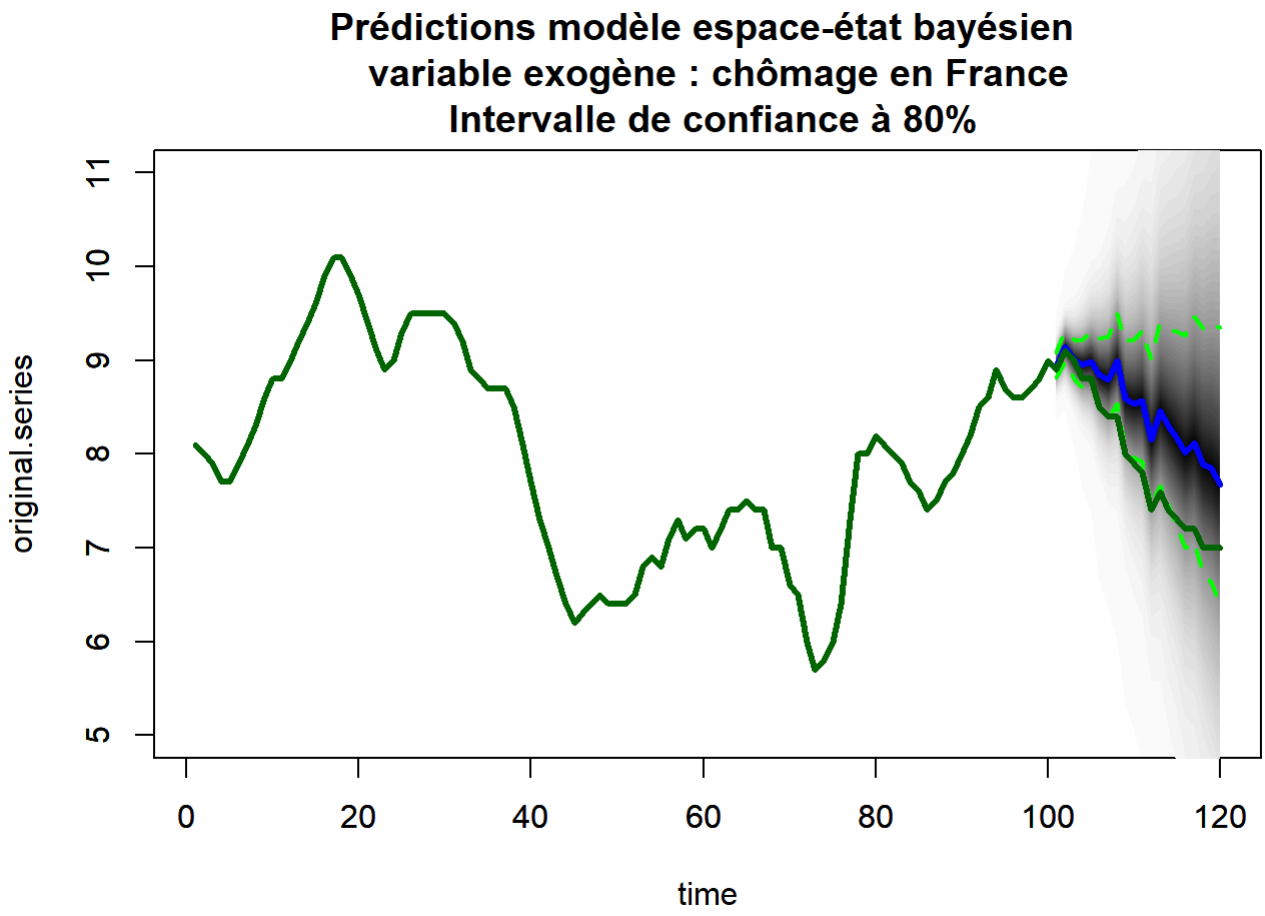
Nous laissons là aussi la fonction *bsts()* utiliser les **priors** par défaut.

```
state_space_bayes <- AddLocalLinearTrend(list(), d_train)
bsts.model <- bsts(d_train ~ df_train, state.specification = state_space_bayes, niter = 6000)
# 6000 itérations, vecf[1:100] comme variable exogène
```

Affichons les prédictions de ce modèle avec l'intervalle de confiance à 80% en utilisant la série **df_test** comme variable exogène.

```
pred <- predict.bsts(bsts.model, newdata=df_test, horizon = 20, burn = 1000)
plot (pred, ylim = c(5, 11),main="Prédictions modèle espace-état bayésien \n  variable exogèn
e : chômage en France\n Intervalle de confiance à 80%",interval.quantiles = c(.2, .8))
zeros <- rep(0, 100)

# Ajouter les 100 valeurs de 0 au début du vecteur d_test_vector
d_test_vector_extended <- c(as.vector(d_train), as.vector(d_test))
lines (d_test_vector_extended, col="darkgreen", lwd=3)
```



Les valeurs réelles se confondent presque avec la courbe inférieure de l'intervalle de confiance à 80%. La partie en dégradé de gris représente la densité de la distribution du *posterior*. C'est l'un des atouts de la méthode bayésienne que d'ajouter cette information aux simples valeurs moyennes et aux intervalles de confiance.

Non représenté ici: nous avons essayé un modèle selon cette méthode mais sans variable exogène, il tend à poursuivre la croissance du chômage des derniers trimestres de la série d'entraînement.

Conclusion

Nous avons essayé plusieurs approches et aucune n'est capable de prédire précisément l'évolution du chômage sur les cinq années suivantes, du moins sans l'ajout d'une variable exogène. Ce résultat n'est pas surprenant, car d'une certaine manière, l'information contenue dans la série d'entraînement n'est pas suffisante: la variable à expliquer est trop "imprévisible". Néanmoins, nous arrivons assez facilement à encadrer les données à prédire dans nos intervalles de confiance.

Nous avons constaté l'intérêt d'intégrer une variable exogène. Nous pourrions envisager de multiplier les variables prises en compte. Toutes les méthodes utilisées ici le permettent. Dans la réalité, le problème serait de choisir quelle variable utiliser (aux États-Unis, un chercheur a montré que l'une des variables les plus utiles pour prédire le taux de chômage n'était pas de nature économique mais correspondait au nombre d'occurences de certains mots clés dans les recherches Google…). Notons que l'utilisation de plusieurs régresseurs exogènes ouvre vers la prédiction multivariée. Dans ce cas, la sortie est un vecteur de plusieurs variables en relation entre elles.

L'approche bayésienne appliquée aux séries temporelles est un outil très puissant. Nous n'avons fait qu'effleurer ici les possibilités offertes par les fonctions des packages retenus. Cette approche peut à minima faire aussi bien que l'approche classique (d'une certaine manière, on peut dire qu'elle contient en elle-même l'approche classique, manipulant des fonctions de lois statistiques plutôt que de simples paramètres). Les

avantages de cette méthode sont la possibilité de quantifier l’incertitude, la flexibilité dans la spécification des modèles et une gestion puissante des variables exogènes. Cependant, elle peut nécessiter des calculs computationnels très coûteux, voire rédhibitoires (chose observée ici lors d’essais de différents modèles).