Taller 4 - Diagnóstico

Series Cronológicas 2024

Mayo 2024

1. Exploración de los datos

```
# Datos trimestrales (32 observaciones)
frequency(comercio)
```

```
## [1] 4
```

```
length(comercio)
```

```
## [1] 32
```

```
# Máximo valor observado y fecha en la que ocurrió
max(comercio)
```

```
## [1] 63814.26
```

```
time(comercio)[which.max(comercio)]
```

```
## [1] 2022.25
```

```
# Mínimo valor observado y fecha en la que ocurrió
min(comercio)
```

```
## [1] 45673.51
```

```
time(comercio)[which.min(comercio)]
```

```
## [1] 2020.25
```

```
# Graficamos la serie PIB Comercio
autoplot(comercio) +
  labs(x = "Fecha",
       y = "PIB Comercio") +
  scale_x_continuous(breaks = 2016:2024) +
  theme(panel.grid.minor = element_blank())
```

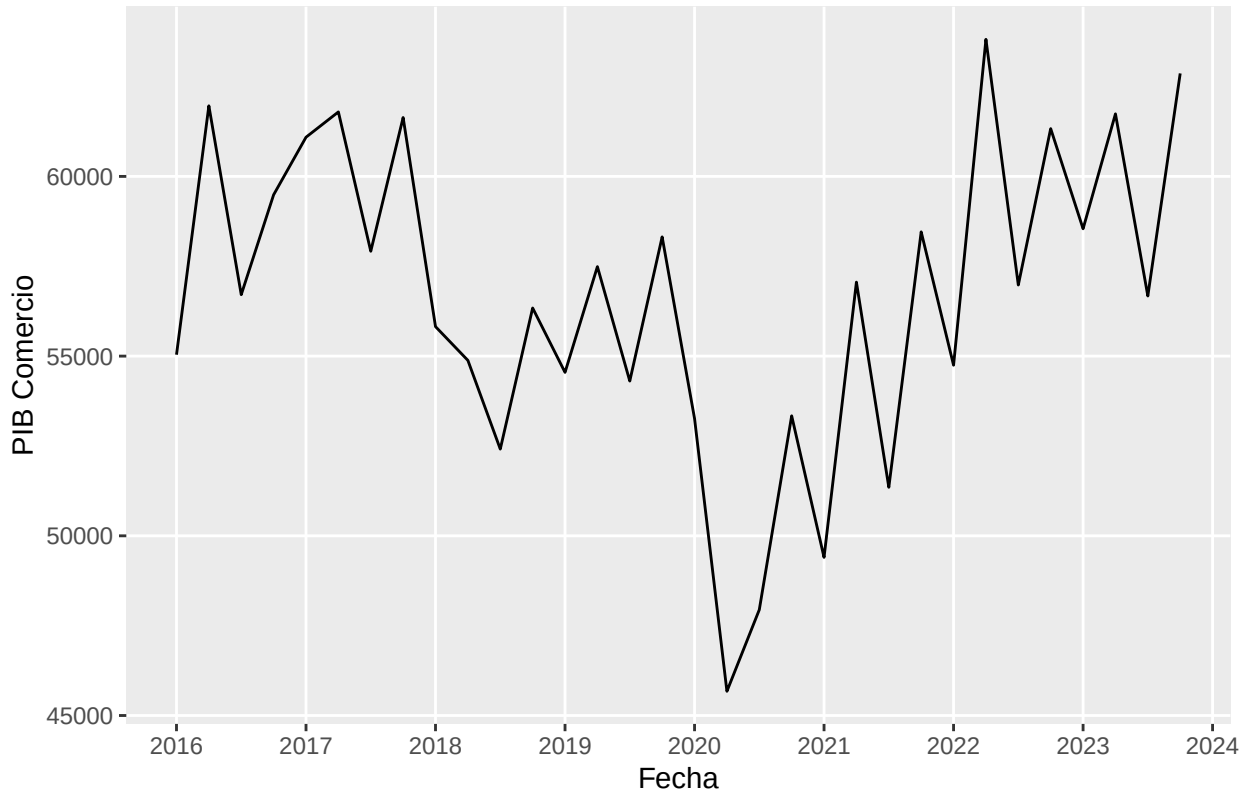


Figura 1: Evolución del PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023.

2. Identificación y estimación del modelo

```
# FAC (no descartamos estacionariedad)
comercio_acf <- ggAcf(comercio, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
comercio_pacf <- ggAcf(comercio, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(comercio_acf, comercio_pacf)
```

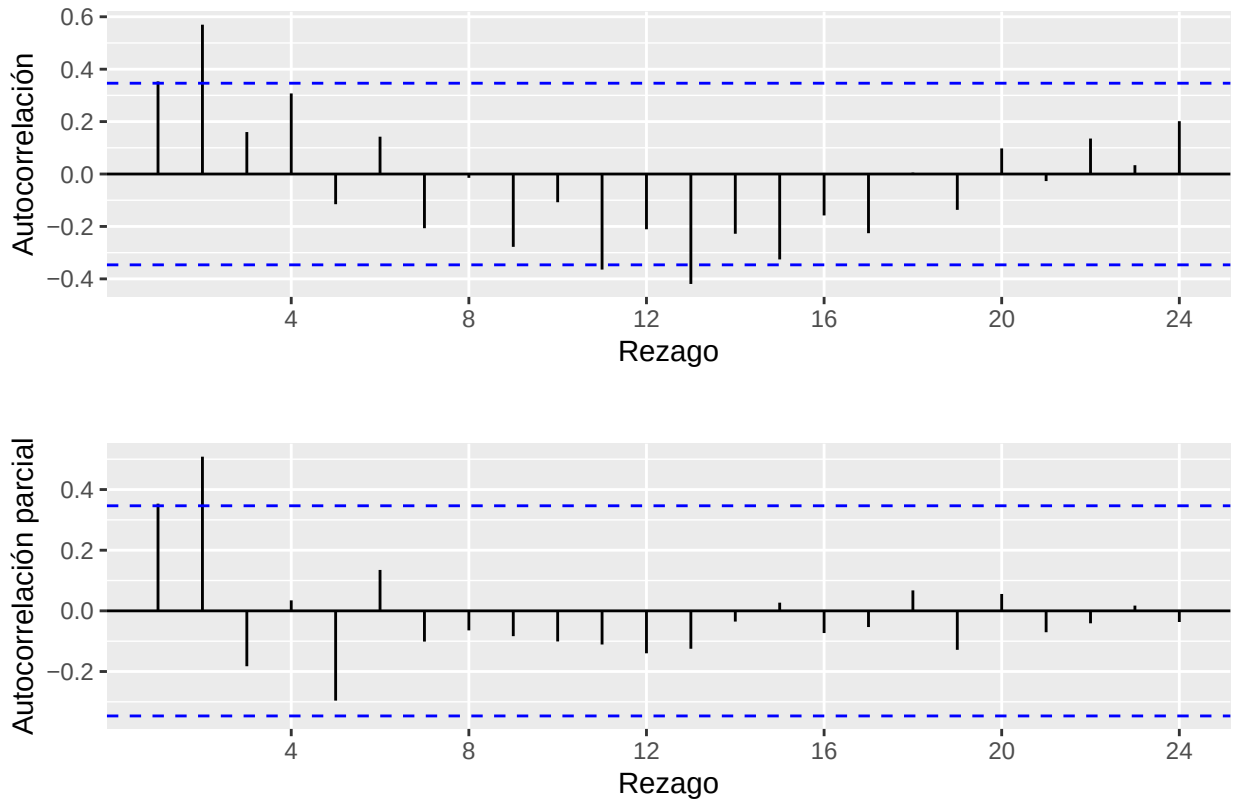


Figura 2: Funciones de Autocorrelación y Autocorrelación Parcial estimadas del PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023.

```
# Dada la forma de la FAC y la FACP, un posible modelo es un AR(2).

# AR(2)
modelo1 <- Arima(y = comercio, # Datos para estimar
                 order = c(2, 0, 0), # Orden del modelo (suponemos estacionariedad)
                 lambda = NULL) # Trabajamos con la serie sin transformar

coeftest(modelo1)
```

```
##
## z test of coefficients:
##
##              Estimate Std. Error z value  Pr(>|z|)
## ar1       1.4814e-01 1.4875e-01  0.9959 0.3193194
## ar2       5.4577e-01 1.5178e-01  3.5958 0.0003234 ***
## intercept 5.7151e+04 1.7677e+03 32.3306 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Probamos un AR(2) con phi1 = 0
modelo1 <- Arima(y = comercio,
                 order = c(2, 0, 0),
                 lambda = NULL,
                 fixed = c(0, NA, NA))

summary(modelo1)
```

```
## Series: comercio
## ARIMA(2,0,0) with non-zero mean
##
## Coefficients:
##       ar1     ar2       mean
##         0  0.6047  57052.412
## s.e.    0  0.1396   1418.146
##
## sigma^2 = 12537510:  log likelihood = -306.34
## AIC=618.67   AICc=619.53   BIC=623.07
##
## Training set error measures:
##                     ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -73.20484 3428.398 2497.484 -0.5355904 4.582298 0.5984372
##                    ACF1
## Training set 0.2758011
```

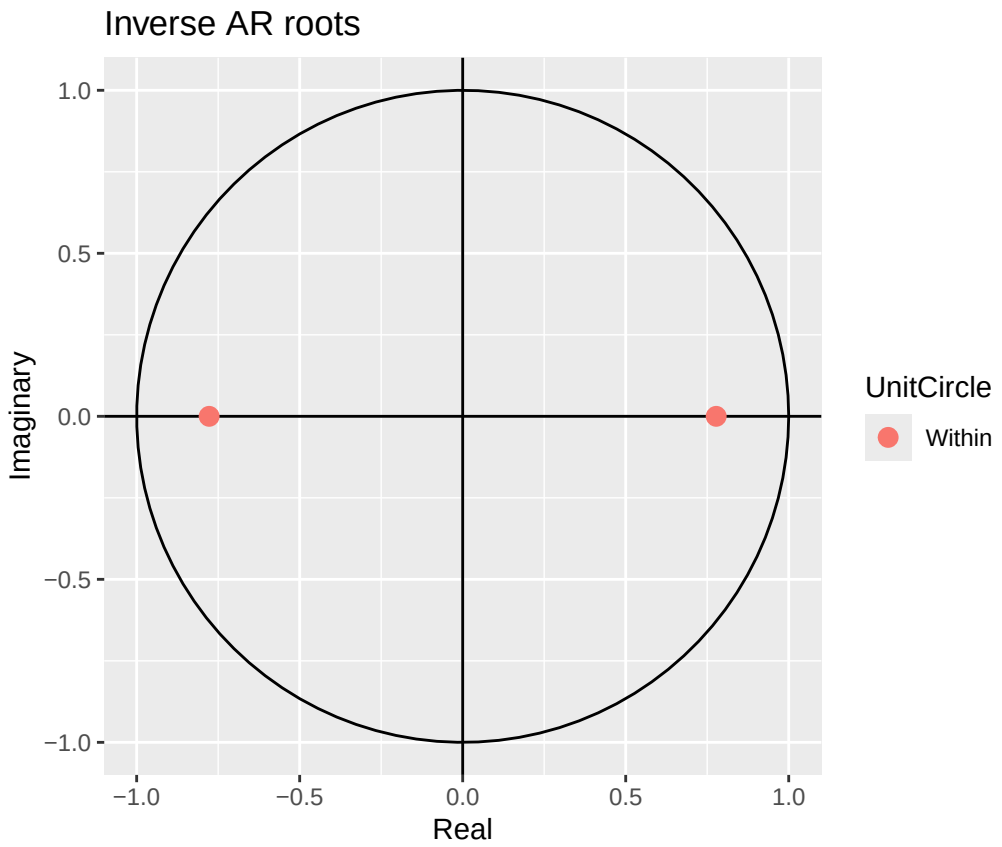
```r
coeftest(modelo1)
```

```
##
## z test of coefficients:
##
##             Estimate Std. Error z value  Pr(>|z|)
## ar2       6.0467e-01 1.3960e-01  4.3315 1.481e-05 ***
## intercept 5.7052e+04 1.4181e+03 40.2303 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
coefci(modelo1)
```

```
##                   2.5 %       97.5 %
## ar2       3.310641e-01 8.782737e-01
## intercept 5.427290e+04 5.983193e+04
```

```r
autoplot(modelo1)
```




```
# Nos quedamos con un modelo sin phi1
```

```
# Graficamos la serie y los valores ajustados

fit1 <- autoplot(comercio) +
  autolayer(modelo1$fitted, color = "blue") +
  labs(x = "Fecha",
       y = "PIB Comercio",
       title = "Modelo AR(2)")

fit1
```

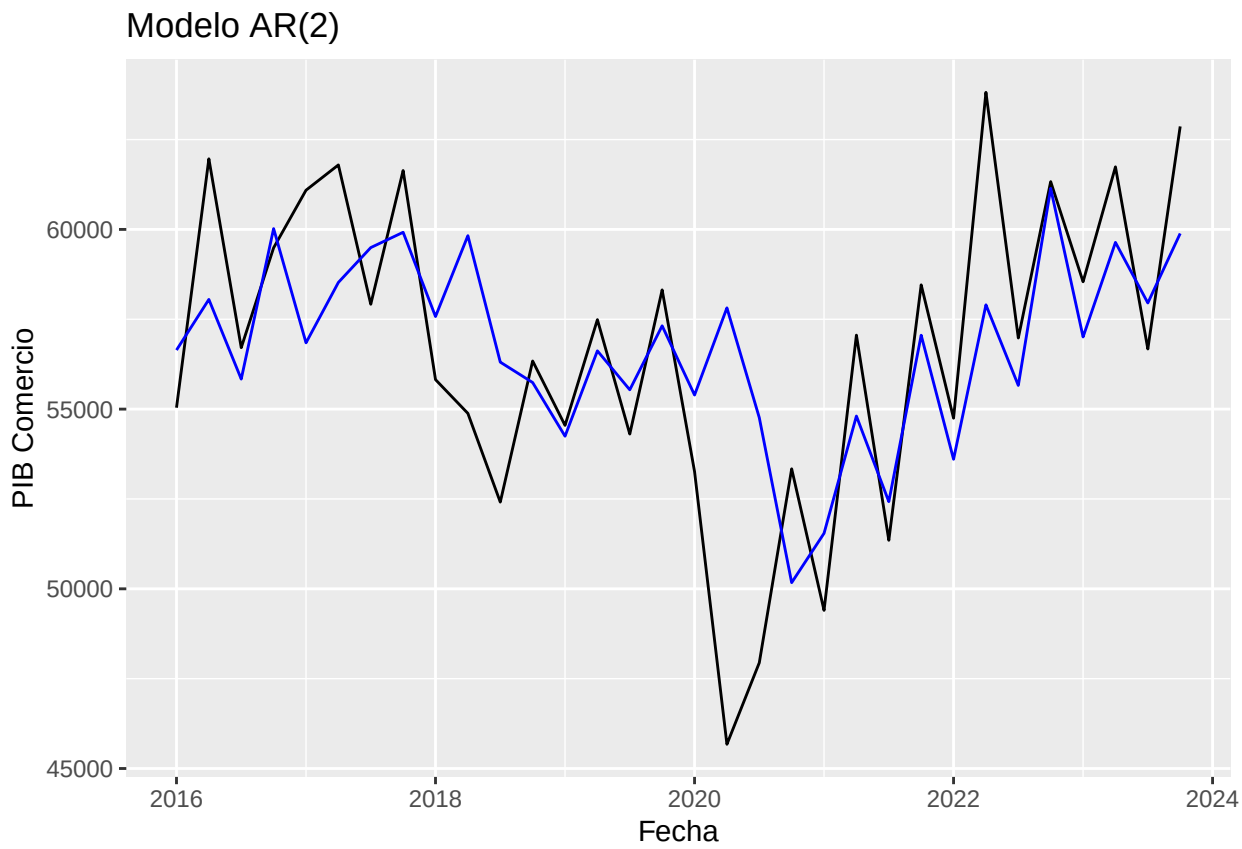


Figura 3: PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023 y valores ajustados para un modelo AR(2). La línea negra corresponde a los valores reales y la azul a los ajustados.

3. Selección de modelos

3.1. AIC

```
# Armamos una matriz con los valores del AIC para valores del 1 al 6 de p y q

norder <- 6 # Máximo orden de rezagos que queremos considerar
aic <- matrix(0, norder + 1, norder + 1)
rownames(aic) <- c(0:norder)
colnames(aic) <- c(0:norder)

for (i in 1:(norder + 1)) {
  for (j in 1:(norder + 1)) {
    modeloij <- Arima(comercio, order = c(i-1, 0, j-1), method = 'ML')
    aic[i,j] <- modeloij$aic
  }
}

aic
```

```
##          0        1        2        3        4        5        6
## 0 630.4252 630.3502 623.2718 624.4670 623.2849 624.7334 625.1860
## 1 627.9577 626.0650 622.9775 624.7430 620.3911 621.0990 623.0796
## 2 619.6811 616.6670 617.9708 619.5529 621.0074 622.9918 624.9242
## 3 620.1737 617.8167 618.9233 614.6685 616.2613 618.1995 620.1974
## 4 622.0031 619.8150 618.6779 623.0360 624.9592 627.0002 628.9016
## 5 621.3374 621.0535 619.7162 618.2230 620.1506 622.2300 623.5399
## 6 622.4286 622.8422 621.7083 623.7123 625.1479 630.4621 632.4785
```

```
min(aic)
```

```
## [1] 614.6685
```

```
which(aic == min(aic), arr.ind = TRUE) # El AIC mínimo se da para un ARMA(3,3)
```

```
##   row col
## 3   4   4
```

3.2. AIC corregido

```
# Armamos una matriz con los valores del AIC corregido para valores del 1 al 6 de p y q

norder <- 6 # Máximo orden de rezagos que queremos considerar
aicc <- matrix(0, norder + 1, norder + 1)
rownames(aicc) <- c(0:norder)
colnames(aicc) <- c(0:norder)

for (i in 1:(norder + 1)) {
  for (j in 1:(norder + 1)) {
    modeloij <- Arima(comercio, order = c(i-1, 0, j-1), method = 'ML')
    aicc[i,j] <- modeloij$aicc
  }
}

aicc
```

```
##          0        1        2        3        4        5        6
## 0 630.8390 631.2073 624.7533 626.7747 626.6449 629.4001 631.4469
## 1 628.8148 627.5464 625.2852 628.1030 625.0578 627.3598 631.2614
## 2 621.1625 618.9747 621.3308 624.2196 627.2682 631.1736 635.4004
## 3 622.4814 621.1767 623.5899 620.9294 624.4431 628.6757 633.3974
## 4 625.3631 624.4817 624.9388 631.2178 635.4354 640.2002 645.3227
## 5 626.0040 627.3144 627.8981 628.6992 633.3506 638.6511 643.7622
## 6 628.6895 631.0240 632.1844 636.9123 641.5690 650.6843 657.1844
```

```
min(aicc)
```

```
## [1] 618.9747
```

```r
which(aicc == min(aicc), arr.ind = TRUE) # El AIC corregido mínimo se da para un ARMA(2,1)
```

```
##   row col
## 2   3   2
```

3.3. BIC

```r
# Armamos una matriz con los valores del BIC para valores del 1 al 6 de p y q

norder <- 6 # Máximo orden de rezagos que queremos considerar
bic <- matrix(0, norder + 1, norder + 1)
rownames(bic) <- c(0:norder)
colnames(bic) <- c(0:norder)

for (i in 1:(norder + 1)) {
  for (j in 1:(norder + 1)) {
    modeloij <- Arima(comercio, order = c(i-1, 0, j-1), method = 'ML')
    bic[i,j] <- modeloij$bic
  }
}

bic
```

```
##          0        1        2        3        4        5        6
## 0 633.3567 634.7474 629.1347 631.7957 632.0793 634.9935 636.9119
## 1 632.3549 631.9279 630.3062 633.5374 630.6513 632.8249 636.2712
## 2 625.5440 623.9957 626.7652 629.8130 632.7333 636.1834 639.5816
## 3 627.5024 626.6112 629.1834 626.3944 629.4529 632.8568 636.3205
## 4 630.7975 630.0752 630.4038 636.2276 639.6166 643.1233 646.4904
## 5 631.5975 632.7794 632.9079 632.8804 636.2737 639.8189 642.5945
## 6 634.1545 636.0338 636.3656 639.8354 642.7368 649.5167 652.9988
```

```r
min(bic)
```

```
## [1] 623.9957
```

```r
which(bic == min(bic), arr.ind = TRUE) # El BIC mínimo se da para un ARMA(2,1)
```

```
##   row col
## 2   3   2
```

4. Diagnóstico del modelo

4.1. Análisis gráfico de los residuos

```r
# Guardamos los residuos del modelo
residuos1 <- modelo1$residuals

# Buscamos los residuos máximos y mínimos

max(residuos1)
```

```
## [1] 5912.371
```

```r
which.max(residuos1)
```

```
## [1] 26
```

```r
time(residuos1)[which.max(residuos1)] # Junio de 2022
```

```
## [1] 2022.25
```

```r
min(residuos1)
```

```
## [1] -12143.76
```

```r
which.min(residuos1)
```

```
## [1] 18
```

```r
time(residuos1)[which.min(residuos1)] # Junio de 2020
```

```
## [1] 2020.25
```

```r
# Graficamos los residuos
residuos1 %>% autoplot() +
  labs(x = "Fecha",
       y = "Residuos") +
  geom_hline(yintercept = 0, color = "red")
```

Figura 4: Residuos de un modelo AR(2) para el PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023.

```
# Residuos estandarizados
residuos1_est <- residuos1/sqrt(modelo1$sigma2)
residuos1_est %>% autoplot() +
  labs(x = "Fecha",
       y = "Residuos estandarizados",
       title = "PIB comercio") +
  geom_hline(yintercept = 0, color = "black") +
  geom_hline(yintercept = 3, color = "red", linetype = "dotted") +
  geom_hline(yintercept = -3, color = "red", linetype = "dotted")
```

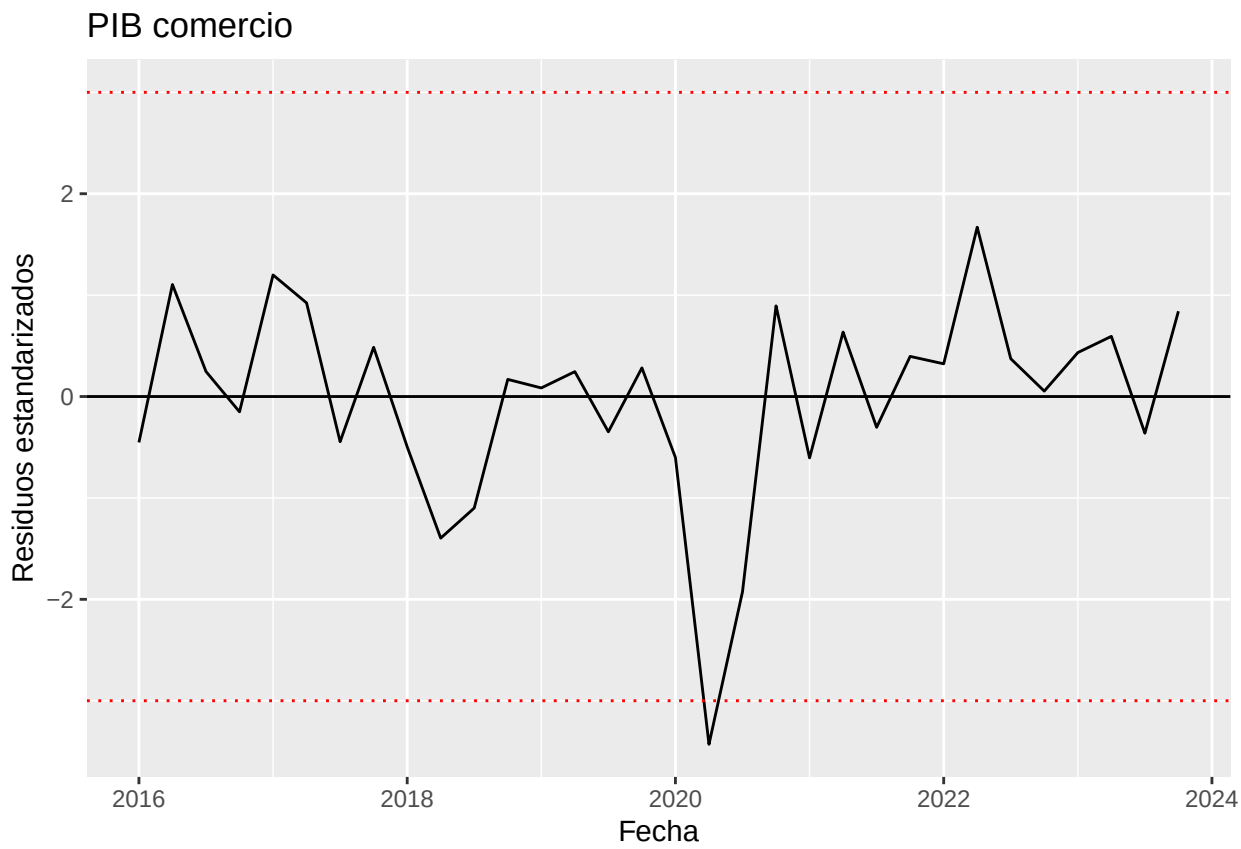


Figura 5: Residuos estandarizados de un modelo AR(2) para el PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023.

```
time(residuos1_est)[which.max(residuos1_est)] # Junio de 2022
```

```
## [1] 2022.25
```

4.2. Autocorrelación de los residuos

4.2.1. FAC y FACP de los residuos

```
# FAC
residuos_acf <- ggAcf(residuos1, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
residuos_pacf <- ggAcf(residuos1, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")
```

```
grid.arrange(residuos_acf, residuos_pacf)
```

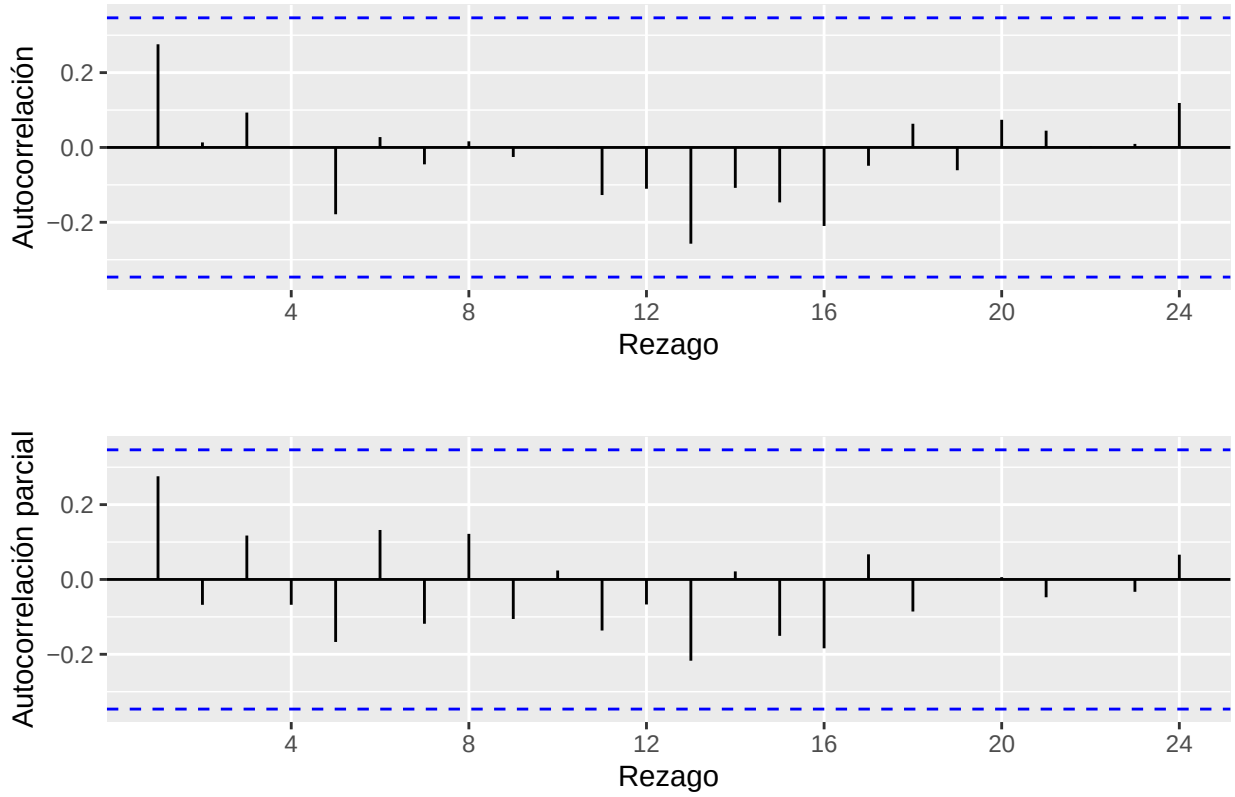


Figura 6: Funciones de Autocorrelación y Autocorrelación Parcial estimadas de los residuos de un modelo AR(2) para el PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023.

```
# Otra posibilidad es graficar a la vez los residuos, su FAC y su FACP
# checkresiduals(residuos1)
# tsdisplay(residuos1)
```

4.2.2. Contraste de autocorrelación de los residuos

```
# Test de Ljung-Box

# Se rechaza la hipótsis nula de no autocorrelación de los residuos
Box.test(residuos1,
         lag = 10,
         type = "Ljung-Box",
         fitdf = 2) # p + q de un modelo ARMA(p,q)
```

```
##
```

```
##  Box-Ljung test
##
## data:  residuos1
## X-squared = 4.4506, df = 8, p-value = 0.8144
```

4.3. Homocedasticidad de los residuos

```
residuos1_2 <- residuos1^2

grafico_residuos2 <- autoplot(residuos1_2) +
  labs(x = "Fecha",
       y = expression(epsilon[t] ^ 2))

acf_residuos2 <- ggAcf(residuos1_2,type = "correlation") +
  labs(x = "Rezago",
       y = "FAC",
       title = "")

grid.arrange(grafico_residuos2, acf_residuos2)
```

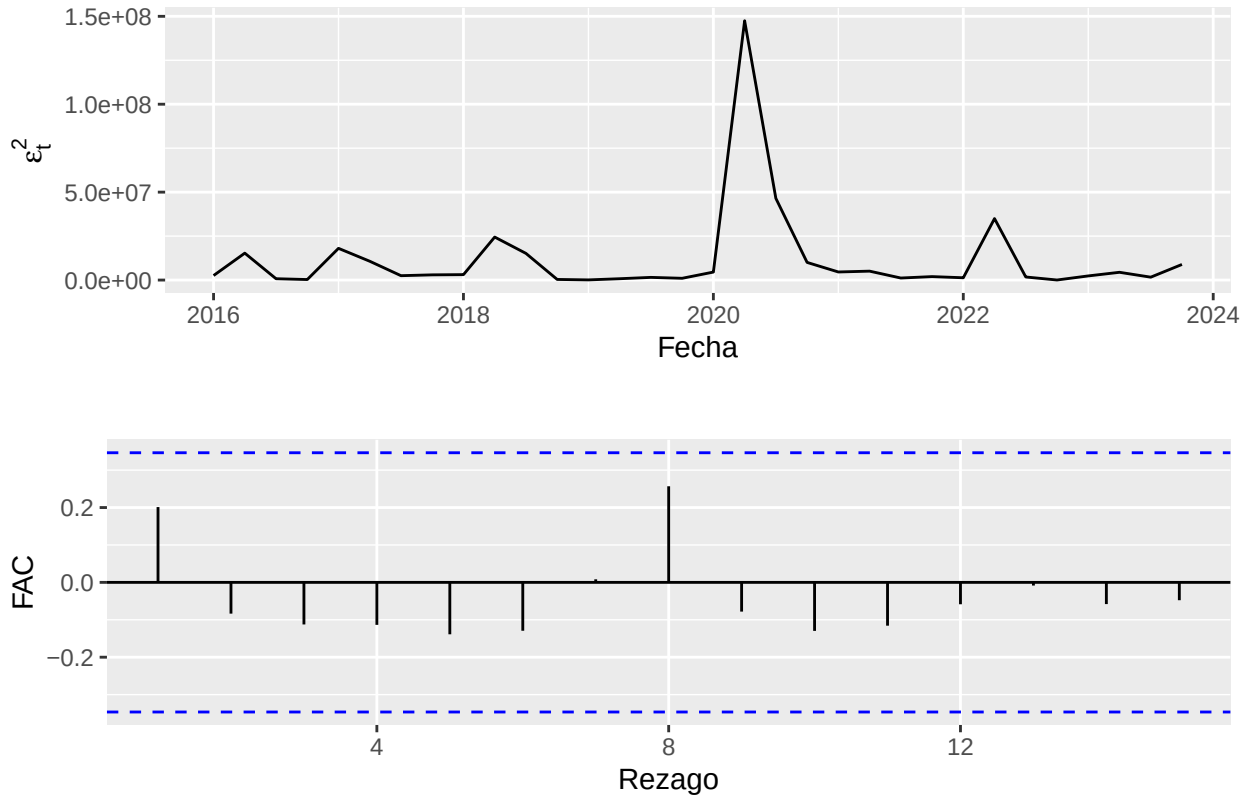


Figura 7: Gráfico y autocorrelograma del cuadrado de los residuos de un modelo AR(2) para el PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023.

4.4. Normalidad de los residuos

4.4.1. QQ-plot de los residuos

```
# Armamos el QQ-plot de los residuos
ggplot(residuos1, aes(sample = residuos1)) +
  stat_qq() +
  stat_qq_line(color = "red") +
  labs(x = "Cuantiles teóricos",
       y = "Cuantiles de la muestra")
```

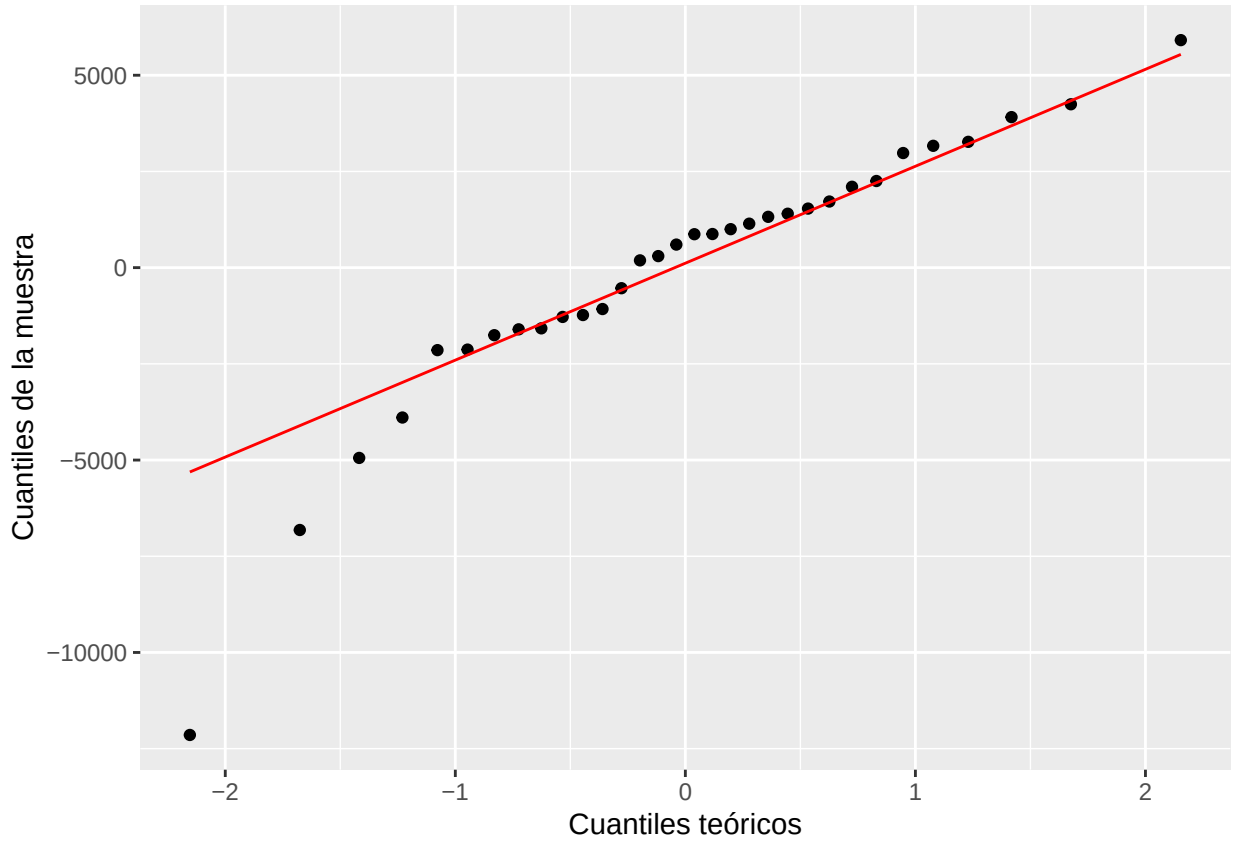


Figura 8: QQ-plot de los residuos de un modelo AR(2) para el PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023.

4.4.2. Histograma de los residuos

```
# Hacemos un histograma de los residuos
ggplot(data = residuos1) +
  geom_histogram(aes(x = residuos1, y = ..density..)) +
  stat_function(fun = dnorm,
                args = list(mean = mean(residuos1),
                            sd = sd(residuos1)),
```

```
                col = "red",
                size = 1) +
  labs(x = "Residuos",
       y = "Densidad")
```

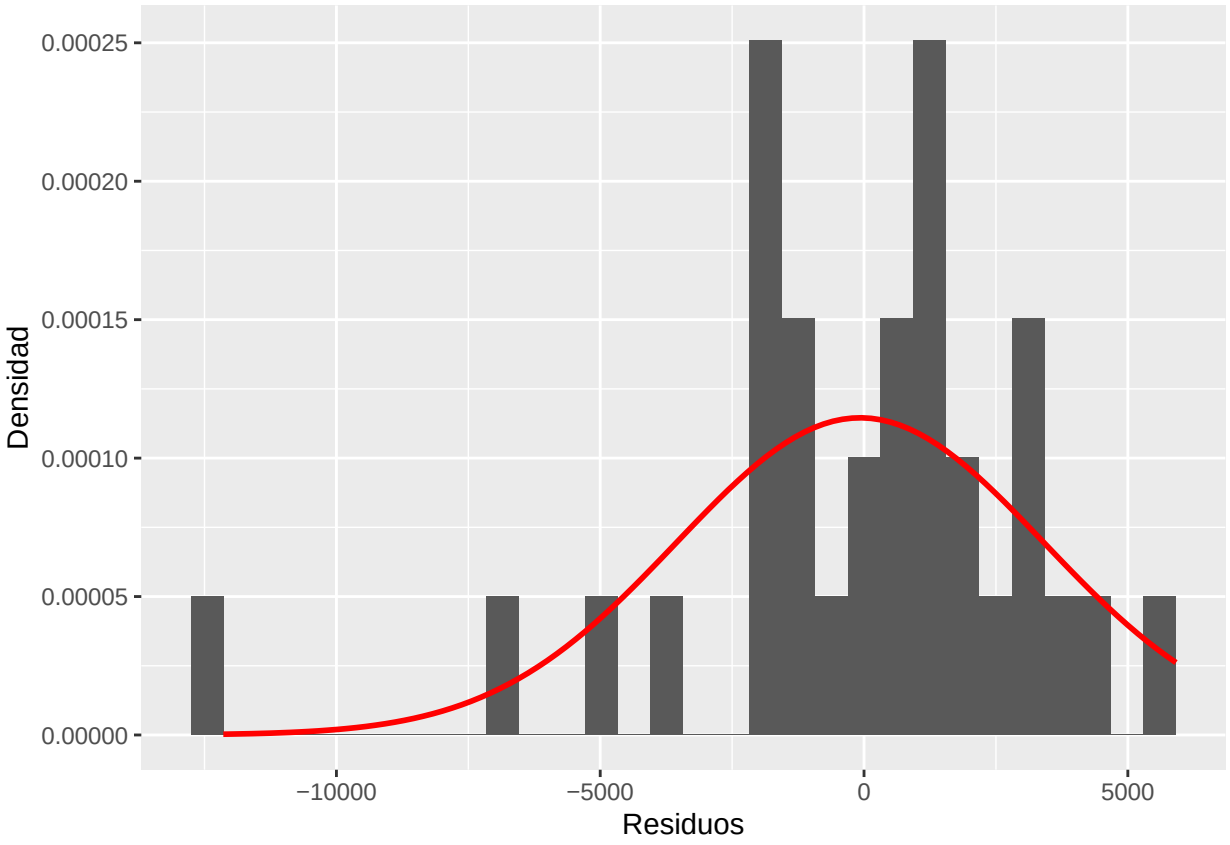


Figura 9: Histograma de los residuos de un modelo AR(2) para el PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) entre 2016 y 2023. La línea roja corresponde a una densidad normal con media y desvío muestrales igual al de los residuos.

4.4.3. Contrastes de normalidad de los residuos

```
# Tests de Shapiro y Jarque-Bera
# Se rechaza la hipótesis nula de normalidad dado que hay outliers
shapiro.test(residuos1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuos1
## W = 0.9036, p-value = 0.007657
```

```r
JarqueBera.test(residuos1)
```

```
##
##  Jarque Bera Test
##
## data:  residuos1
## X-squared = 21.904, df = 2, p-value = 1.753e-05
##
##
##  Skewness
##
## data:  residuos1
## statistic = 1.3726, p-value = 0.001525
##
##
##  Kurtosis
##
## data:  residuos1
## statistic = 5.9819, p-value = 0.0005749
```

5. Validación del modelo

5.1. Errores de predicción a un paso dentro de la muestra

```r
# Obtenemos medidas de los errores de predicción a un paso dentro de la muestra (residuos)

accuracy(modelo1)
```

```
##                      ME     RMSE      MAE        MPE     MAPE      MASE
## Training set -73.20484 3428.398 2497.484 -0.5355904 4.582298 0.5984372
##                    ACF1
## Training set 0.2758011
```

5.2. Ajuste fuera de la muestra

```r
# Definimos una muestra de entrenamiento ("training set") hasta 2022 inclusive
train_comercio <- window(comercio, end = c(2022,4))

# Dejamos los datos de 2023 como conjunto de entrenamiento ("test set")
test_comercio <- window(comercio, start = 2023)
n <- length(test_comercio)
```

```r
# Estimamos los modelos para el training set

modelo1_train <- Arima(y = train_comercio,
                  order = c(2, 0, 0),
                  lambda = NULL,
                  fixed = c(0, NA, NA))
```

```r
# Predecimos fuera de la muestra (el horizonte de predicción
# será igual al largo del test set)

pred1_test <- forecast(modelo1_train, h = n)
```

```r
# Graficamos las predicciones obtenidas

grafico_pred1_test <- autoplot(pred1_test) +
  autolayer(comercio, color = "black") +
  labs(x = "Fecha",
       y = "PIB Comercio",
       title = "")

grafico_pred1_test
```

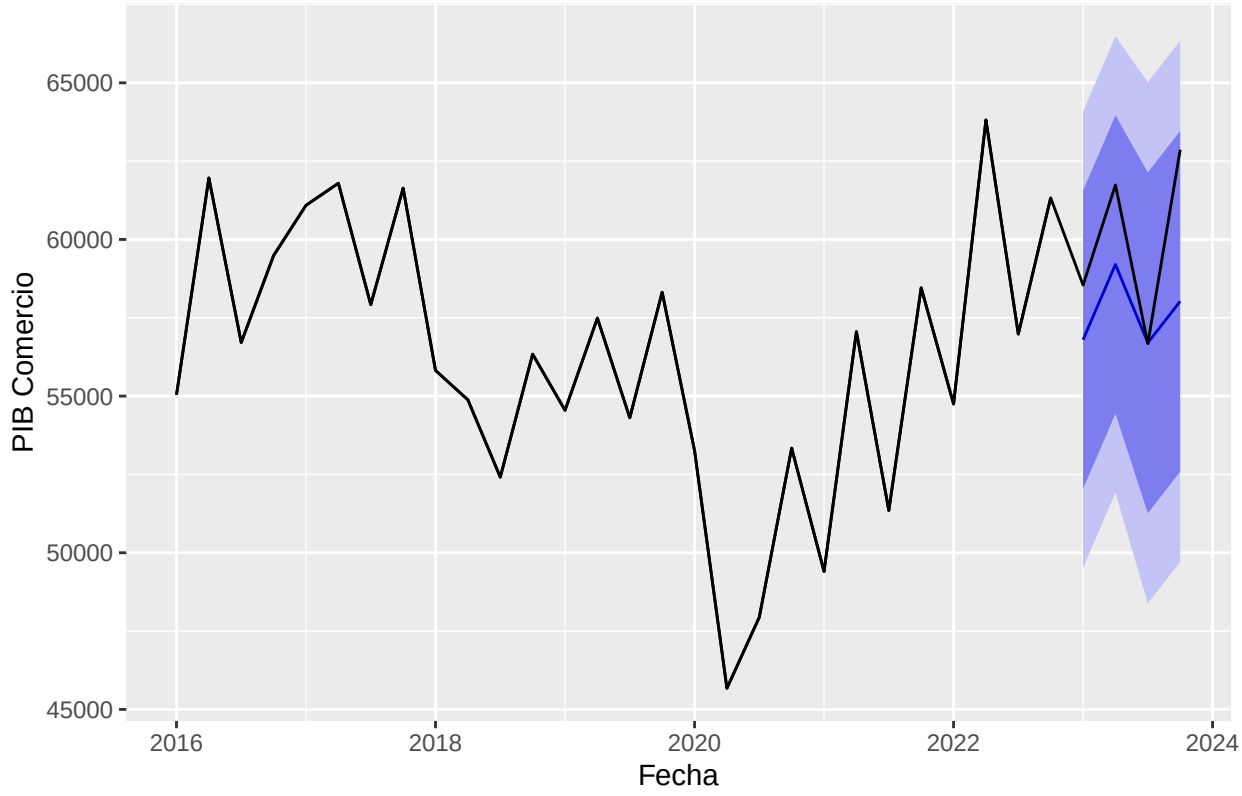


Figura 10: Predicciones en el conjunto de prueba del PIB del sector Comercio, alojamiento y suministro de comidas y bebidas (millones de pesos a precios constantes de 2016) para un modelo AR(2). La línea azul corresponde a las predicciones.

```r
# Obtenemos medidas de los errores de predicción fuera de la muestra
# El segundo argumento de la función accuracy() corresponde al
# verdadero valor de la serie (conjunto de prueba)

accuracy(pred1_test, test_comercio)
```

```
##                       ME     RMSE      MAE        MPE     MAPE      MASE
## Training set  -98.67626 3580.095 2600.855 -0.6200449 4.804871 0.5719184
## Test set     2273.17083 2867.727 2287.110  3.6842910 3.708887 0.5029271
##                    ACF1 Theil's U
## Training set  0.3001828        NA
## Test set     -0.5442531 0.6506197
```