Cómputo Estadístico

September 30, 2024

Godinez Bravo Diego

Tarea 2 - Procesos Estocásticos

Centro de Investigación en Matemáticas

Maestría en Cómputo Estadístico

0.1 Problema 1

0.1.1 Comprobación del Diagnóstico de los Residuales del Modelo.

Si se ha extraído toda la información sistemática con un modelo de pronóstico, entonces lo que queda, el residual, debe ser ruido blanco. Con más precisión, las innovaciones verdaderas son ruido blanco, y si un modelo es buena aproximación de Wold, entonces sus errores de pronóstico a una etapa se deben aproximar al ruido blanco.

Los residuales del modelo están en el análogo dentro de la muestra de los errores de pronóstico a una etapa fuera de la muestra. En consecuencia, vemos la utilidad de varias pruebas de la hipótesis que los residuales son ruido blanco. La de Durbin-Watson es la prueba más común.

Recuérdese que la dócima de Durbin-Watson, descrita en el apéndice del capítulo 1 es:

$$DW = \frac{\sum_{t=2}^{T}(e_t - e_{t-1})^2}{\sum_{t=1}^{T} e_t^2}$$

Observe que

$$\sum_{t=2}^{T}(e_t - e_{t-1})^2 \sim 2\sum_{t=2}^{T} e_t^2 - 2\sum_{t=2}^{T} e_t e_{t-1}$$

y así

$$DW = 2(1 - \hat{p}(1))$$

Y entonces la prueba de Durbin-Watson se basa efectivamente solo en la correlación de la primera muestra, y en realidad solo prueba si la primera autocorrelación es cero. En consecuencia, se dice que la de Durbin-Watson es una prueba de **correlación seriada de primer orden**, o correlación en serie de primer orden. Además, la prueba de Durbin-Watson no es válida en presencia de variables dependientes rezagadas. En ambos casos nos gustaría un marco más general y flexible para diagnosticar la correlación seriada. El correlograma de residuales, formado por las autocorrelaciones muestrales residuales, las autocorrelaciones parciales muestrales y los estadísticos Q asociados, desempeñan este papel.

a) Cuando describimos el correlograma en la prueba, nos enfocamos al caso de una serie temporal observada, para el que demostramos que los estadísticos Q se distribuyen como χ^2_m. Sin embargo, ahora deseamos evaluar si las perturbaciones no observadas del modelo son ruido blanco. Para hacerlo, usaremos los residuales del modelo, que son estimados de las perturbaciones no observadas. Ya que un modelo se ajusta para obtener los residuales, necesitamos tomar en cuenta los grados de libertad usados. La consecuencia es que la distribución del estadístico Q con la hipótesis de ruido blanco se aproxima mejor como una variable aleatoria χ^2_{m-k} en la que k es la cantidad de parámetros que se estiman. Es la razón, por ejemplo, por la que no se mencionan los valores p (de hecho, ni los programas estadísticos los calculan) para los estadísticos Q asociados con el correlograma de residuales de nuestro modelo de pronóstico de empleo, sino hasta que $m > k$.

b) La prueba h de Durbin es una alternativa en la prueba de Durbin-Watson. Como en el caso de la prueba Durbin-Watson, el fin es detectar correlación en serie de primer orden, pero es válida en presencia de variables dependientes demoradas. Busque información acerca de las generalidades de la prueba h de Durbin en la bibliografía y escriba lo encontrado.

c) La prueba de Breusch-Godfrey es otra alternativa a la de Durbin-Watson. Permite detectar correlación seriada de orden p, y también es válida en presencia de variables rezagadas. Investigue en la bibliografía acerca del procedimiento Breusch-Godfrey y escriba lo que aprendió.

d) ¿Cuál de las pruebas es la más útil para evaluar las propiedades de residuos a partir de modelos de pronóstico: el correlograma de residuales, la prueba h de Durbin o la prueba de Breusch-Godfrey? ¿Por qué?

0.1.2 Prueba H de Durbin

El estadístico de Durbin-Watson, desarrollado por los econometristas Durbin & Watson, fue diseñado para detectar la autocorrelación significativa de primer orden en los residuos de un análisis de regresión. Su eficiencia en contextos donde se asume independencia en modelos de regresión lineal condujo a su uso generalizado. Lo que resultó en su aplicación en situaciones inapropiadas, como en los casos donde existe la presencia de **variables dependientes rezagadas**, y por ende a conclusiones erróneas. Para abordar estos casos, Durbin (1970) propuso la prueba h, que permite evaluar la **autocorrelación en procesos autorregresivos de primer orden**.

El estadístico h se define como:

$$h = \left(1 - \frac{1}{2}d\right)\left[\frac{n}{1 - n\hat{V}(\hat{\beta}_1)}\right]^{\frac{1}{2}}$$

donde d es el estadístico Durbin-Watson, n es el tamaño de muestra y $\hat{V}$ representa la estimación de la varianza de $\hat{\beta}_1$. Aquí β_1 es el coeficiente de la variable dependiente rezagada y $\hat{\beta}_1$ su estimación.

Durbin demostró que, bajo ciertos supuestos, h sigue asintóticamente una distribución normal con media cero y varianza unitaria (i.e., $h \sim N(0,1)$) cuando los errores no están correlacionados. Además, sugirió utilizar la tabla de la distribución normal estándar para contrastar la hipótesis nula de errores no correlacionados.

Referencias

- Park, S.-B. (1975). On the small-sample power of Durbin's h test. Journal of the American Statistical Association, 70(349), 60-63.
- Proïa, F. (2013). Further results on the h-test of Durbin for stable autoregressive processes. Journal of Multivariate Analysis, 118, 77-101.

0.1.3 Prueba de Breusch-Godfrey

Breusch & Godfrey (1978) propusieron el uso de los multiplicadores de Lagrange (LM) para detectar autocorrelación en los residuos de un modelo de regresión. Este método se basa en el concepto de 'alternativas localmente equivalentes' (*locally equivalent alternatives*, LEA), lo que permite identificar incluso leves autocorrelaciones en los errores. En otras palabras, la prueba está diseñada para detectar pequeñas desviaciones de la independencia en los residuos del modelo.

El enfoque del método consiste en aplicar dos regresiones. La primera, llamada regresión primaria, se realiza sobre el modelo original. Luego, los residuos estimados de esta regresión se emplean como variable dependiente en una segunda regresión, conocida como regresión auxiliar, donde las variables independientes incluyen los regresores del modelo original mas los residuos rezagados (valores residuales pasados).

El estadístico de la prueba se obtiene multiplicando el número de observaciones por el coeficiente de determinación de la regresión auxiliar. De manera que el estadístico TR^2 sigue una distribución χ^2 bajo la hipótesis nula (H_0 : independencia de los residuos).

La prueba de Breusch-Godfrey tiene la ventaja de ser válida para modelos dinámicos y puede extenderse fácilmente para detectar **autocorrelación de orden superior**.

Referencias

- Edgerton, D. & Shukur, G. (1999) Testing autocorrelation in a system perspective testing autocorrelation, Econometric Reviews, 18:4, 343-386,

---

La **prueba de Breusch-Godfrey** es considerada más útil debido a su capacidad para identificar la autocorrelación en múltiples rezagos ($t-1, t-2$, etc.). A diferencia de la **prueba h de Durbin**, que está limitada a los residuos de los periodos inmediatamente anteriores (t y $t-1$). Por otro lado, el **correlograma de residuos** podría ser una herramienta muy valiosa para visualizar gráficamente la autocorrelación de los residuos y ser un paso inicial antes de aplicar una prueba estadística más formal para la evaluación de los residuos de un modelo.

En resumen, cada una de estas pruebas tiene su propia importancia y utilidad, dependiendo de la complejidad del problema y los objetivos que se persigan.

0.2 Problema 2

Demuestre paso a paso que:

$$\gamma(\tau) = E(y_t y_{t-\tau}) = E((\epsilon_\tau + \theta\epsilon_{t-1})(\epsilon_{t-\tau} + \theta\epsilon_{t-\tau-1})) \begin{cases} \theta\sigma^2, \tau = 1 \\ 0 \text{ de otro modo} \end{cases}$$

Completando los pasos que faltan evaluando en forma explicita la expectativa $E((\epsilon_\tau + \theta\epsilon_{t-1})(\epsilon_{t-\tau} + \theta\epsilon_{t-\tau-1}))$.

0.2.1 Solución

Desarrollando la expresión:

$$\begin{aligned} \gamma(\tau) = E(y_t y_{t-\tau}) &= E((\epsilon_t + \theta\epsilon_{t-1})(\epsilon_{t-\tau} + \theta\epsilon_{t-\tau-1})) \\ &= E(\epsilon_t\epsilon_{t-\tau} + \theta\epsilon_t\epsilon_{t-\tau-1} + \theta\epsilon_{t-1}\epsilon_{t-\tau} + \theta^2\epsilon_{t-1}\epsilon_{t-\tau-1}) \\ &= E(\epsilon_t\epsilon_{t-\tau}) + E(\theta\epsilon_t\epsilon_{t-\tau-1}) + E(\theta\epsilon_{t-1}\epsilon_{t-\tau}) + E(\theta^2\epsilon_{t-1}\epsilon_{t-\tau-1}) \end{aligned}$$

donde $\epsilon_t \sim WN(0, \sigma^2)$.

Sea $\tau = 1$.

$$\begin{aligned} E(y_t y_{t-1}) &= E(\epsilon_t\epsilon_{t-1}) + E(\theta\epsilon_t\epsilon_{t-1-1}) + E(\theta\epsilon_{t-1}\epsilon_{t-1}) + E(\theta^2\epsilon_{t-1}\epsilon_{t-1-1}) \\ &= E(\epsilon_t\epsilon_{t-1}) + E(\theta\epsilon_t\epsilon_{t-2}) + E(\theta\epsilon_{t-1}^2) + E(\theta^2\epsilon_{t-1}\epsilon_{t-2}) \\ &= E(\epsilon_t)E(\epsilon_{t-1}) + \theta E(\epsilon_t)E(\epsilon_{t-2}) + \theta E(\epsilon_{t-1}^2) + \theta^2 E(\epsilon_{t-1})E(\epsilon_{t-2}) \\ &= \theta E(\epsilon_{t-1}^2) \\ &= \theta\sigma^2 \end{aligned}$$

Por lo tanto, cuando $\tau = 1$:

$$E(y_t y_{t-\tau}) = \theta\sigma^2$$

Sea $\tau > 1$.

$$\begin{aligned} E(y_t y_{t-i}) &= E(\epsilon_t\epsilon_{t-i}) + E(\theta\epsilon_t\epsilon_{t-i-1}) + E(\theta\epsilon_{t-1}\epsilon_{t-i}) + E(\theta^2\epsilon_{t-1}\epsilon_{t-i-1}) \\ &= E(\epsilon_t\epsilon_{t-i}) + E(\theta\epsilon_t\epsilon_{t-k}) + E(\theta\epsilon_{t-1}\epsilon_{t-i}) + E(\theta^2\epsilon_{t-1}\epsilon_{t-k}) \\ &= E(\epsilon_t)E(\epsilon_{t-i}) + \theta E(\epsilon_t)E(\epsilon_{t-k}) + \theta E(\epsilon_{t-1})E(\epsilon_{t-i}) + \theta^2 E(\epsilon_{t-1})E(\epsilon_{t-k}) \\ &= 0 \end{aligned}$$

con $i > 1$.

De manera que, cuando $\tau > 1$:

$$E(y_t y_{t-\tau}) = 0$$

0.3 Problema 3

Modelos de Agregación y Desagregación: Pronóstico de Arriba Abajo y de Abajo Arriba. El asunto de la agregación se relaciona con el de los métodos y la complejidad.

Con frecuencia se desea pronosticar un agregado, como por ejemplo las ventas totales de una empresa manufacturera, pero podemos emplear un método agregado o desagregado.

Supongamos que las ventas totales están formadas por las de 35 productos cuya información se encuentra en **Tarea02_Datos.txt**. El método agregado, o de arriba abajo o macro, es simplemente modelar y pronosticar las ventas totales. El método desagregado, o de abajo arriba, o micro, es modelar y pronosticar por separado las ventas de los productos individuales, para después sumarlas.

Quizá sea sorprendente, pero es imposible saber cuál de los métodos es mejor, el agregado o desagregado. Todo depende de las circunstancias del caso; la única forma de saberlo es probar ambos métodos y comparar los resultados del pronóstico.

Argumente ventajas y desventajas de utilizar un método agregado o desagregado en función de la información proporcionada en el archivo **Tarea02_Datos.txt**.

0.3.1 Solución

Cargar el conjunto de datos contenido en el archivo .txt

[2]:
```
library("forecast")
library("ggplot2")
library("dplyr")
```

[3]:
```
path = "/home/aspphem/Desktop/MCE/StatisticalComputing/T2/Tarea02_Datos.txt" #␣
↪file path
data <- read.delim(path, header = TRUE, sep = "|") # read txt file
rows <- nrow(data) # no.of rows
head(data, n = 5) # data preview
```

A data.frame: 5 × 6

	Semana <int>	Articulo <chr>	Unidades <int>	Venta_Pesos <dbl>	Num_Tiendas <dbl>	Cant_Tickets <dbl>
1	20190918	Art_01	89301	2014625.5	1.00	5.51
2	20200115	Art_01	44710	1088903.5	1.11	5.31
3	20200909	Art_01	59393	1520705.9	1.08	4.38
4	20201216	Art_01	35851	918757.3	1.09	4.64
5	20210106	Art_01	39455	1003456.5	1.17	4.79

[4]:
```
data$Unidades <- data$Unidades/data$Num_Tiendas
data <- data %>%
    mutate_at(vars(Unidades), as.integer)
head(data, n = 5) # data preview
```

A data.frame: 5 × 6

	Semana <int>	Articulo <chr>	Unidades <int>	Venta_Pesos <dbl>	Num_Tiendas <dbl>	Cant_Tickets <dbl>
1	20190918	Art_01	89301	2014625.5	1.00	5.51
2	20200115	Art_01	40279	1088903.5	1.11	5.31
3	20200909	Art_01	54993	1520705.9	1.08	4.38
4	20201216	Art_01	32890	918757.3	1.09	4.64
5	20210106	Art_01	33722	1003456.5	1.17	4.79

[5]:
```
summary(data) # data summary statistics
```

```
     Semana            Articulo            Unidades          Venta_Pesos
 Min.   :20190109   Length:4620        Min.   :     0    Min.   :     34
 1st Qu.:20190826   Class :character   1st Qu.: 22185    1st Qu.: 461215
 Median :20200412   Mode  :character   Median : 41116    Median : 882456
 Mean   :20198869                      Mean   : 57627    Mean   :1342606
 3rd Qu.:20201144                      3rd Qu.: 78178    3rd Qu.:1830230
 Max.   :20210714                      Max.   :361907    Max.   :8303722
  Num_Tiendas      Cant_Tickets
 Min.   :1.000   Min.   :1.000
 1st Qu.:1.000   1st Qu.:4.468
 Median :1.080   Median :4.985
 Mean   :1.073   Mean   :4.776
 3rd Qu.:1.090   3rd Qu.:5.330
 Max.   :1.170   Max.   :5.670
```

[6]:
```
str(data) # data types
```

```
'data.frame':	4620 obs. of  6 variables:
 $ Semana      : int  20190918 20200115 20200909 20201216 20210106 20210127
20191225 20200513 20201007 20201230 …
 $ Articulo    : chr  "Art_01" "Art_01" "Art_01" "Art_01" …
 $ Unidades    : int  89301 40279 54993 32890 33722 36524 48134 55167 68417
34382 …
 $ Venta_Pesos : num  2014625 1088903 1520706 918757 1003457 …
 $ Num_Tiendas : num  1 1.11 1.08 1.09 1.17 1.17 1.07 1.07 1.09 1.09 …
 $ Cant_Tickets: num  5.51 5.31 4.38 4.64 4.79 4.25 5.54 3.19 4.68 4.37 …
```

[7]:
```
data <- data[order(data$Semana, decreasing = FALSE),]
```

0.3.2 Modelo de Agregación

[9]:
```
head(data) # data preview
```

A data.frame: 6 × 6

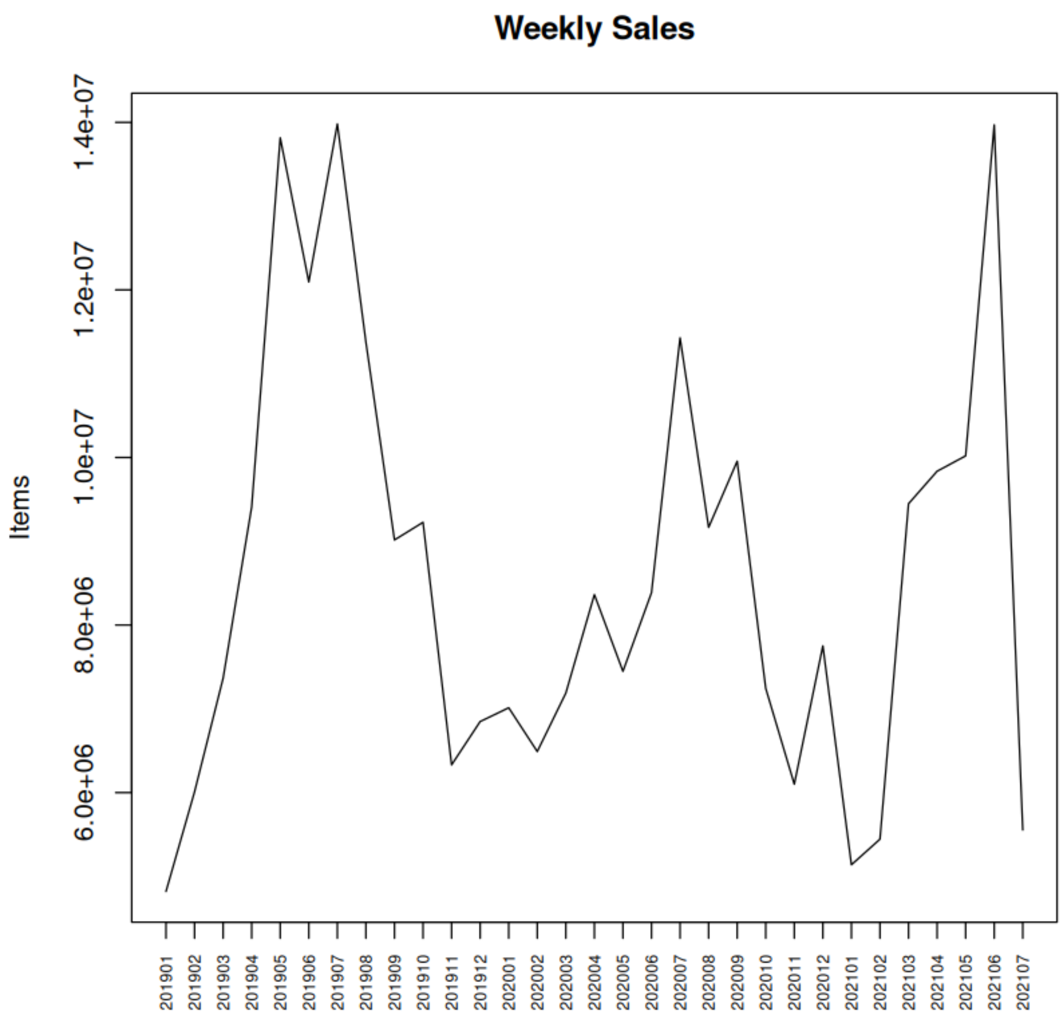
	Semana <int>	Articulo <chr>	Unidades <int>	Venta_Pesos <dbl>	Num_Tiendas <dbl>	Cant_Tickets <dbl>
21	20190109	Art_01	39524	963451.7	1	4.93
198	20190109	Art_02	55323	935570.1	1	4.93
277	20190109	Art_03	82992	2215560.0	1	4.93
410	20190109	Art_04	31119	758731.0	1	4.93
580	20190109	Art_05	6937	130192.4	1	4.93
760	20190109	Art_06	10864	197202.3	1	4.93

[10]:
```
agg <- aggregate(data[, 3], list(substring(data[, 1], 1, 6)), sum) # group up
 ↪data by week
head(agg)
```

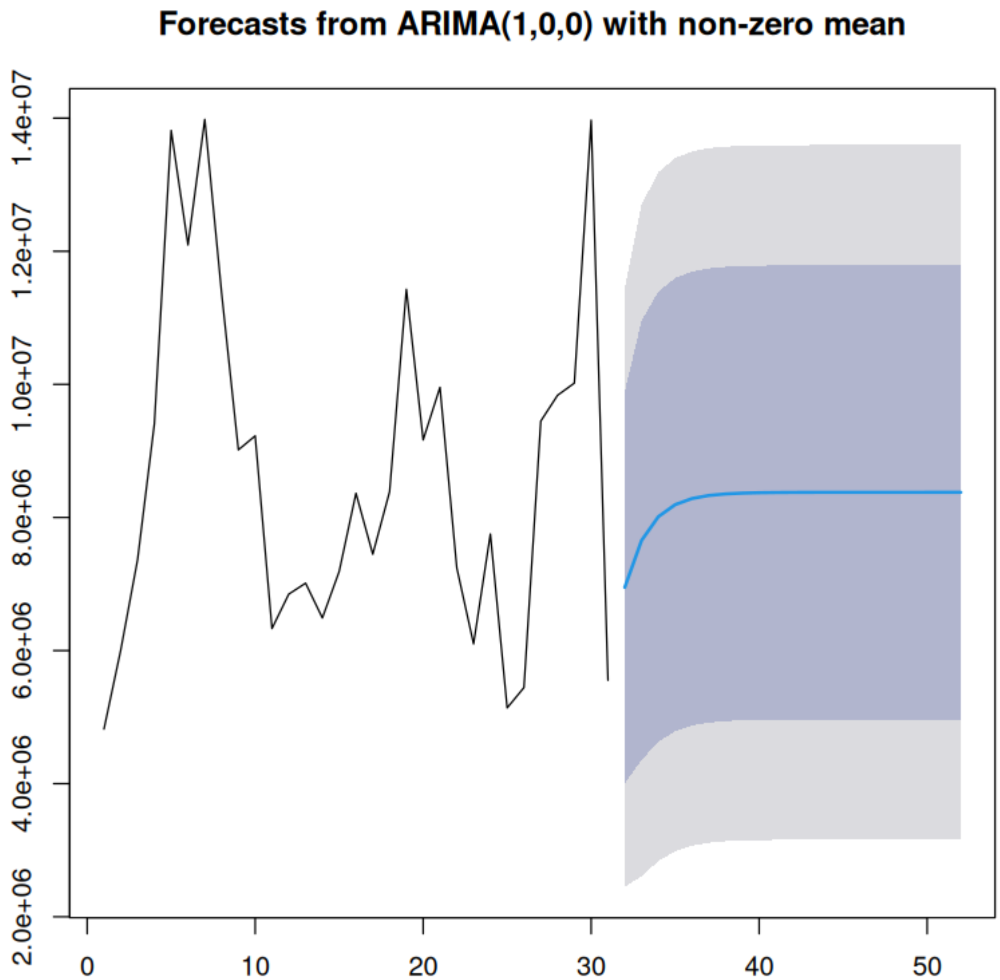
A data.frame: 6 × 2

	Group.1 <chr>	x <int>
1	201901	4820380
2	201902	6009624
3	201903	7364992
4	201904	9408627
5	201905	13815278
6	201906	12094447

[11]:
```
rows  <- nrow(agg) # no.of rows
plot.ts(agg[, 2], ylab = "Items", main = "Weekly Sales", xlab = "", xaxt = "n")
 ↪# plotting time-series
axis(1, 1:rows, agg[,1], las = 2, cex.axis = 0.6)
```



[12]:
```
arima_model <- auto.arima(agg[, "x"]) # fit an ARIMA model on aggregated data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



[13]:
```
fcast$x # original time series
```

A Time Series:
1. 4820380 2. 6009624 3. 7364992 4. 9408627 5. 13815278 6. 12094447 7. 13981023 8. 11377165 9. 9015856 10. 9226113 11. 6331311 12. 6848393 13. 7012905 14. 6490397 15. 7187925 16. 8363336 17. 7447890 18. 8386307 19. 11428137 20. 9165479 21. 9955325 22. 7244257 23. 6100546 24. 7750861 25. 5139019 26. 5444571 27. 9448118 28. 9836013 29. 10019467 30. 13969504 31. 5554454

[14]:
```
fcast$mean # point forecasts as a time series
```

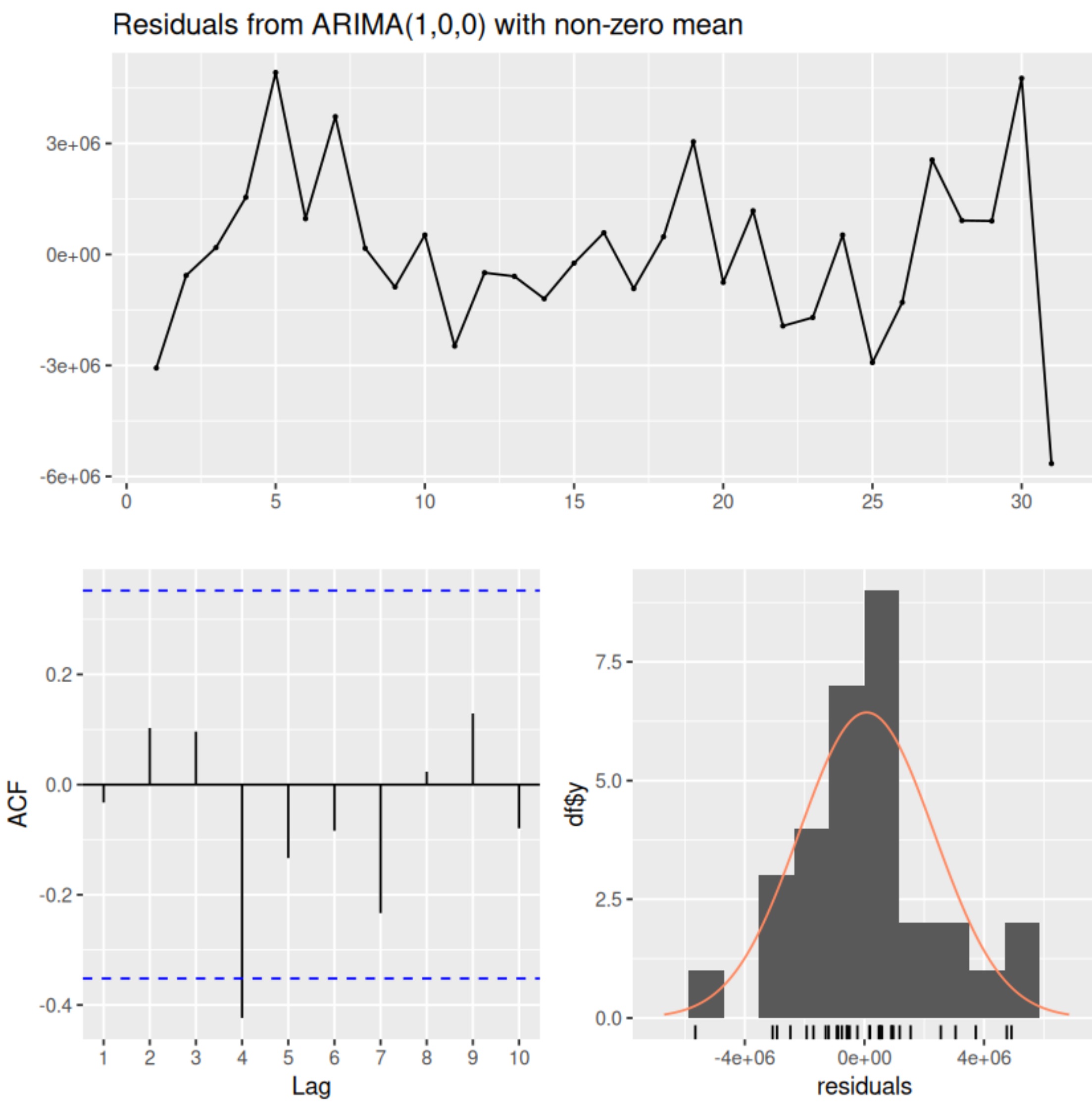
A Time Series:
1. 6949582.89994805 2. 7655299.44834603 3. 8012281.40657319 4. 8192858.32345148 5. 8284201.93944599 6. 8330407.491779 7. 8353780.25891936 8. 8365603.21560344 9. 8371583.77853441 10. 8374609.00597345 11. 8376139.2968781 12. 8376913.38421534

13. 8377304.95107616 14. 8377503.02253286 15. 8377603.21564387 16. 8377653.8976536 17. 8377679.53480654 18. 8377692.50318751 19. 8377699.06315559 20. 8377702.38147122 21. 8377704.06001881

Evaluación de los Residuos

[20]:
```
checkresiduals(forecast(arima_model, h = 21)) # check residuals from a time
↪series model
```

```
        Ljung-Box test

data:  Residuals from ARIMA(1,0,0) with non-zero mean
Q* = 8.5295, df = 5, p-value = 0.1294

Model df: 1.   Total lags used: 6
```



La mayoría de los puntos se encuentran dentro del intervalo de confianza, lo que sugiere que los residuos no presentan autocorrelación y se comportan como **ruido blanco**. Sin embargo, en el **gráfico ACF** se aprecia un pico significativo en el retraso no. 4 (lag = 4). No obstante, la autocorrelación observada no es lo suficientemente alta como para tener un impacto relevante en las predicciones del modelo.

En conclusión, aunque se identifica un pico en el lag 4, su magnitud no es lo suficientemente fuerte para comprometer la precisión de las predicciones del modelo.

0.3.3 Modelo de Desagregación

[14]:
```r
head(data) # data preview
```

A data.frame: 6 × 6

	Semana <int>	Articulo <chr>	Unidades <int>	Venta_Pesos <dbl>	Num_Tiendas <dbl>	Cant_Tickets <dbl>
21	20190109	Art_01	39524	963451.7	1	4.93
198	20190109	Art_02	55323	935570.1	1	4.93
277	20190109	Art_03	82992	2215560.0	1	4.93
410	20190109	Art_04	31119	758731.0	1	4.93
580	20190109	Art_05	6937	130192.4	1	4.93
760	20190109	Art_06	10864	197202.3	1	4.93

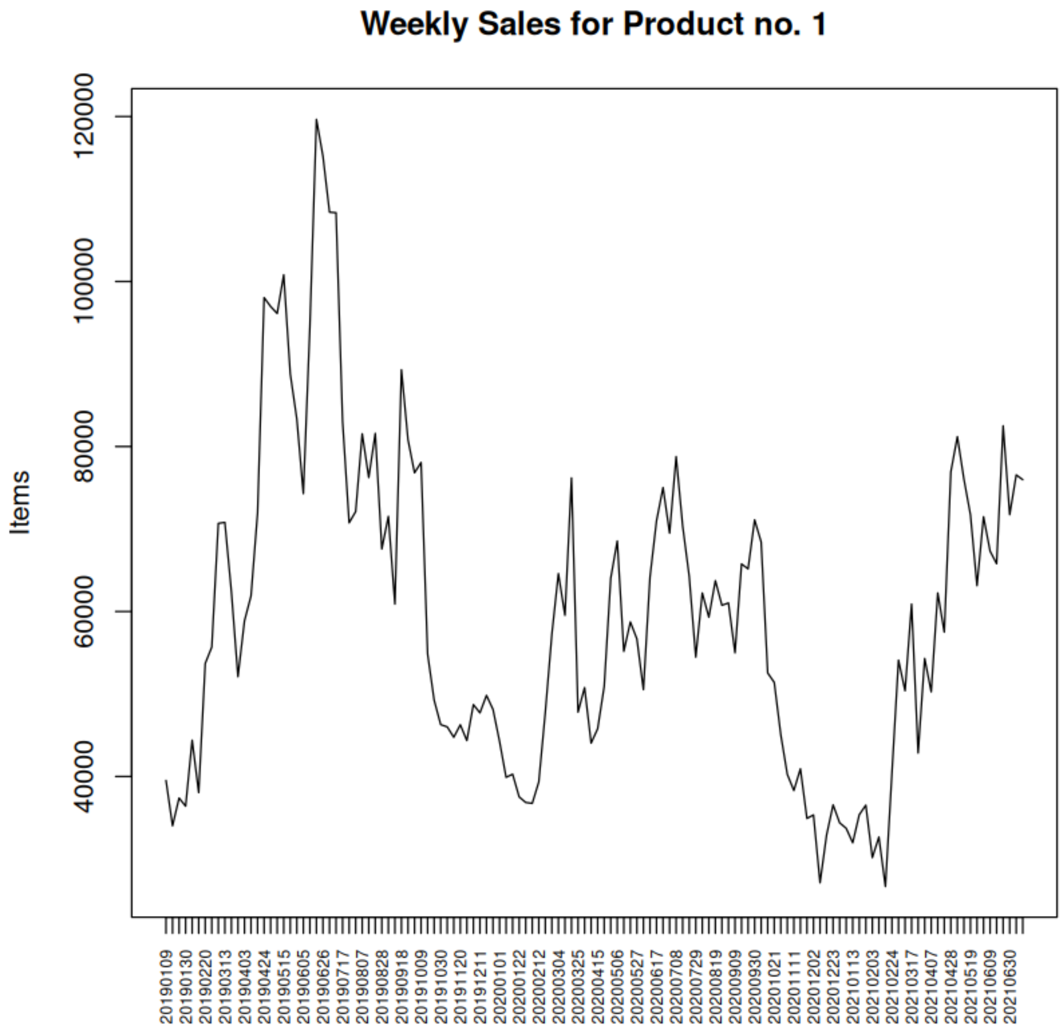
Filtrar el conjunto de datos por el no. de articulo.

[8]:
```r
# Test using Product no. 1
test_product_1 <- data[data$Articulo == "Art_01", ] # filter data by product no.
head(test_product_1)
```

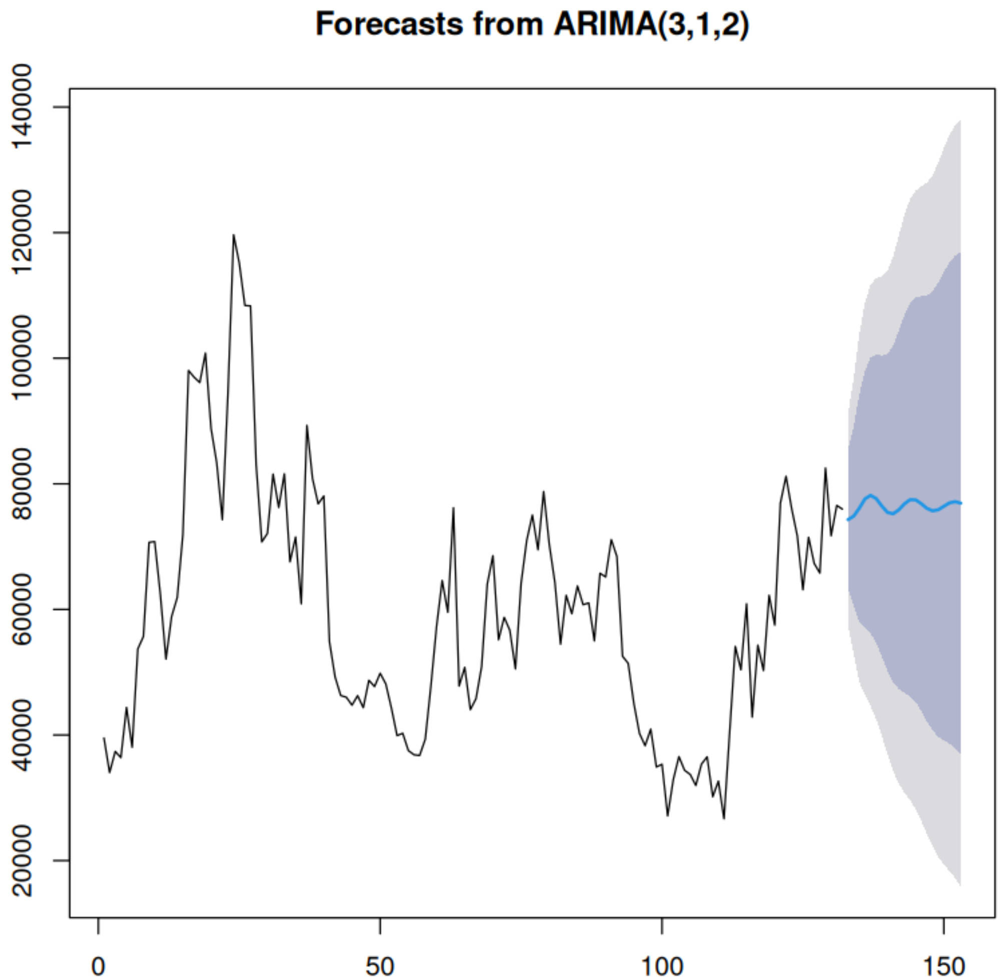
A data.frame: 6 × 6

	Semana <int>	Articulo <chr>	Unidades <int>	Venta_Pesos <dbl>	Num_Tiendas <dbl>	Cant_Tickets <dbl>
21	20190109	Art_01	39524	963451.7	1	4.93
117	20190116	Art_01	34025	827828.1	1	4.98
68	20190123	Art_01	37390	918111.2	1	5.04
62	20190130	Art_01	36403	894717.1	1	4.97
70	20190206	Art_01	44400	1088314.9	1	5.23
86	20190213	Art_01	38053	931962.3	1	5.00

[9]:
```r
rows  <- nrow(test_product_1) # no.of rows
plot.ts(test_product_1[, 3], ylab = "Items", main = "Weekly Sales for Product␣
↪no. 1", xlab = "", xaxt = "n") # plotting time-series
axis(1, 1:rows, test_product_1[,1], las = 2, cex.axis = 0.6)
```



[10]:
```
arima_model <- auto.arima(test_product_1[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



[11]:
```
fcast$x # original time series
```

A Time Series:
1. 39524 2. 34025 3. 37390 4. 36403 5. 44400 6. 38053 7. 53703 8. 55689 9. 70682 10. 70805 11. 62393 12. 52097 13. 58866 14. 61894 15. 71937 16. 98046 17. 96968 18. 96125 19. 100813 20. 88817 21. 83352 22. 74290 23. 94524 24. 119655 25. 115180 26. 108400 27. 108337 28. 83041 29. 70748 30. 72110 31. 81541 32. 76242 33. 81590 34. 67579 35. 71534 36. 60900 37. 89301 38. 80830 39. 76810 40. 78062 41. 54850 42. 49249 43. 46287 44. 46016 45. 44755 46. 46270 47. 44357 48. 48714 49. 47715 50. 49833 51. 48134 52. 44257 53. 39900 54. 40279 55. 37524 56. 36855 57. 36741 58. 39339 59. 47817 60. 57254 61. 64597 62. 59540 63. 76190 64. 47801 65. 50774 66. 44049 67. 45787 68. 50908 69. 64031 70. 68549 71. 55167 72. 58734 73. 56678 74. 50529 75. 64151 76. 70988 77. 75029 78. 69499 79. 78777 80. 70359 81. 64261 82. 54468 83. 62243 84. 59311 85. 63735 86. 60752 87. 61038 88. 54993 89. 65762 90. 65167 91. 71113 92. 68417 93. 52546 94. 51409 95. 45060 96. 40257 97. 38300 98. 40935 99. 34914 100. 35342

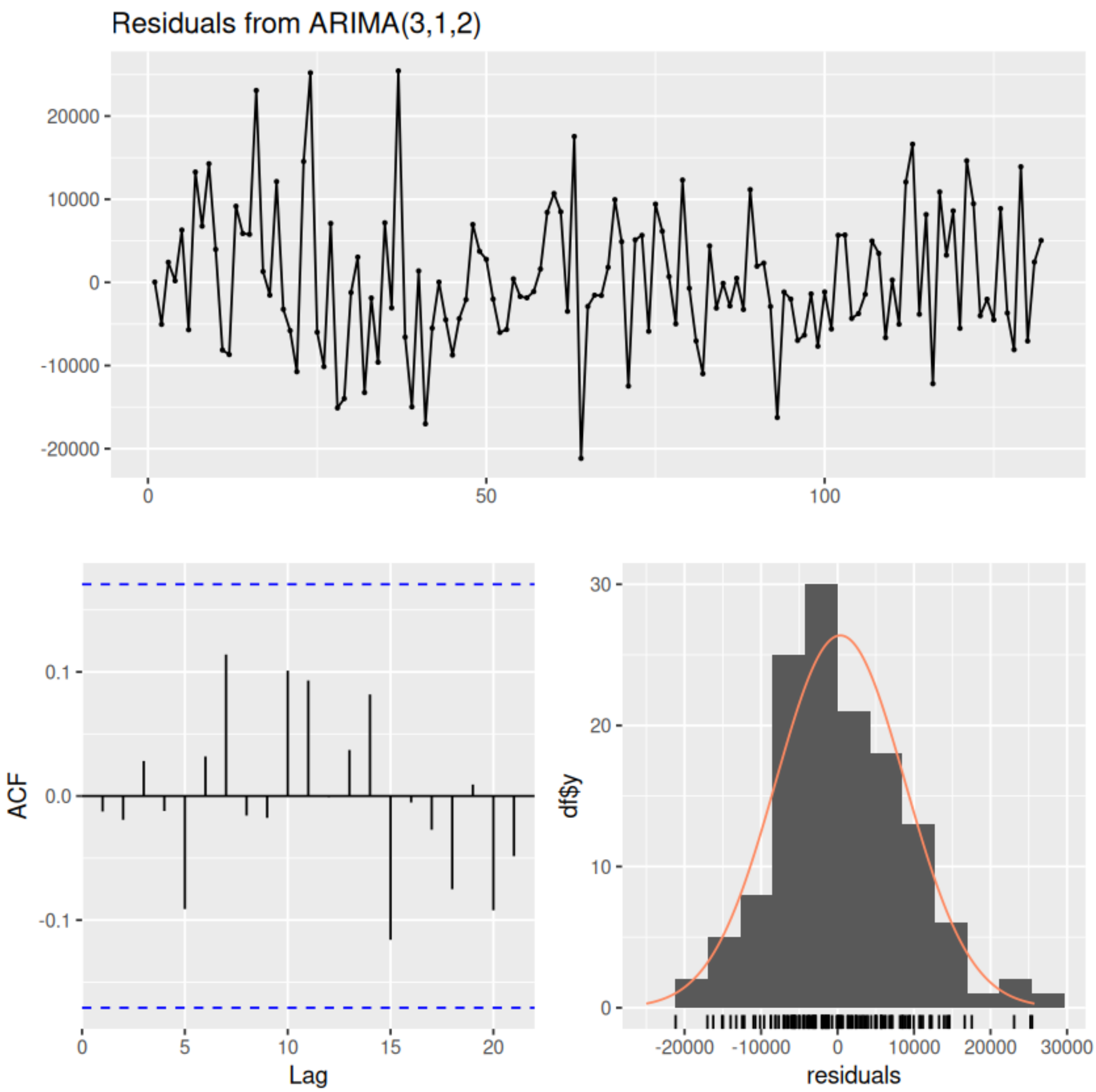
101. 27121 102. 32890 103. 36562 104. 34382 105. 33722 106. 31969 107. 35368 108. 36524 109. 30170 110. 32664 111. 26664 112. 40471 113. 54105 114. 50390 115. 60898 116. 42852 117. 54307 118. 50266 119. 62247 120. 57503 121. 76922 122. 81193 123. 76050 124. 71713 125. 63133 126. 71487 127. 67318 128. 65793 129. 82510 130. 71737 131. 76556 132. 75974

[12]:
```
fcast$mean # point forecasts as a time series
```

A Time Series:
1. 74298.5976365602 2. 74844.9318193606 3. 76117.8095727438 4. 77596.4162088956 5. 78167.8034369761 6. 77642.6109038461 7. 76455.8707507527 8. 75447.1194812156 9. 75228.1191008895 10. 75845.6498094186 11. 76809.5313152394 12. 77469.9183913496 13. 77445.2778642229 14. 76832.3980284821 15. 76088.2700313296 16. 75698.7240850531 17. 75867.9309376274 18. 76422.7605478596 19. 76966.8875200054 20. 77157.7052517715 21. 76916.4791731862

[13]:
```
checkresiduals(forecast(arima_model, h = 21)) # check residuals from a time␣
↪series model
```

```
	Ljung-Box test

data:  Residuals from ARIMA(3,1,2)
Q* = 4.8954, df = 5, p-value = 0.4288

Model df: 5.   Total lags used: 10
```




```
[48]: unique(data$Articulo) # no. of articles in the dataframe
```

1. 'Art_01' 2. 'Art_02' 3. 'Art_03' 4. 'Art_04' 5. 'Art_05' 6. 'Art_06' 7. 'Art_07' 8. 'Art_08' 9. 'Art_09' 10. 'Art_10' 11. 'Art_11' 12. 'Art_12' 13. 'Art_13' 14. 'Art_14' 15. 'Art_15' 16. 'Art_16' 17. 'Art_17' 18. 'Art_18' 19. 'Art_19' 20. 'Art_20' 21. 'Art_21' 22. 'Art_22' 23. 'Art_23' 24. 'Art_24' 25. 'Art_25' 26. 'Art_26' 27. 'Art_27' 28. 'Art_28' 29. 'Art_29' 30. 'Art_30' 31. 'Art_31' 32. 'Art_32' 33. 'Art_33' 34. 'Art_34' 35. 'Art_35'

```
[49]: global_forecast <- rep(0, 21)

articles <-␣
 ↪c('Art_01','Art_02','Art_03','Art_04','Art_05','Art_06','Art_07','Art_08','Art_09',
```

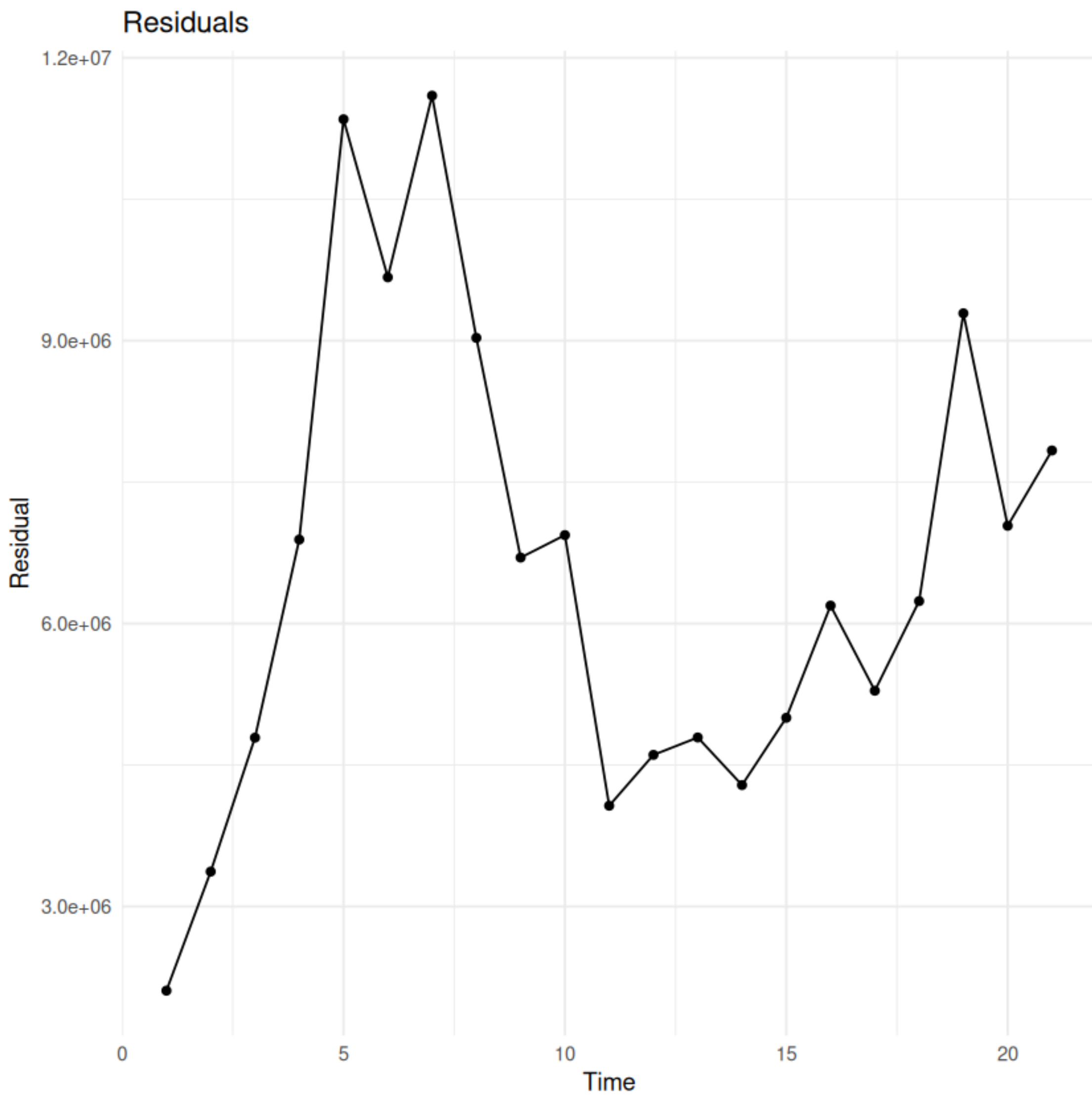
```
␣
↪'Art_10','Art_11','Art_12','Art_13','Art_14','Art_15','Art_16','Art_17','Art_18',
␣
↪'Art_19','Art_20','Art_21','Art_22','Art_23','Art_24','Art_25','Art_26','Art_27',
␣
↪'Art_28','Art_29','Art_30','Art_31','Art_32','Art_33','Art_34','Art_35') #␣
↪no. of articles

for (article in articles) {

    desagg <- data[data$Articulo == article, ] # filter data by product no.

    arima_model <- arima(desagg[, 3], order = c(1, 0, 1)) # fit an ARIMA model␣
↪on filtered data
    fcast <- forecast(arima_model, h = 21) # forecasting time series

    global_forecast <- global_forecast + as.numeric(fcast$mean) # global␣
↪forecast for all articles
}

print(global_forecast)
```

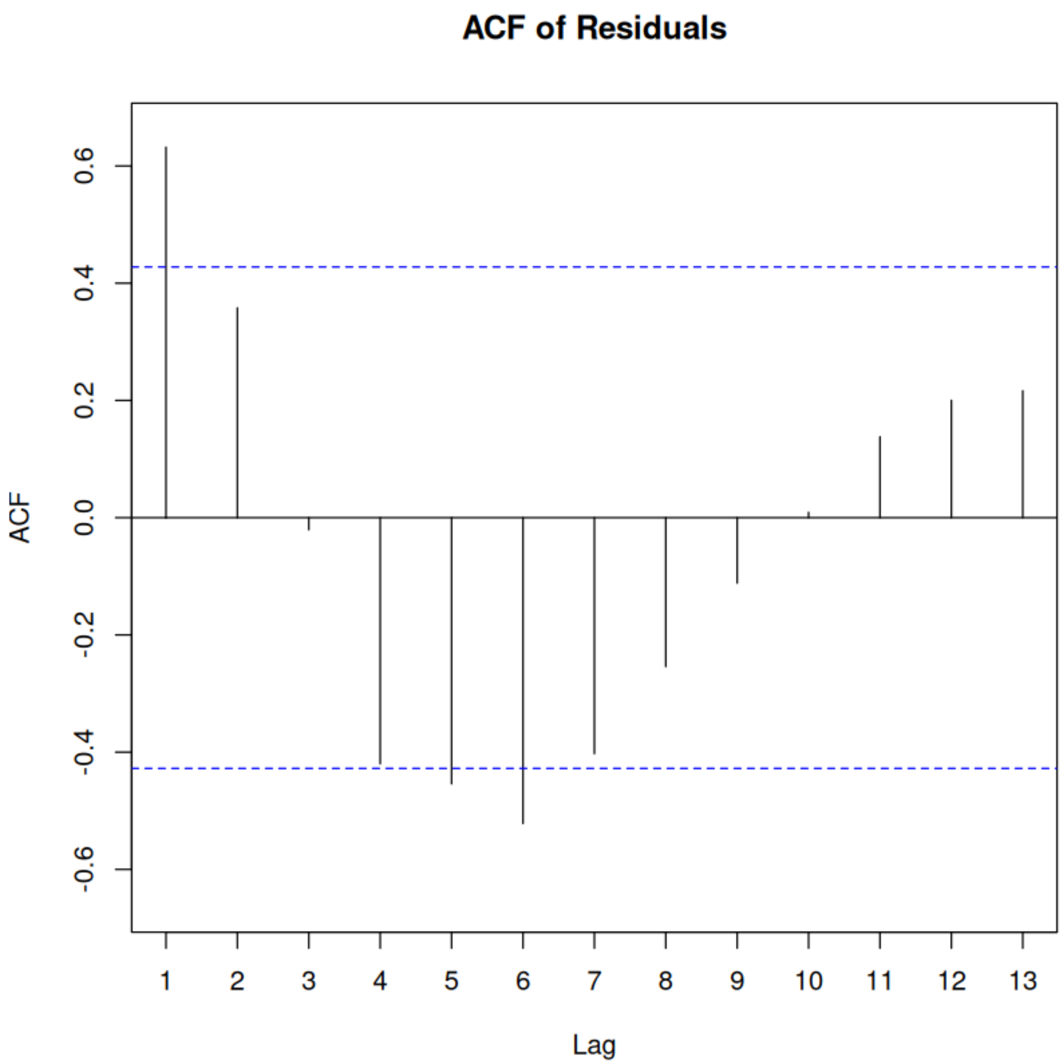
```
 [1] 2713036 2639722 2574975 2517738 2467088 2422218 2382424 2347090 2315677
[10] 2287714 2262789 2240543 2220660 2202864 2186913 2172593 2159720 2148129
[19] 2137677 2128237 2119697
```

Evaluación de los Residuos

[50]:
```
original_series <- c(
  4820380, 6009624, 7364992, 9408627, 13815278, 12094447, 13981023,
  11377165, 9015856, 9226113, 6331311, 6848393, 7012905, 6490397,
  7187925, 8363336, 7447890, 8386307, 11428137, 9165479, 9955325,
  7244257, 6100546, 7750861, 5139019, 5444571, 9448118, 9836013,
  10019467, 13969504, 5554454
) # original time series

fitted_values <- c(
  2713036, 2639722, 2574975, 2517738, 2467088, 2422218, 2382424,
  2347090, 2315677, 2287714, 2262789, 2240543, 2220660, 2202864,
  2186913, 2172593, 2159720, 2148129, 2137677, 2128237, 2119697
) # fitted values of disaggregate model

residuals <- original_series[1:length(fitted_values)] - fitted_values #␣
↪calculate residuals

ggplot(data.frame(Index = 1:length(residuals), Residuals = residuals), aes(x =␣
↪Index, y = Residuals)) +
  geom_line() + geom_point() + ggtitle("Residuals") +
```

```
  xlab("Time") + ylab("Residual") + theme_minimal() # residuals plot

Acf(residuals, main = "ACF of Residuals") # ACF plot
```





En el **gráfico ACF** se aprecian picos significativos en distintos intervalos. Con **valores de autocorrelación mayores** al modelo agregado, lo que podria sugerir un impacto en las predicciones del modelo.

0.3.4 Comparación de los Modelos

En este análisis se optó por aplicar ambos modelos, un modelo agregado y un modelo desagregado al conjunto de datos.

En el caso del **modelo agregado**, una de sus principales ventajas es que es más fácil de interpretar y manejar. Además, este enfoque permite identificar tendencias generales a nivel macro, como patrones estacionales o cambios en la demanda. Sin embargo, una de las desventajas del modelo agregado es la pérdida de detalle, ya que no permite explorar diferencias específicas entre artículos, lo cual es fundamental para tomar decisiones más precisas e informadas.

Por otro lado, el **modelo desagregado** ofrece la capacidad de identificar patrones, cambios o anomalías un nivel más profundo, permitiendo un análisis más detallado de productos específicos.

Para este conjunto de datos, el gráfico de ACF del modelo desagregado muestra un mayor número de picos en diferentes retrasos (lags) y presenta valores de autocorrelación más altos en comparación con el modelo agregado. Esto sugiere una mayor complejidad en la estructura de los datos desagregados. Por lo tanto, se puede concluir que el **modelo agregado es más adecuado** para realizar predicciones en este caso, ya que simplifica la autocorrelación y mejora la capacidad predictiva.

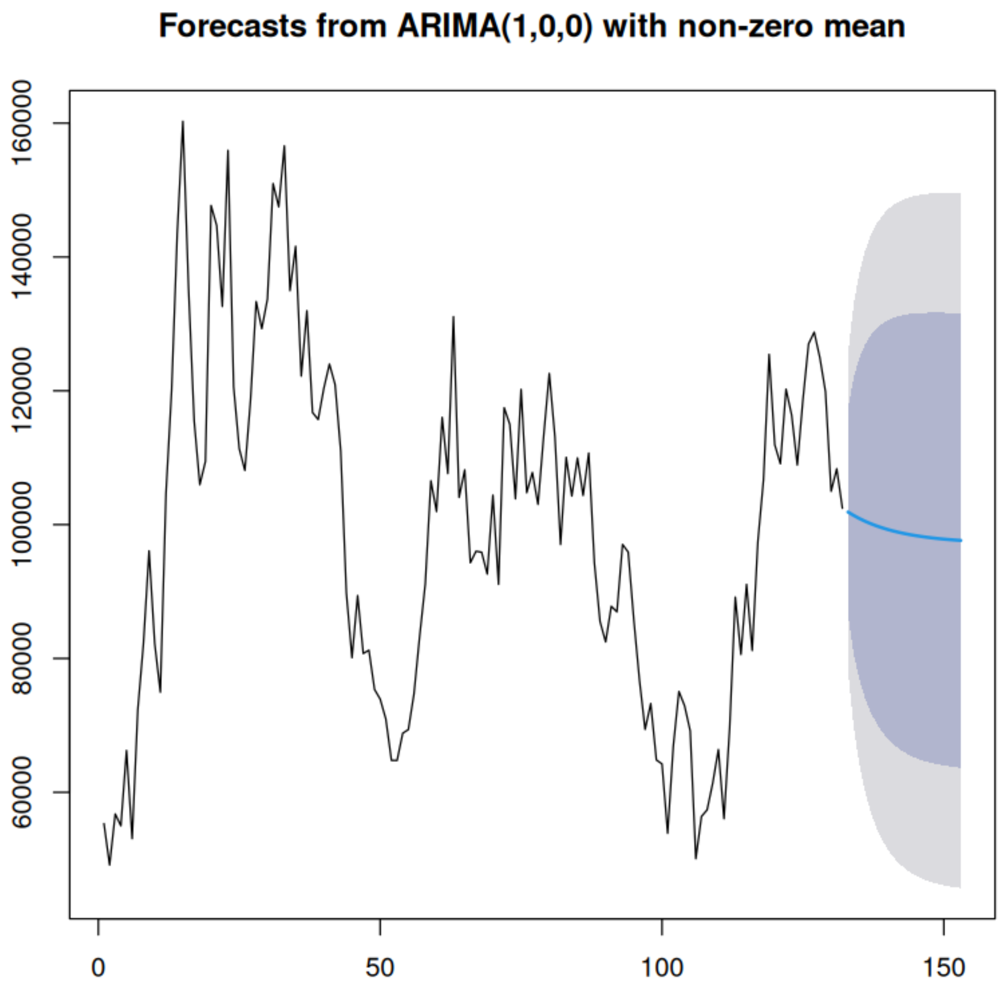
0.3.5 Conclusión

El **Modelo agregado** es útil cuando se buscan tendencias generales y se prioriza análisis más "simple", especialmente cuando la variabilidad a un nivel de detalle más profundo no es muy relevante. Por otro lado el **Modelo desagregado** es recomendable si se busca un análisis profundo que capte variaciones significativas entre productos específicos.

0.3.6 Anexo A - Modelos Ajustados por Producto

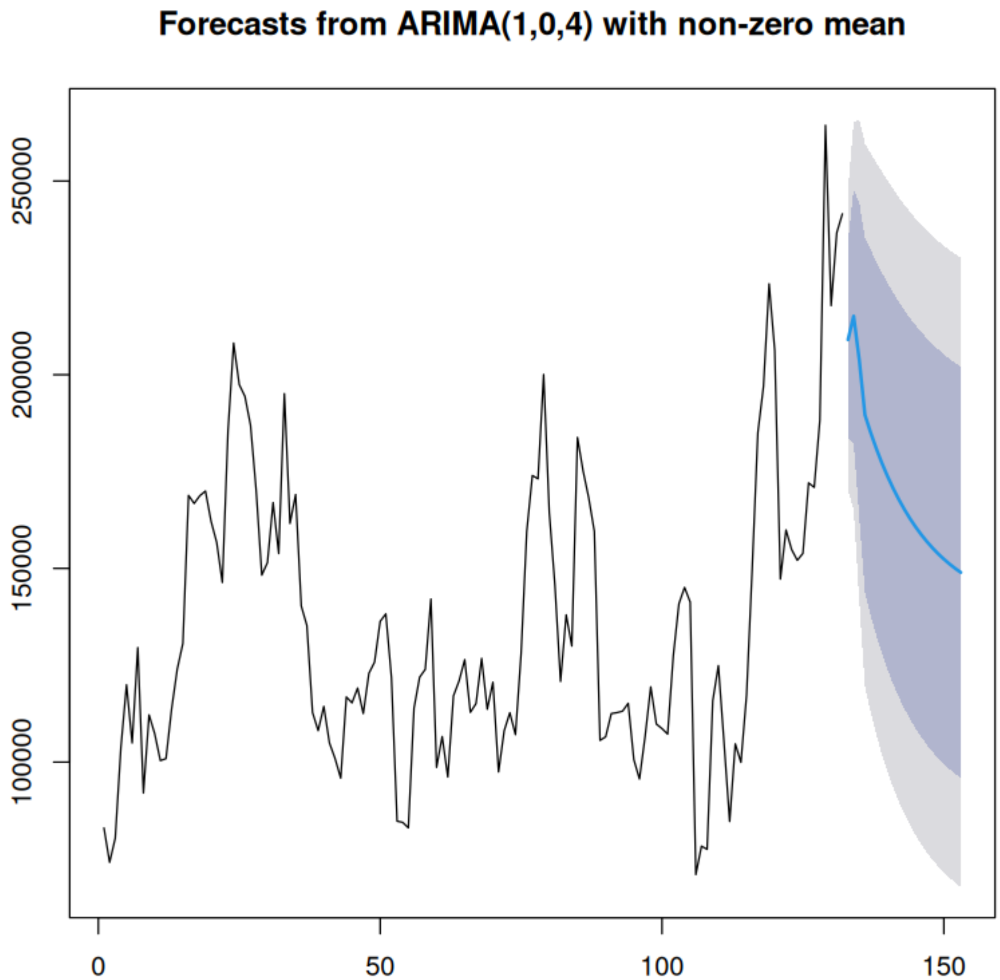
Product no. 2

[14]:
```
product_2 <- data[data$Articulo == "Art_02", ] # filter data by product no.
arima_model <- auto.arima(product_2[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



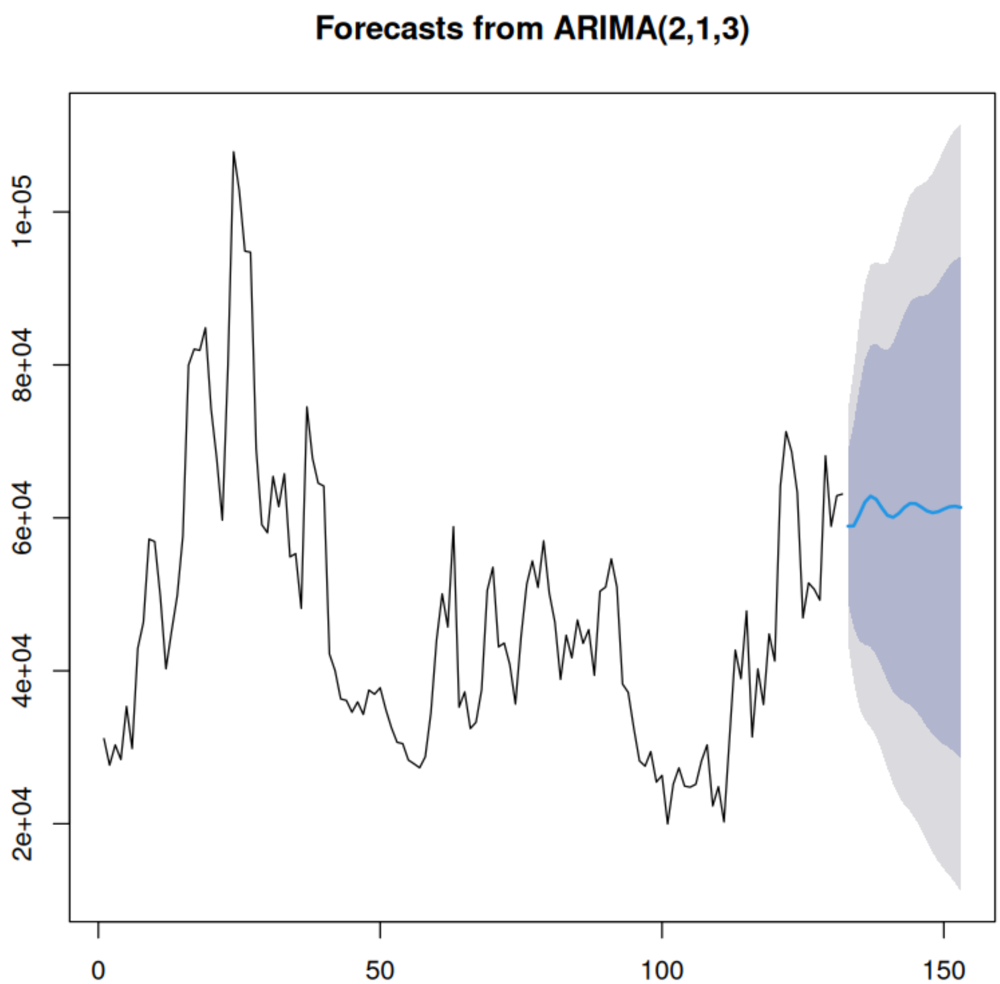
Product no. 3

[15]:
```
product_3 <- data[data$Articulo == "Art_03", ] # filter data by product no.
arima_model <- auto.arima(product_3[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



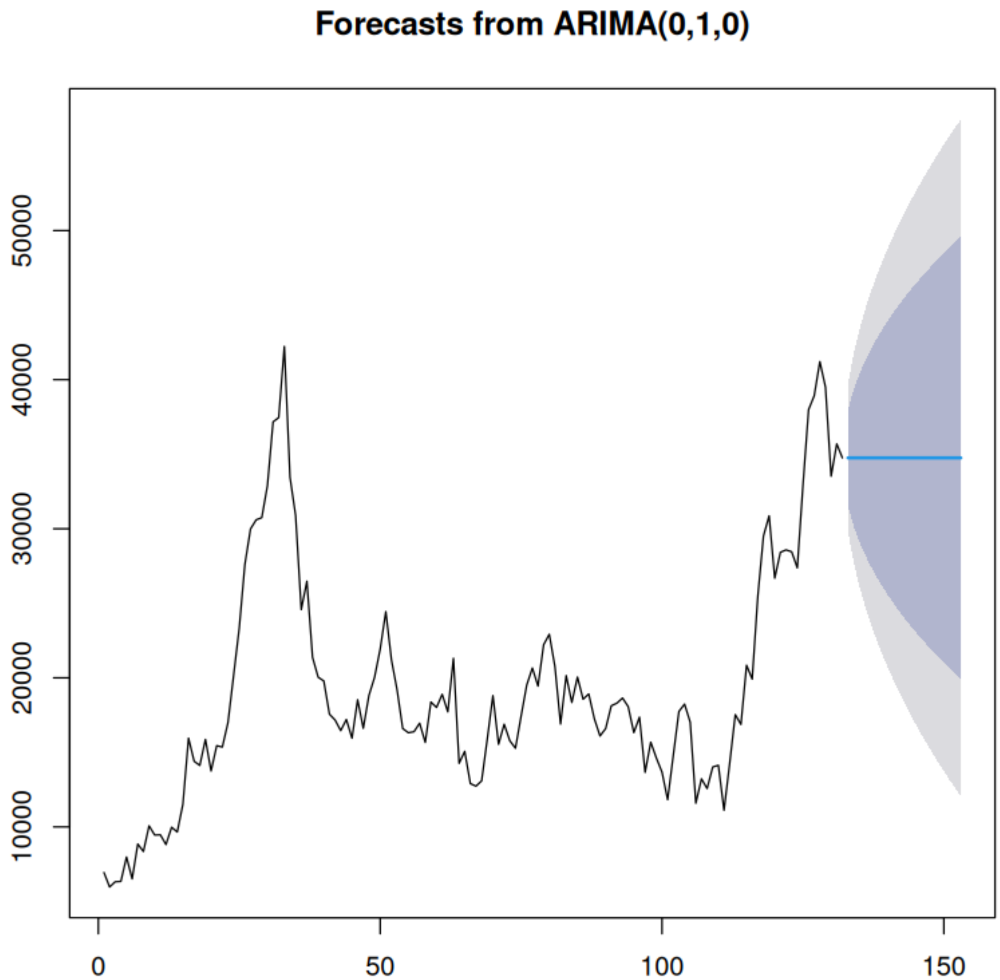
Product no. 4

[45]:
```
product_4 <- data[data$Articulo == "Art_04", ] # filter data by product no.
arima_model <- auto.arima(product_4[, 3]) # fit a model on aggregated data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



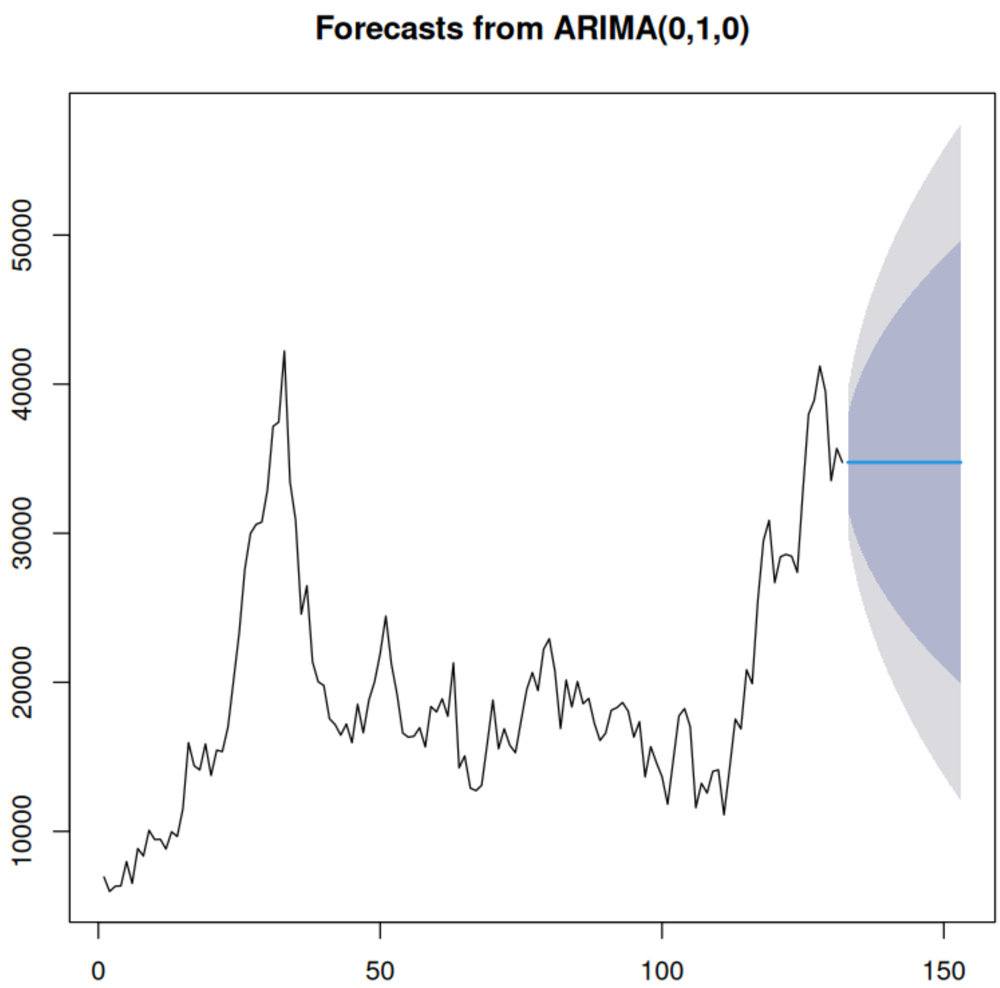
Product no. 5

[16]:
```
product_5 <- data[data$Articulo == "Art_05", ] # filter data by product no.
arima_model <- auto.arima(product_5[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



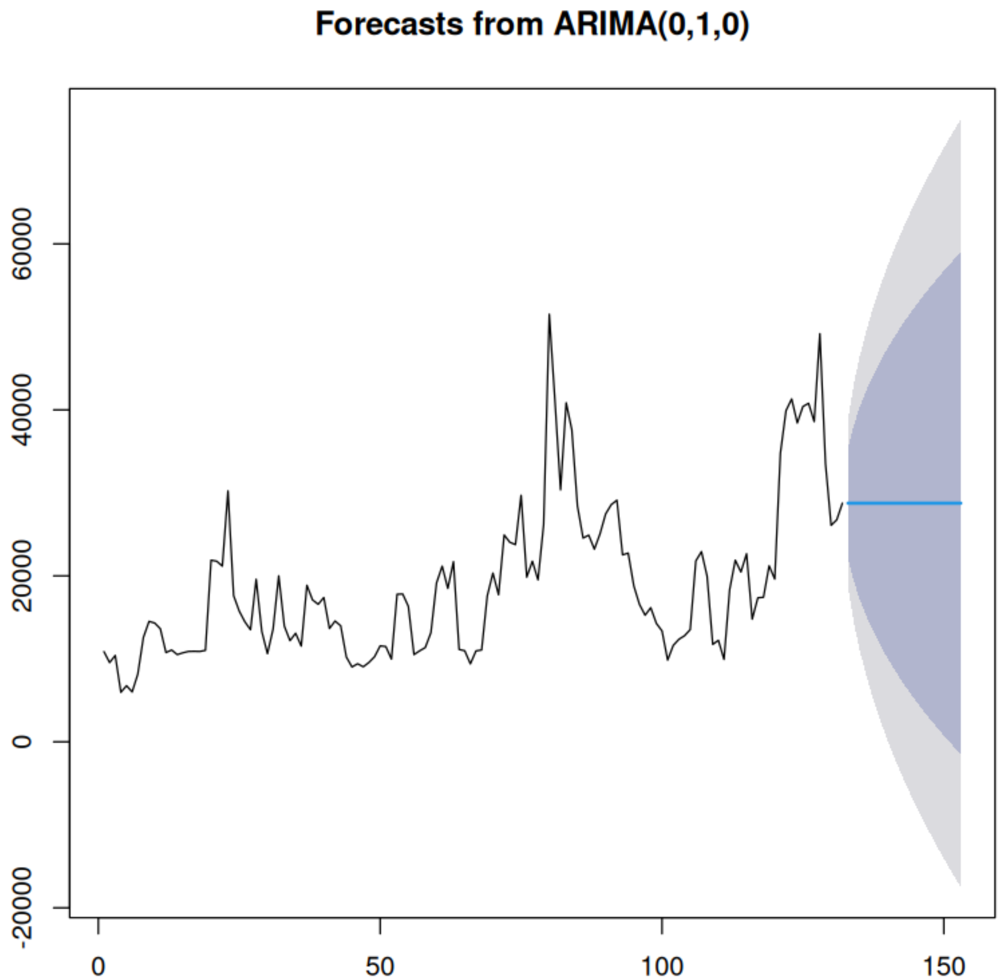
Product no. 5

[46]:
```
product_5 <- data[data$Articulo == "Art_05", ] # filter data by product no.
arima_model <- auto.arima(product_5[, 3]) # fit a model on aggregated data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



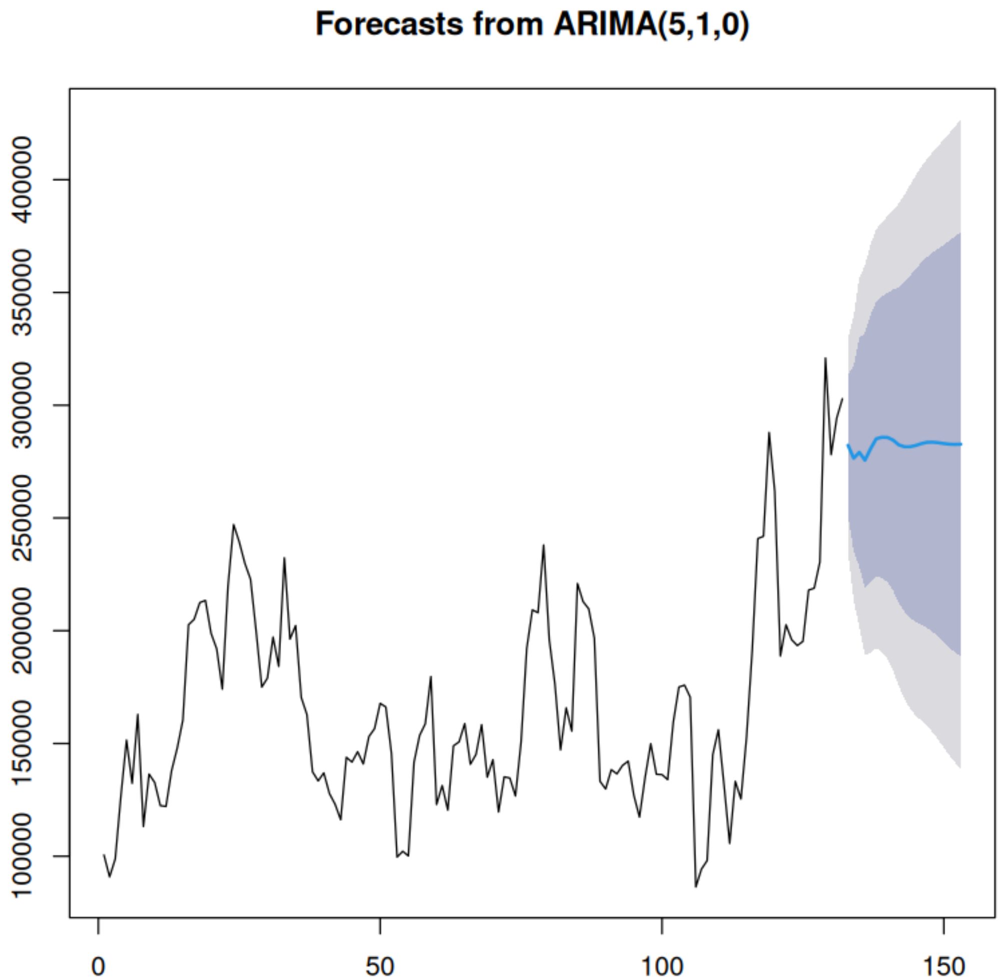
Product no. 6

[17]:
```
product_6 <- data[data$Articulo == "Art_06", ] # filter data by product no.
arima_model <- auto.arima(product_6[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



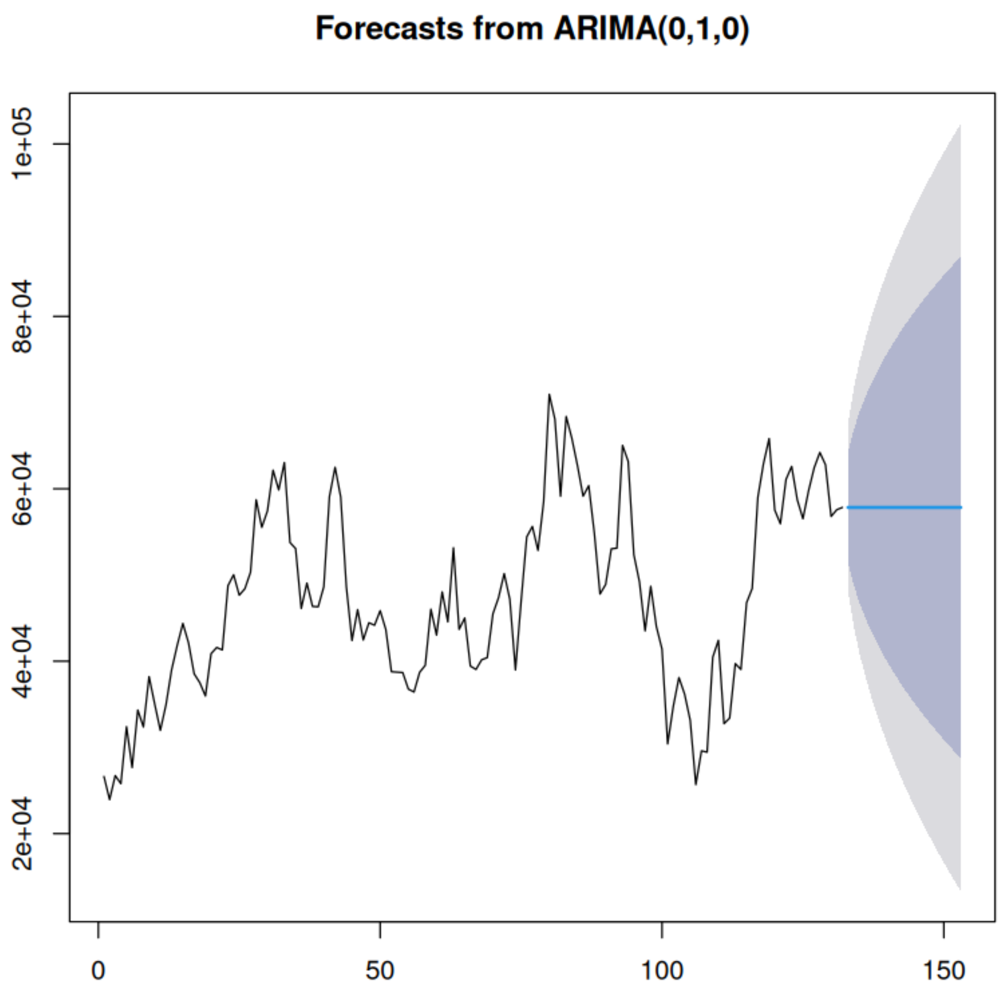
Product no. 7

```
[18]: product_7 <- data[data$Articulo == "Art_07", ] # filter data by product no.
      arima_model <- auto.arima(product_7[, 3]) # fit a model on filtered data
      fcast <- forecast(arima_model, h = 21) # forecasting time series
      plot(fcast)
```



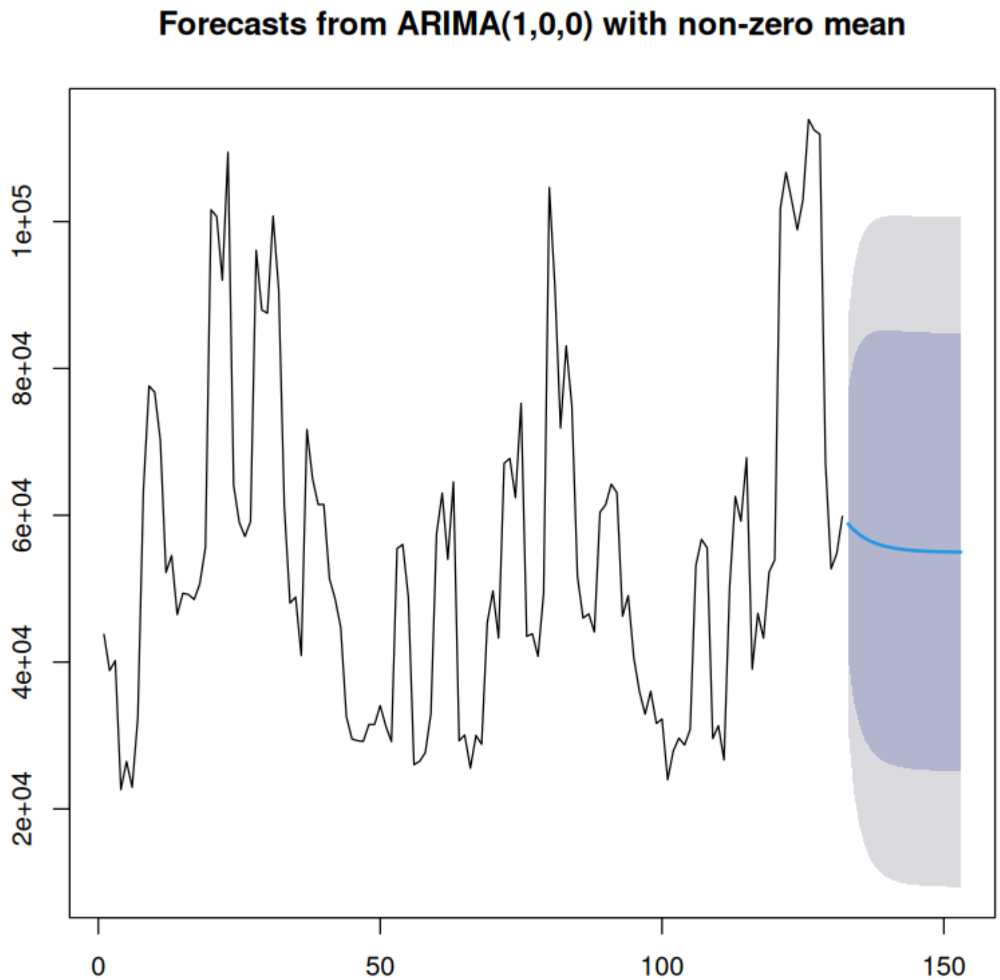
Product no. 8

[19]:
```
product_8 <- data[data$Articulo == "Art_08", ] # filter data by product no.
arima_model <- auto.arima(product_8[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```

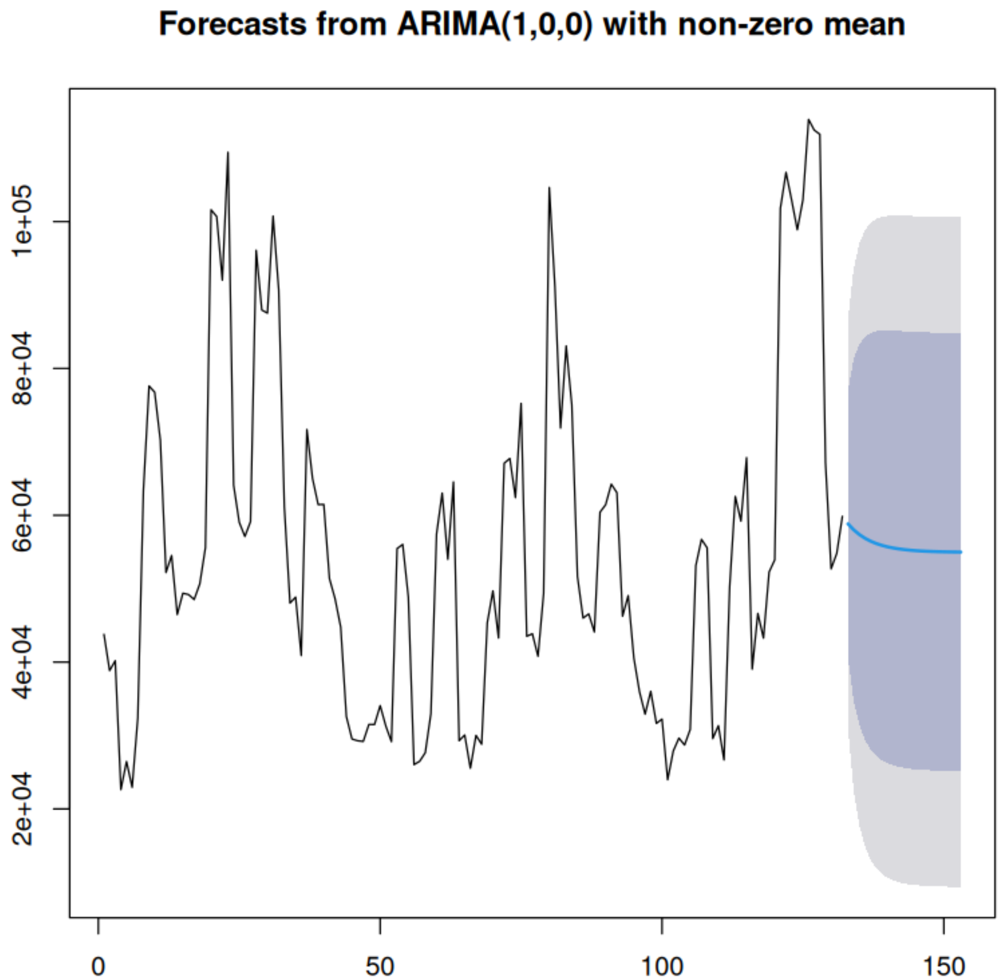


Product no. 9

[20]:
```
product_9 <- data[data$Articulo == "Art_09", ] # filter data by product no.
arima_model <- auto.arima(product_9[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```

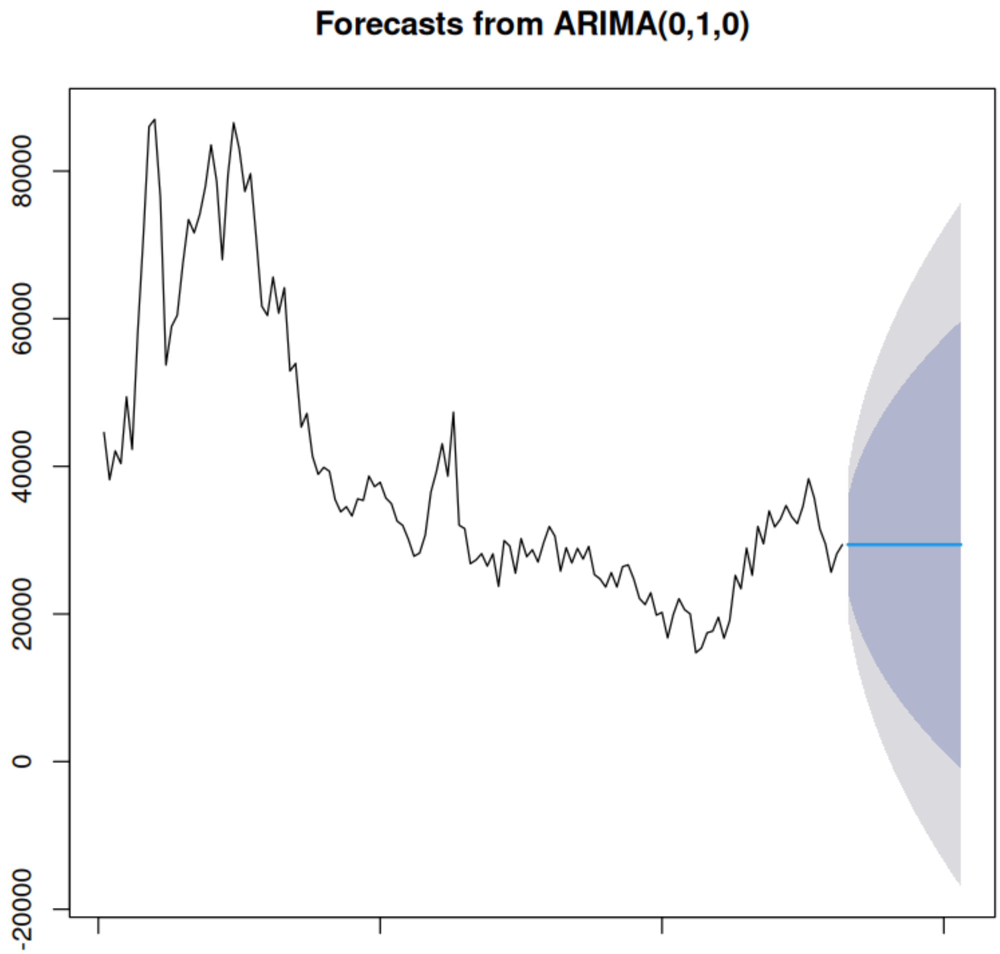


[21]:
```
product_9 <- data[data$Articulo == "Art_09", ] # filter data by product no.
arima_model <- auto.arima(product_9[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



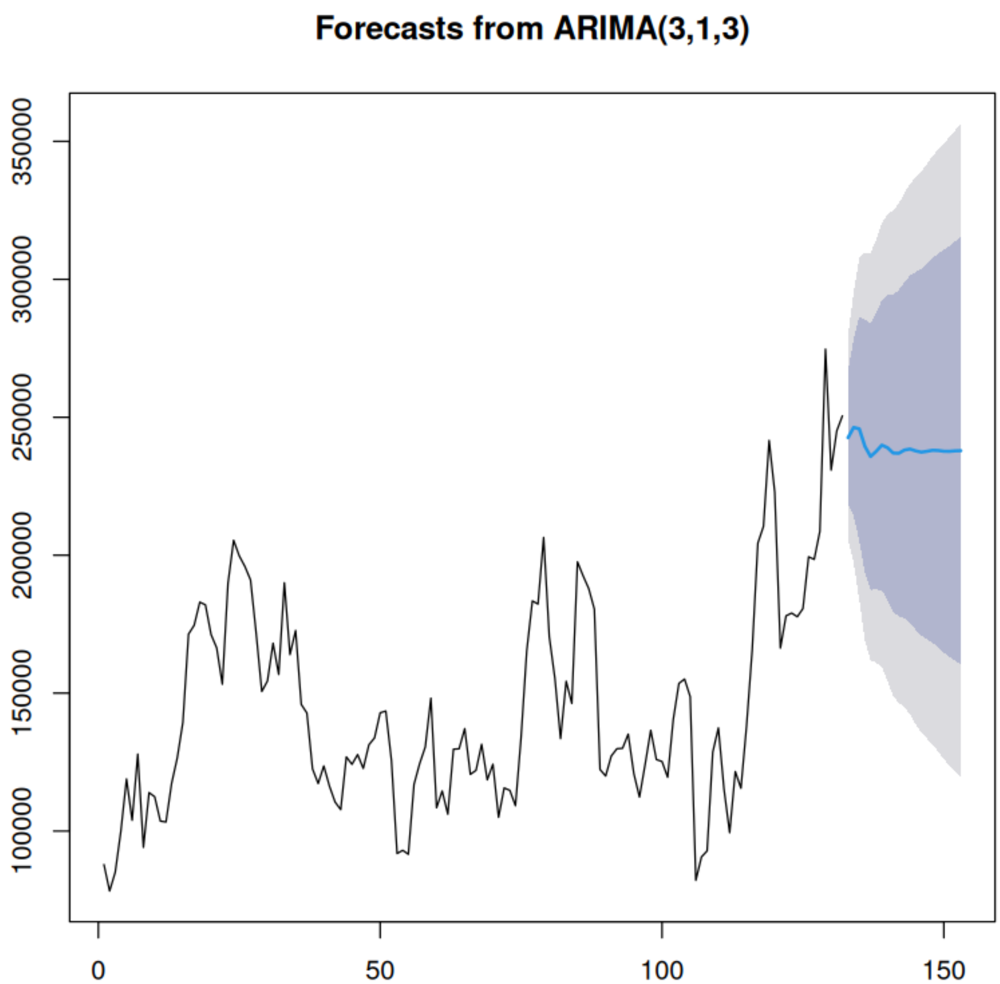
Product no. 10

[22]:
```r
product_10 <- data[data$Articulo == "Art_10", ] # filter data by product no.
arima_model <- auto.arima(product_10[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



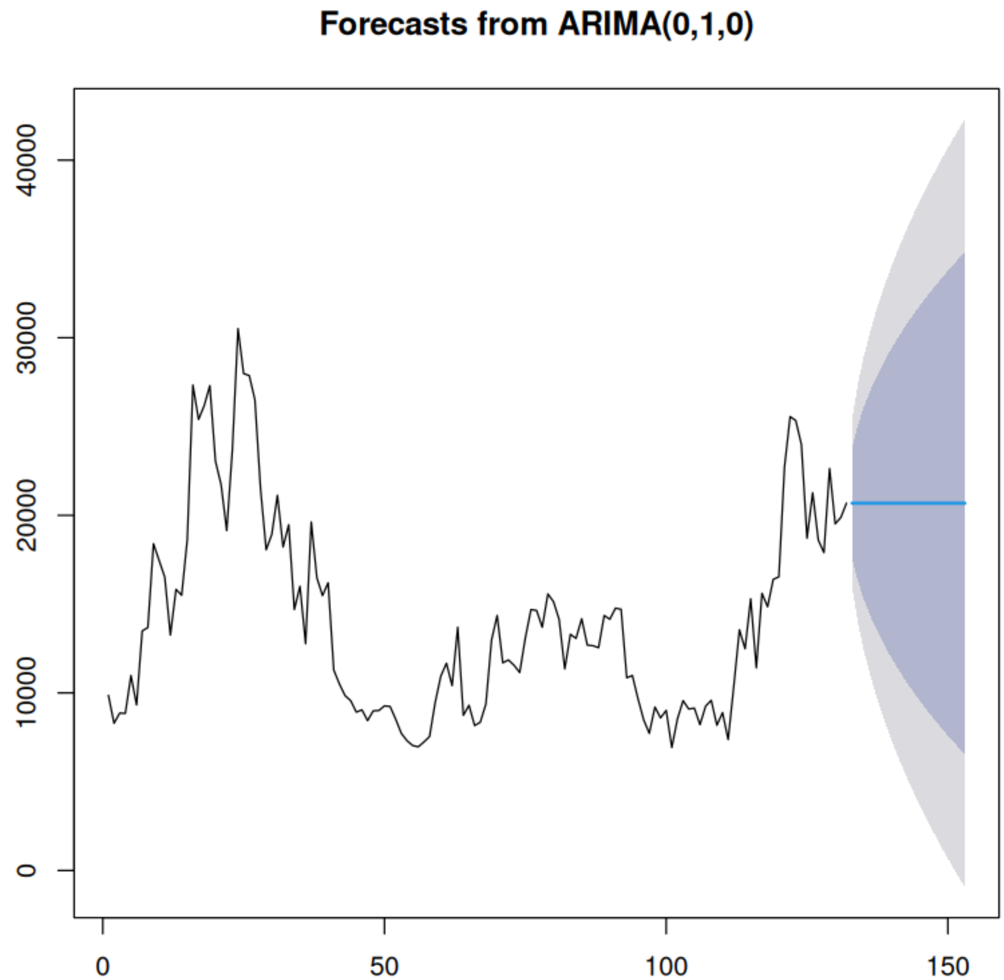
Product no. 11

[23]:
```
product_11 <- data[data$Articulo == "Art_11", ] # filter data by product no.
arima_model <- auto.arima(product_11[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



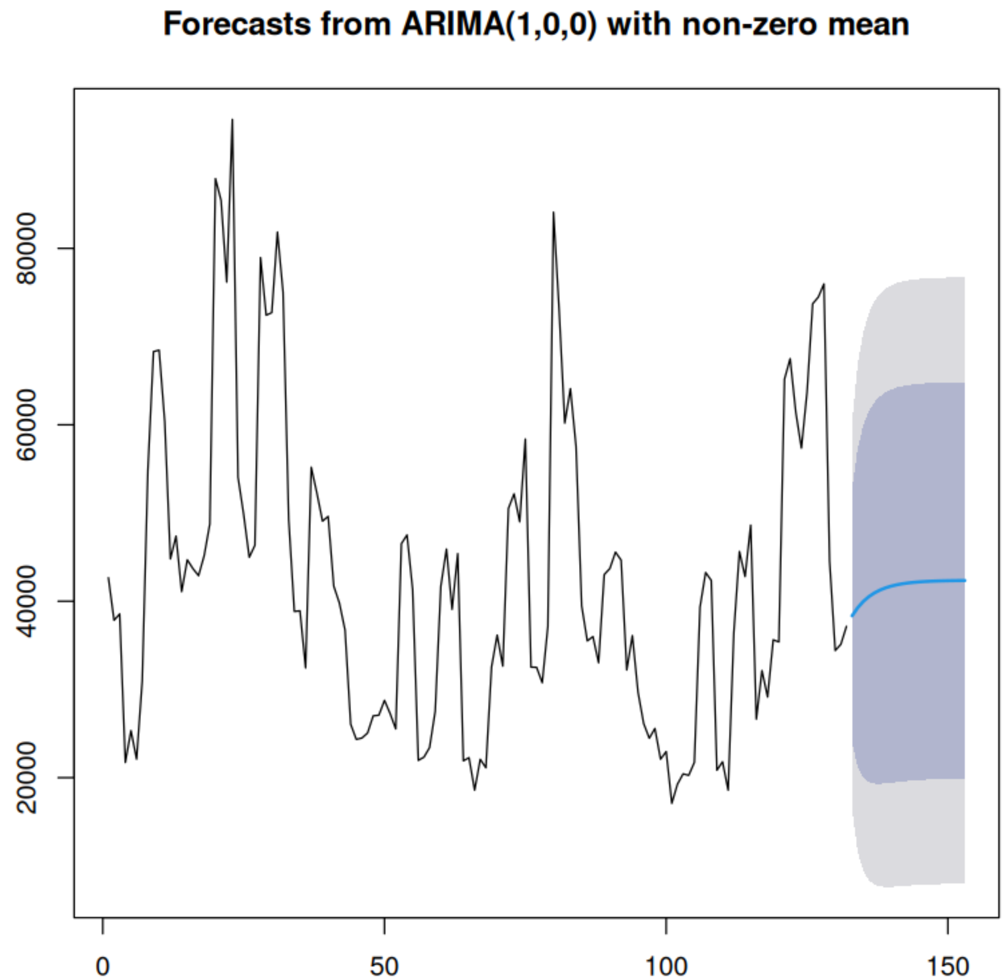
Product no. 12

[24]:
```
product_12 <- data[data$Articulo == "Art_12", ] # filter data by product no.
arima_model <- auto.arima(product_12[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



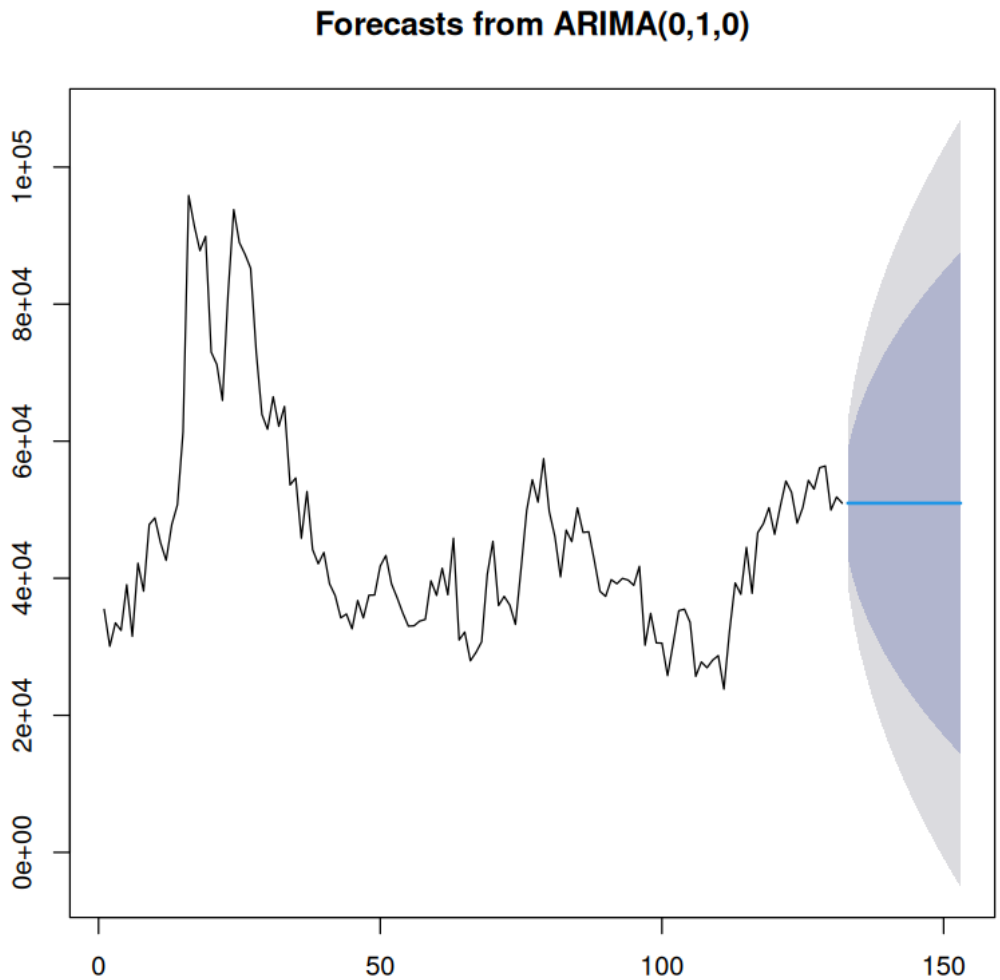
Product no. 13

[25]:
```
product_13 <- data[data$Articulo == "Art_13", ] # filter data by product no.
arima_model <- auto.arima(product_13[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



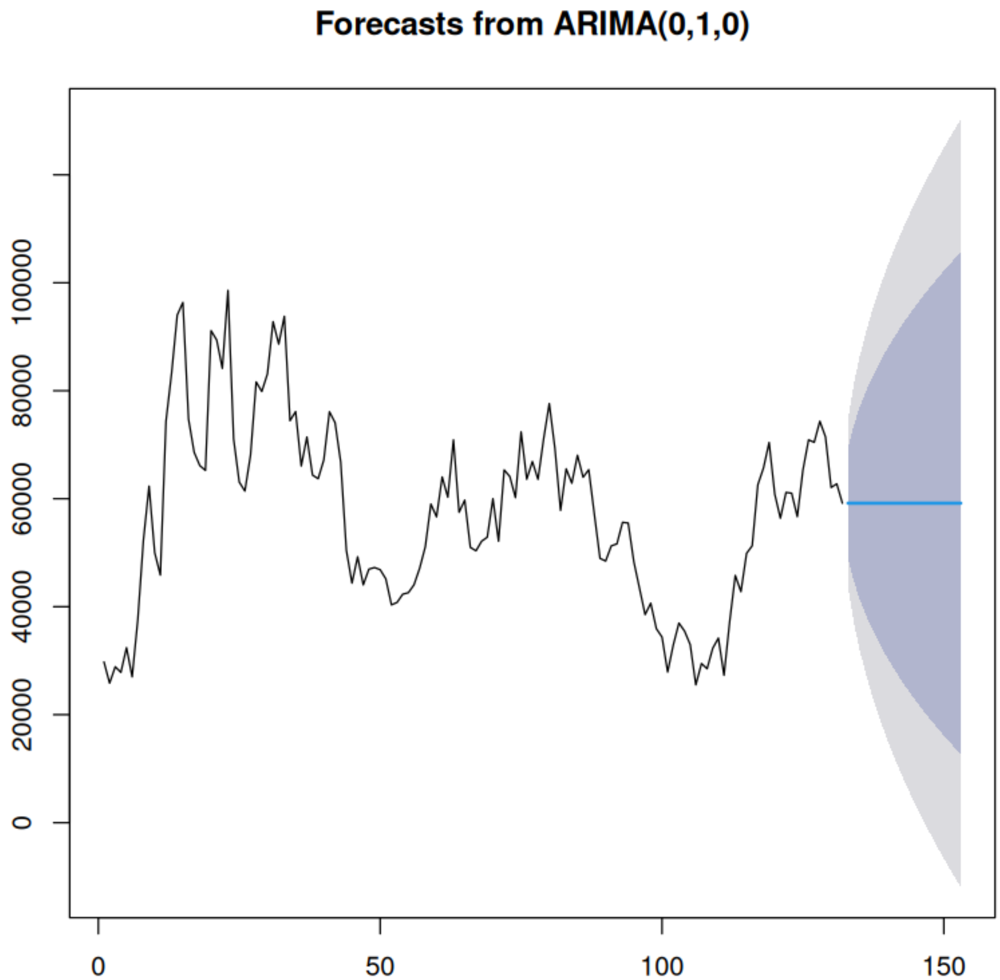
Product no. 14

[26]:
```
product_14 <- data[data$Articulo == "Art_14", ] # filter data by product no.
arima_model <- auto.arima(product_14[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



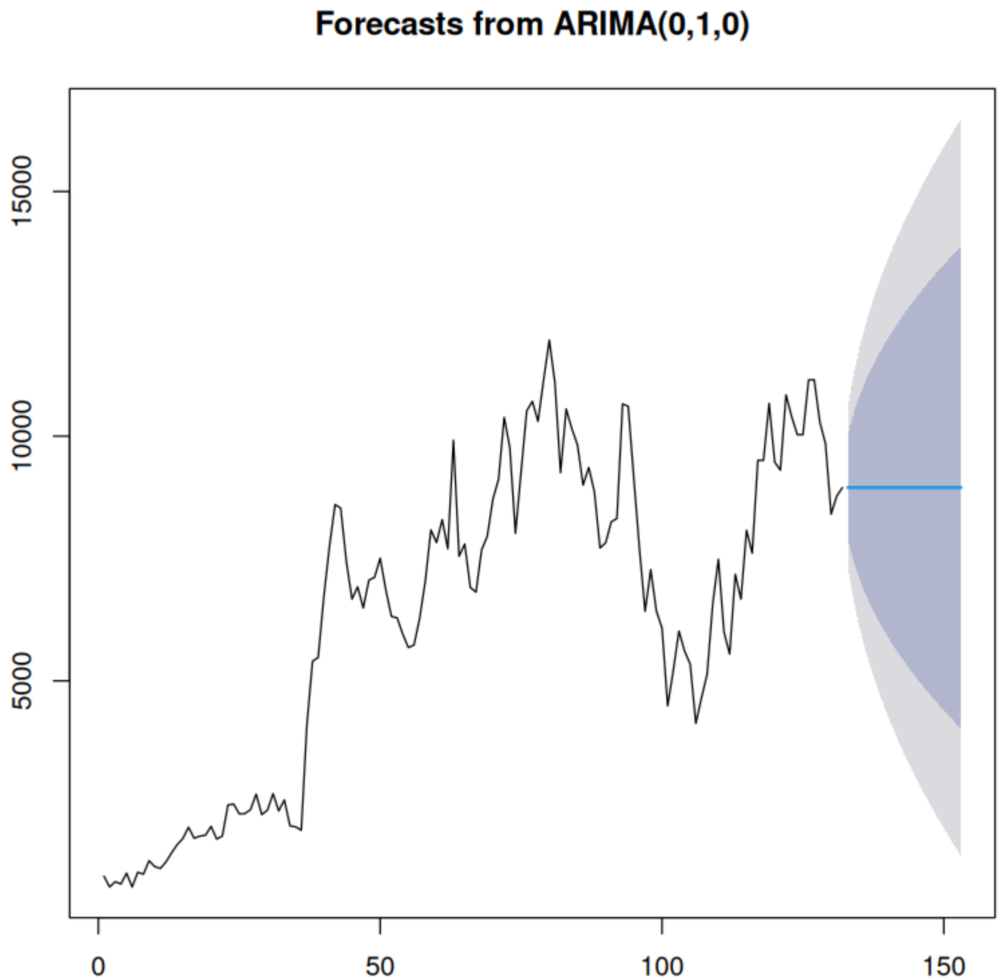
Product no. 15

[27]:
```
product_15 <- data[data$Articulo == "Art_15", ] # filter data by product no.
arima_model <- auto.arima(product_15[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



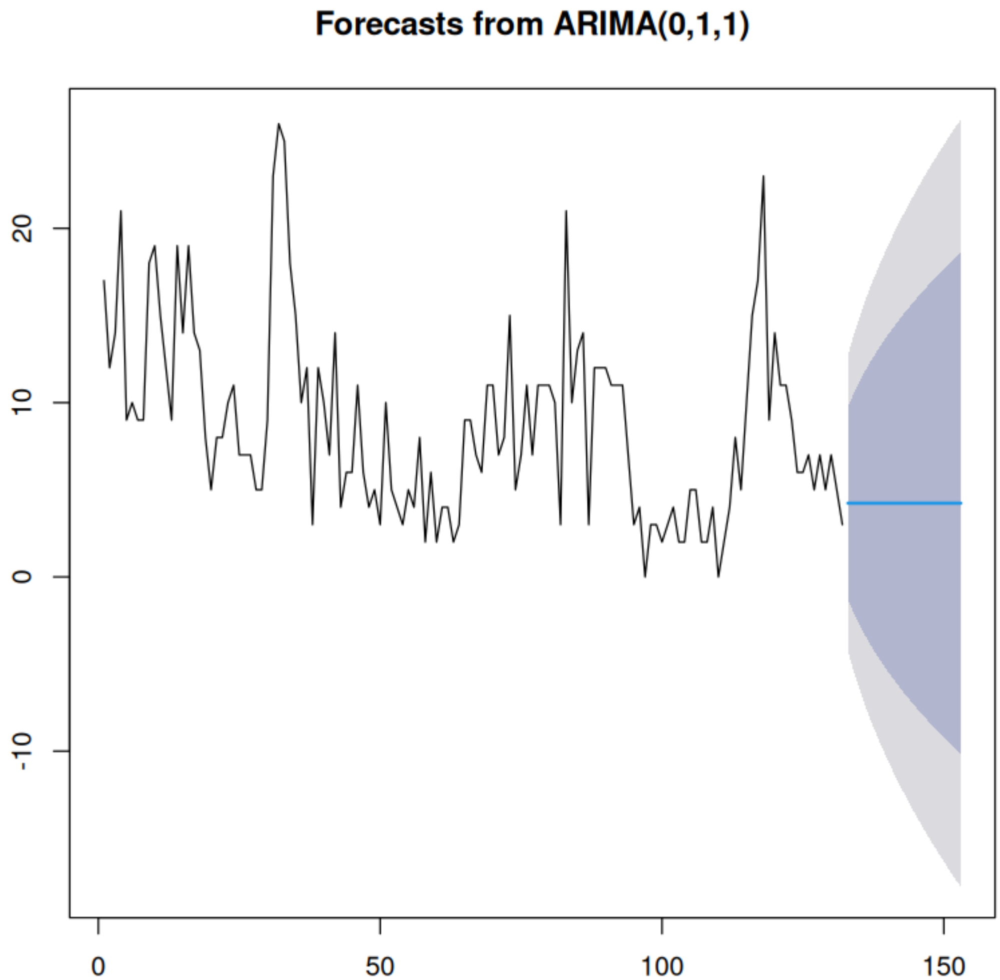
Product no. 16

[28]:
```
product_16 <- data[data$Articulo == "Art_16", ] # filter data by product no.
arima_model <- auto.arima(product_16[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



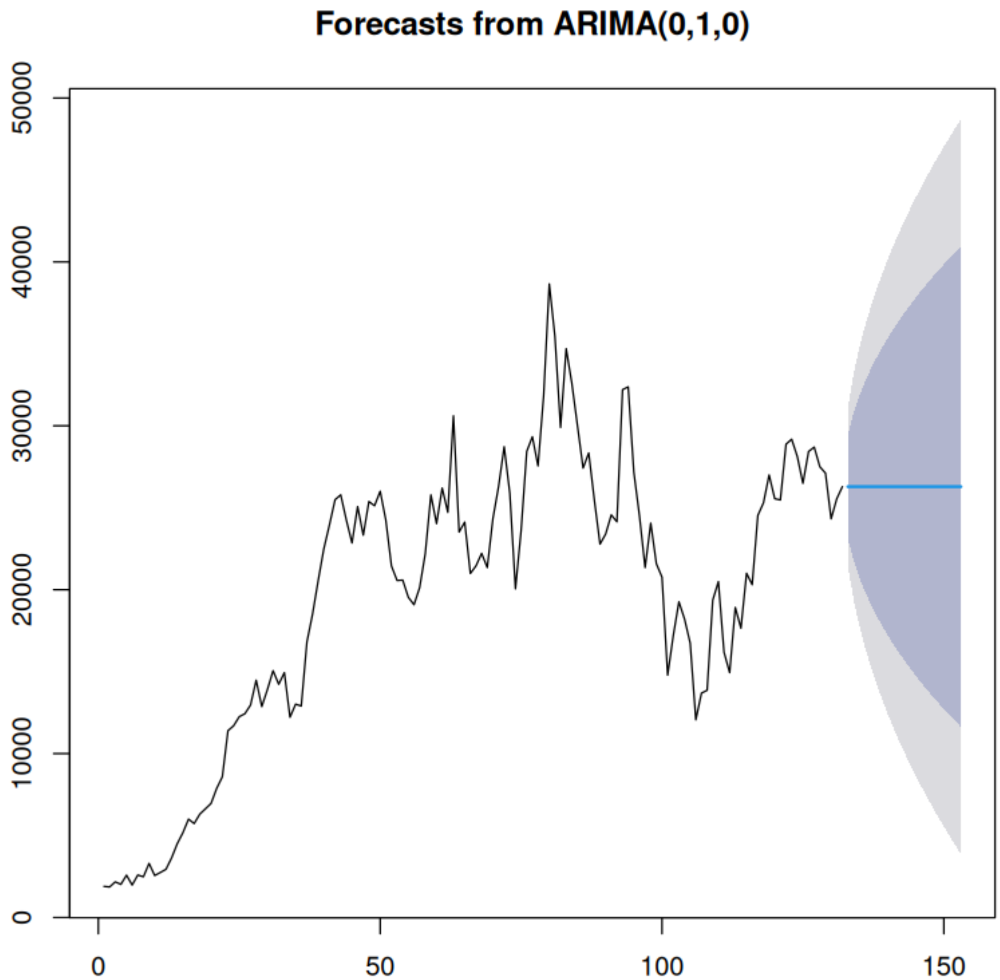
Product no. 17

[29]:
```
product_17 <- data[data$Articulo == "Art_17", ] # filter data by product no.
arima_model <- auto.arima(product_17[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



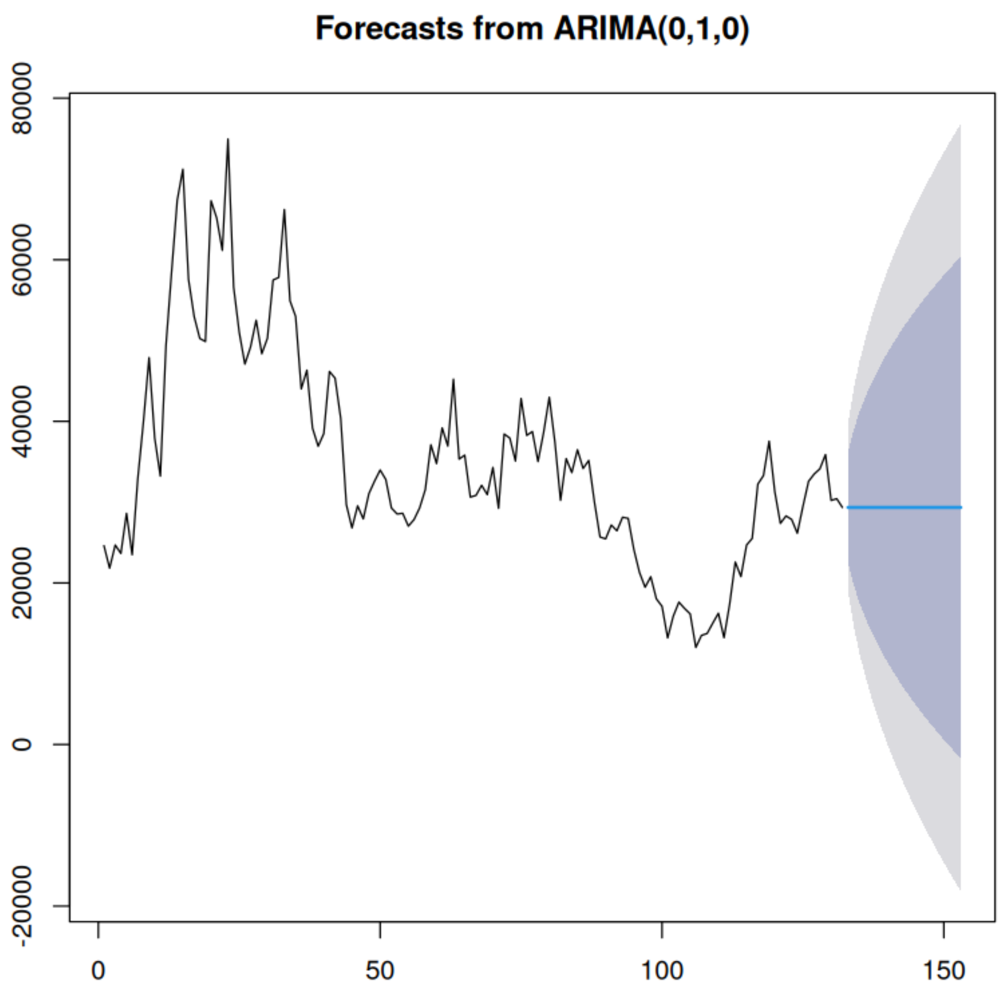
Product no. 18

```
[30]: product_18 <- data[data$Articulo == "Art_18", ] # filter data by product no.
arima_model <- auto.arima(product_18[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



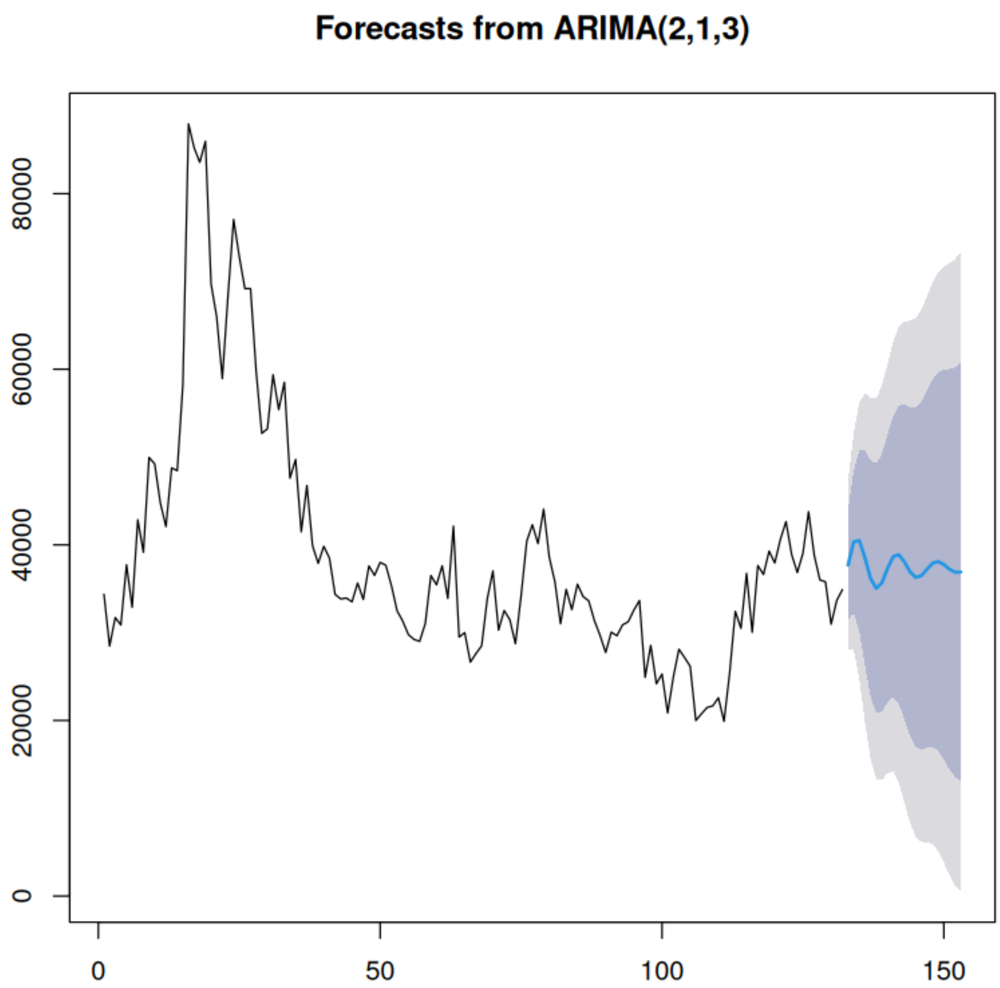
Product no. 19

[31]:
```
product_19 <- data[data$Articulo == "Art_19", ] # filter data by product no.
arima_model <- auto.arima(product_19[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



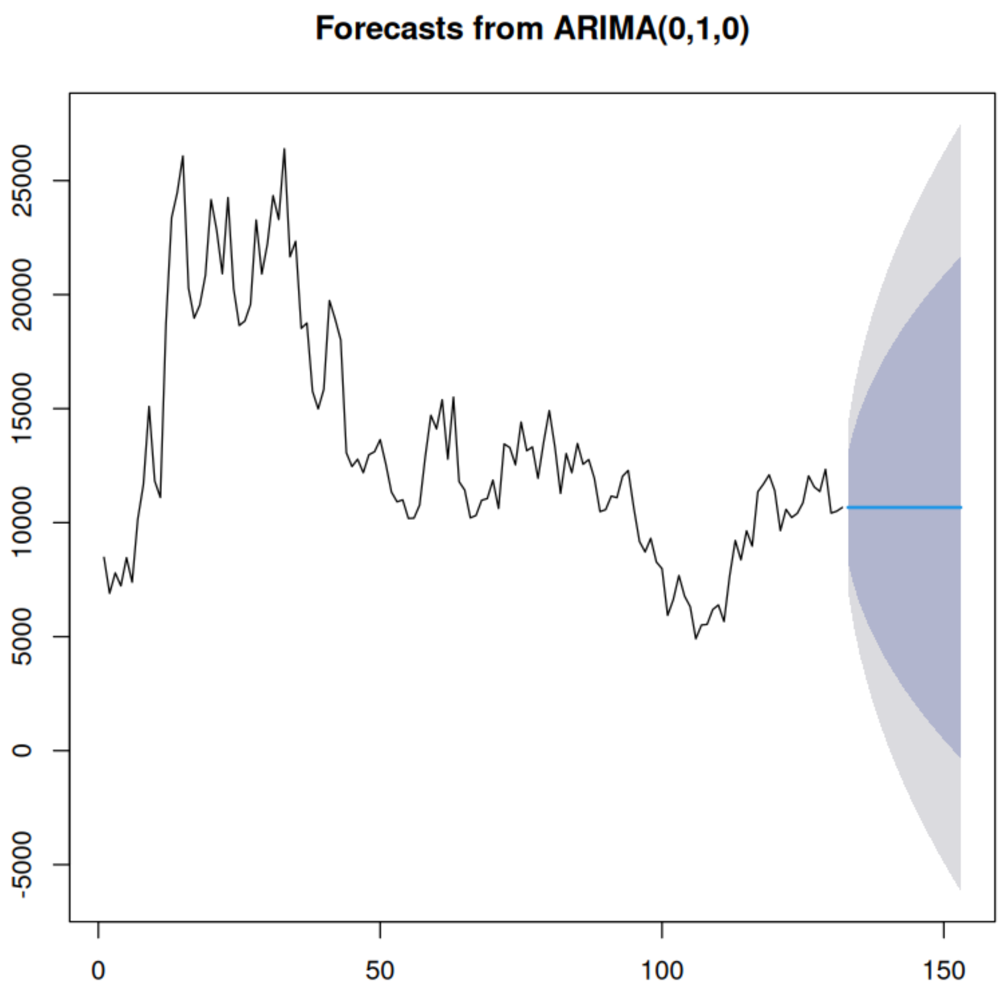
Product no. 20

[32]:
```
product_20 <- data[data$Articulo == "Art_20", ] # filter data by product no.
arima_model <- auto.arima(product_20[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



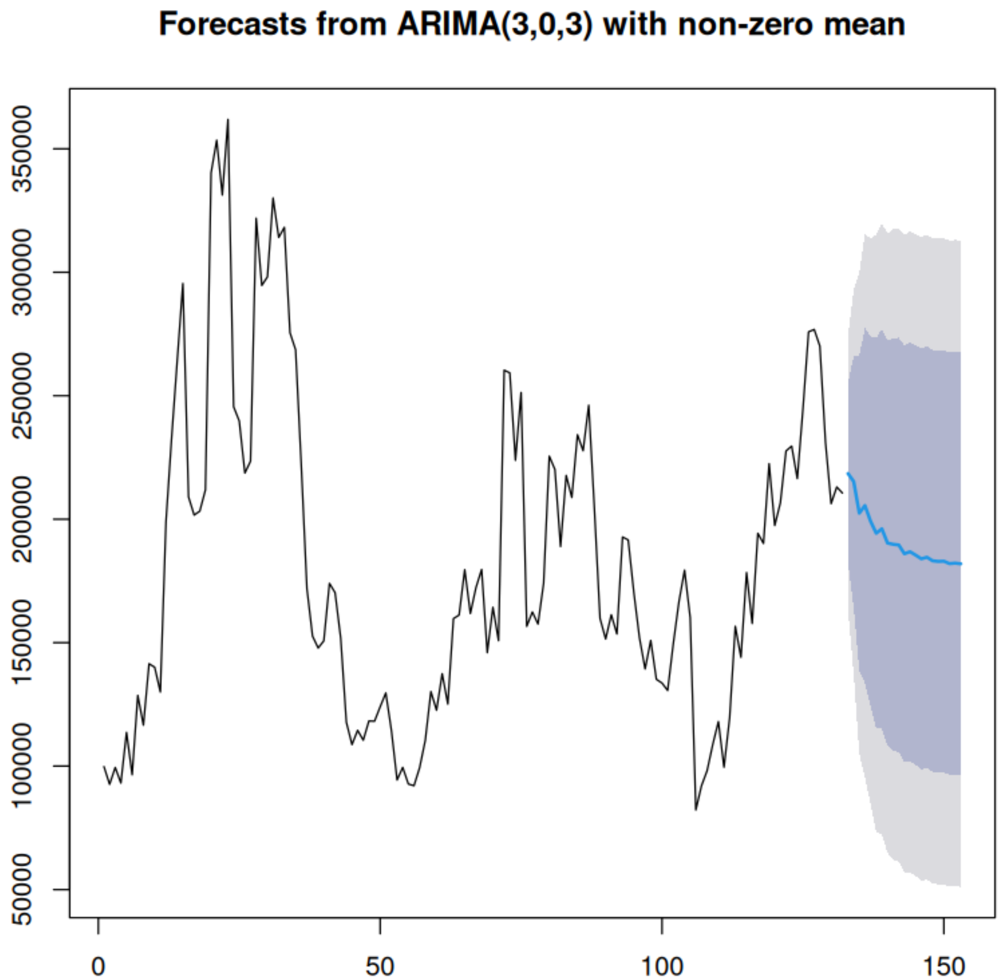
Product no. 21

[33]:
```
product_21 <- data[data$Articulo == "Art_21", ] # filter data by product no.
arima_model <- auto.arima(product_21[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



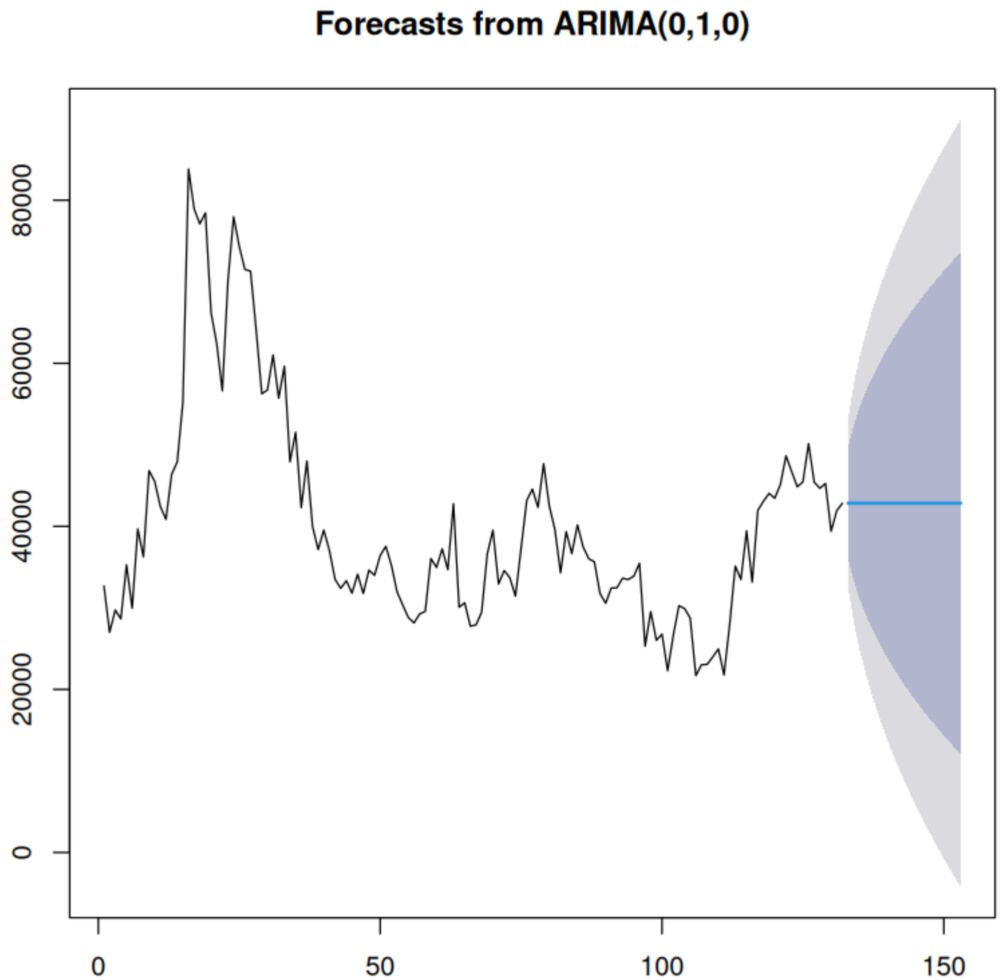
Product no. 22

[34]:
```
product_22 <- data[data$Articulo == "Art_22", ] # filter data by product no.
arima_model <- auto.arima(product_22[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



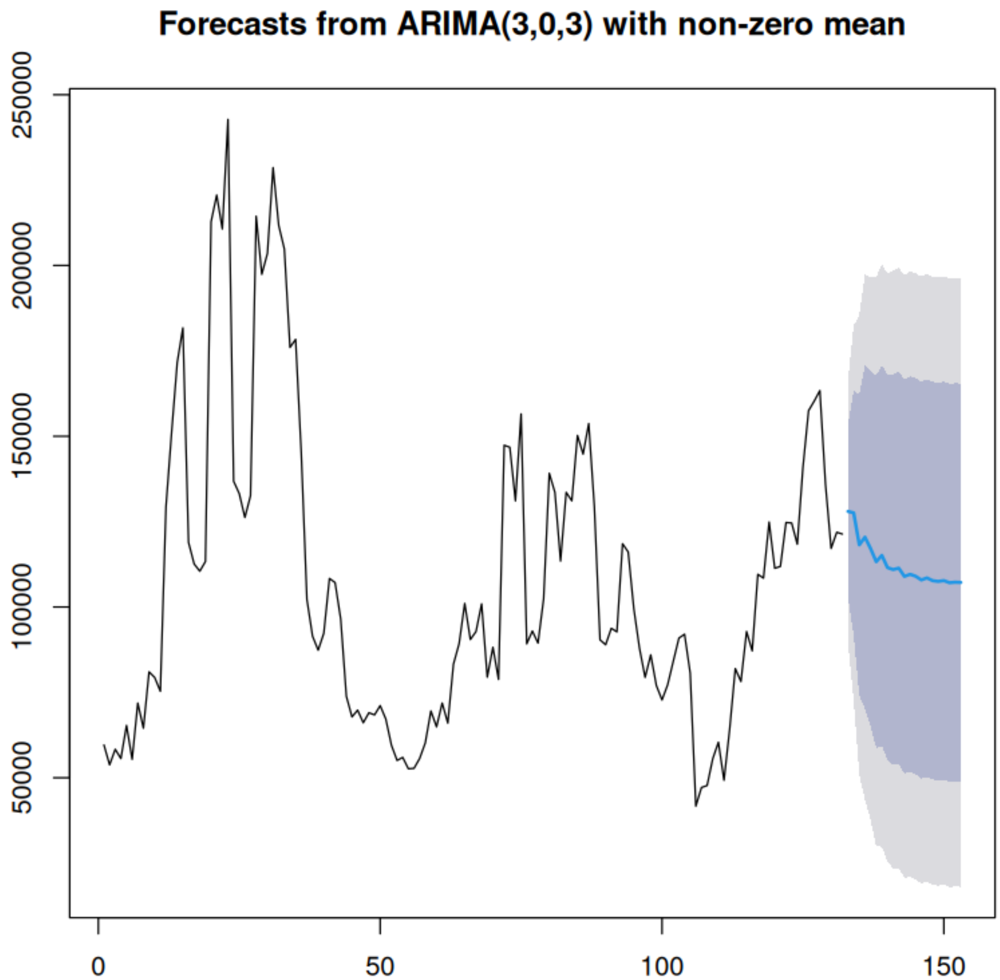
Product no. 23

[35]:
```
product_23 <- data[data$Articulo == "Art_23", ] # filter data by product no.
arima_model <- auto.arima(product_23[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



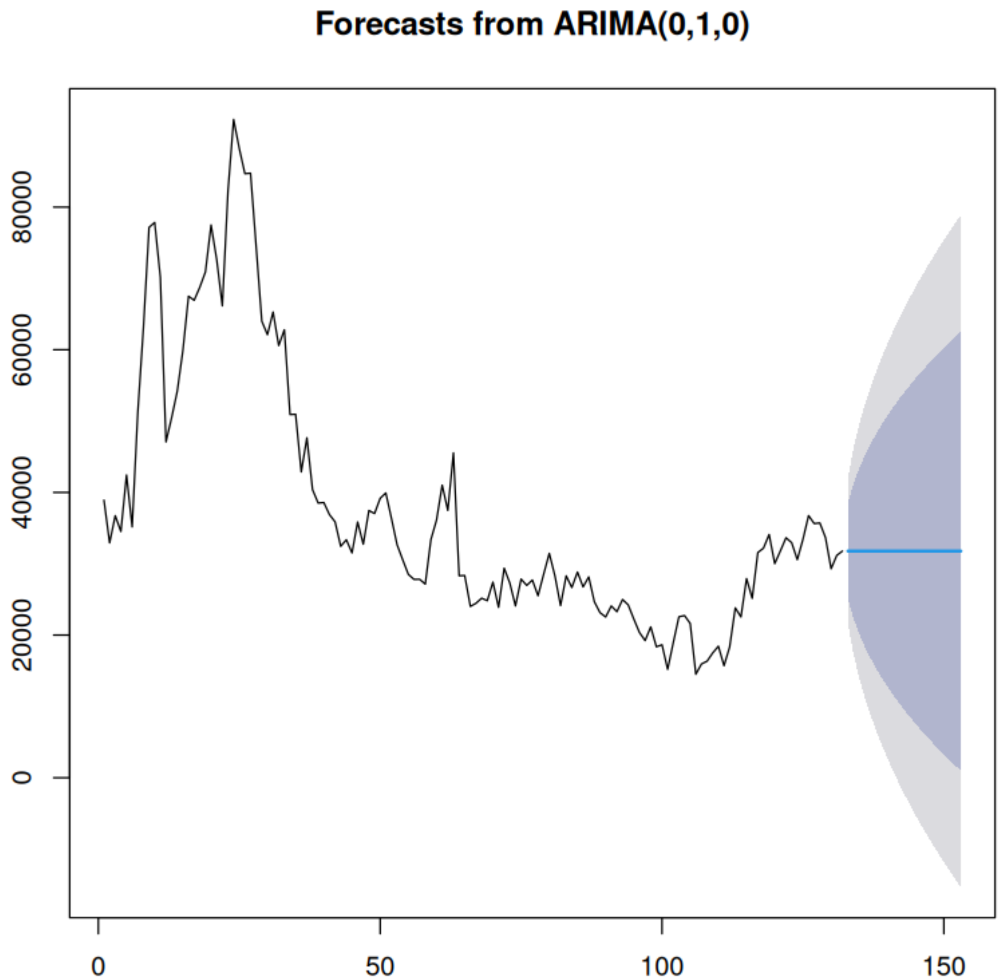
Product no. 24

[36]:
```
product_24 <- data[data$Articulo == "Art_24", ] # filter data by product no.
arima_model <- auto.arima(product_24[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



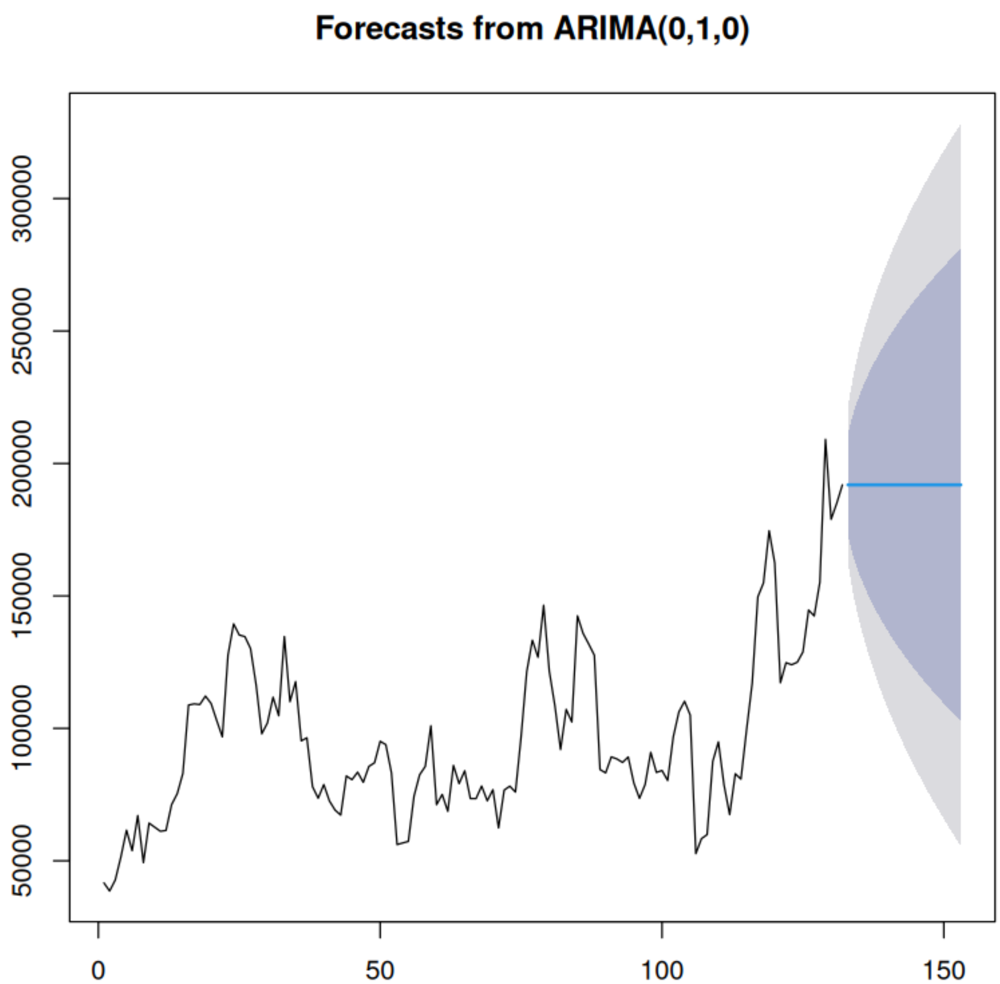
Product no. 25

[37]:
```
product_25 <- data[data$Articulo == "Art_25", ] # filter data by product no.
arima_model <- auto.arima(product_25[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



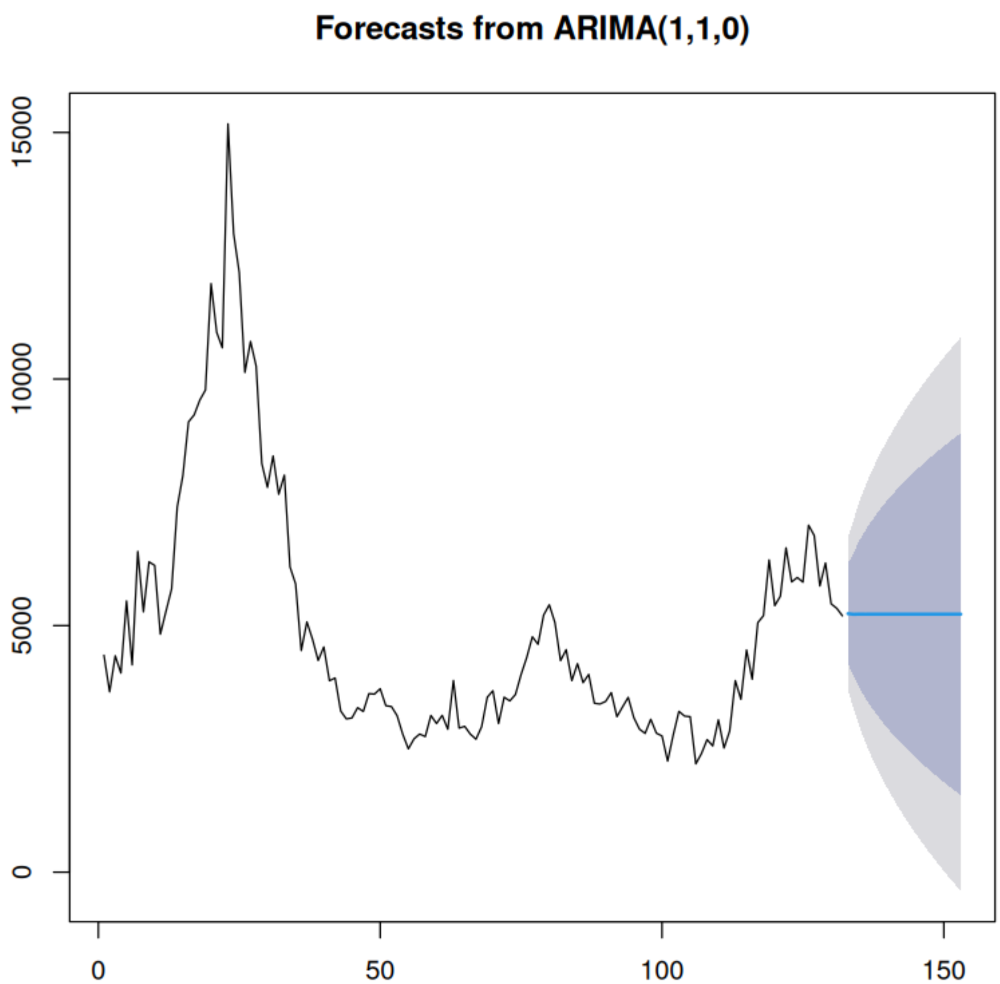
Product no. 26

[38]:
```
product_26 <- data[data$Articulo == "Art_26", ] # filter data by product no.
arima_model <- auto.arima(product_26[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



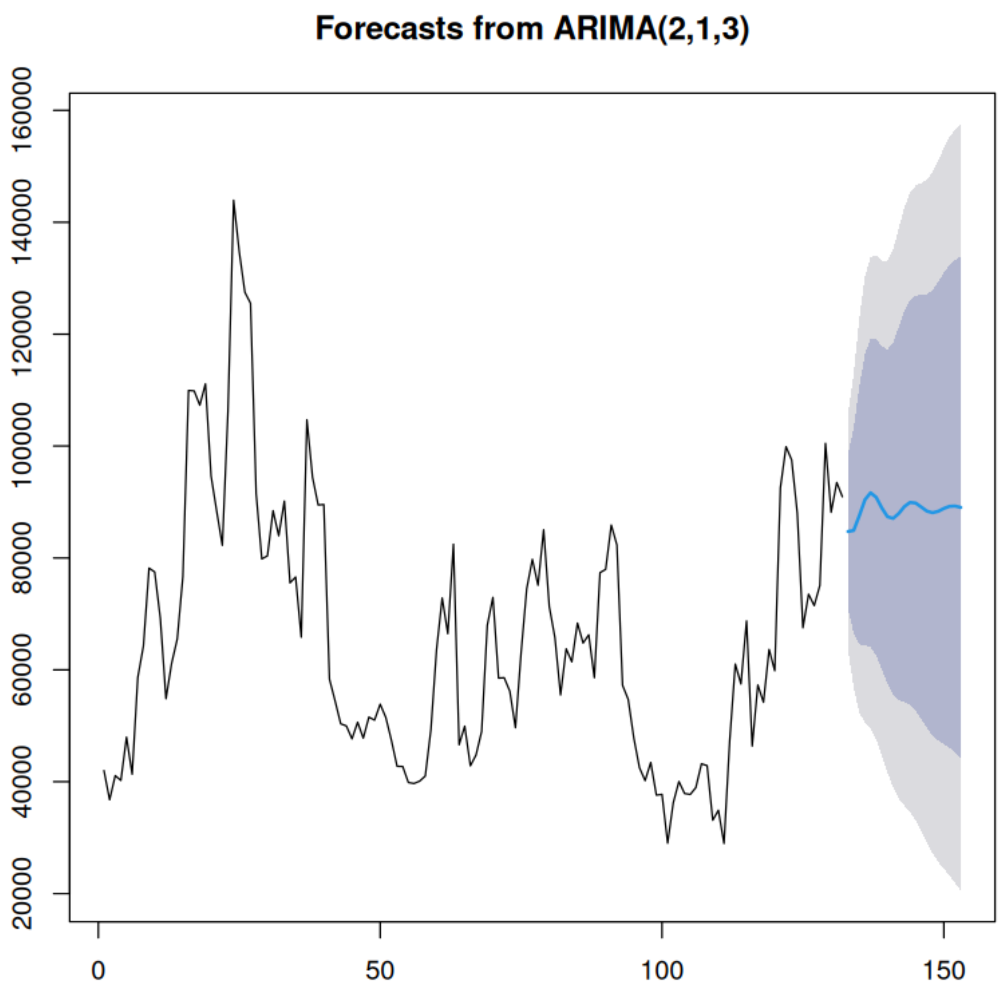
Product no. 27

```
[39]: product_27 <- data[data$Articulo == "Art_27", ] # filter data by product no.
arima_model <- auto.arima(product_27[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



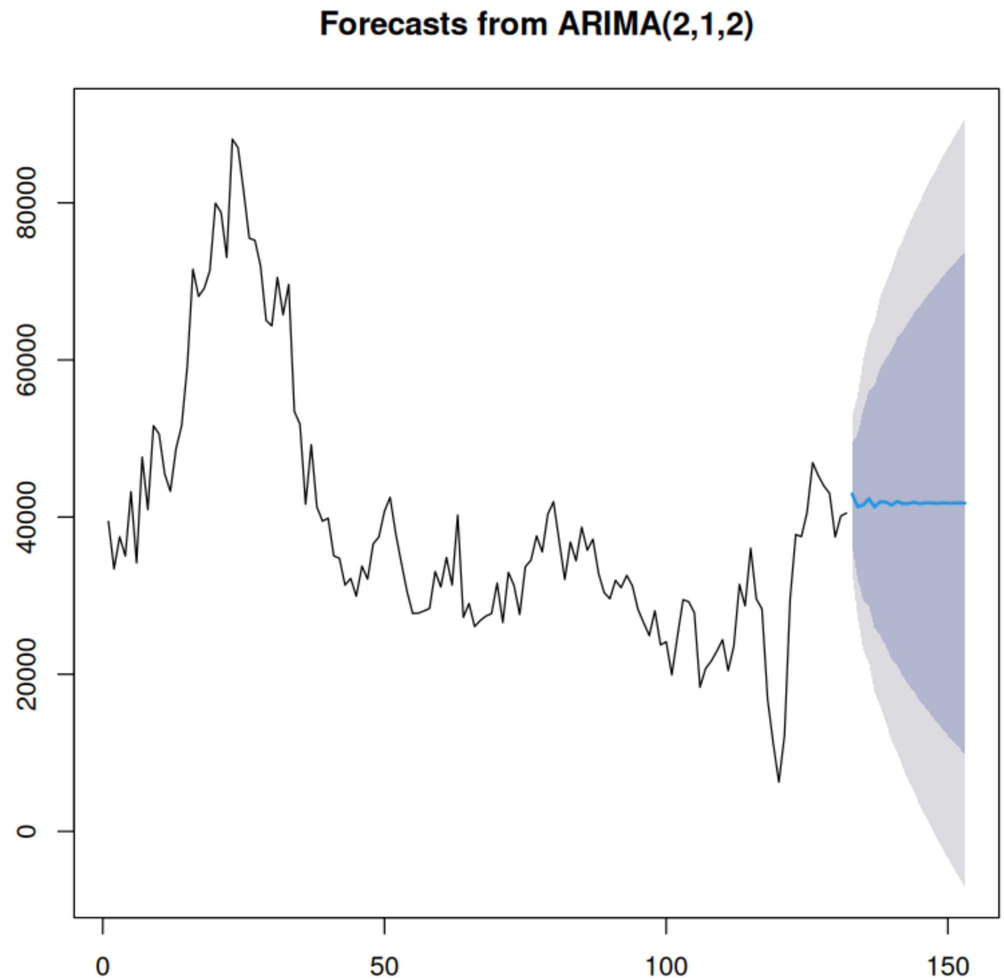
Product no. 28

```
[40]: product_28 <- data[data$Articulo == "Art_28", ] # filter data by product no.
arima_model <- auto.arima(product_28[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



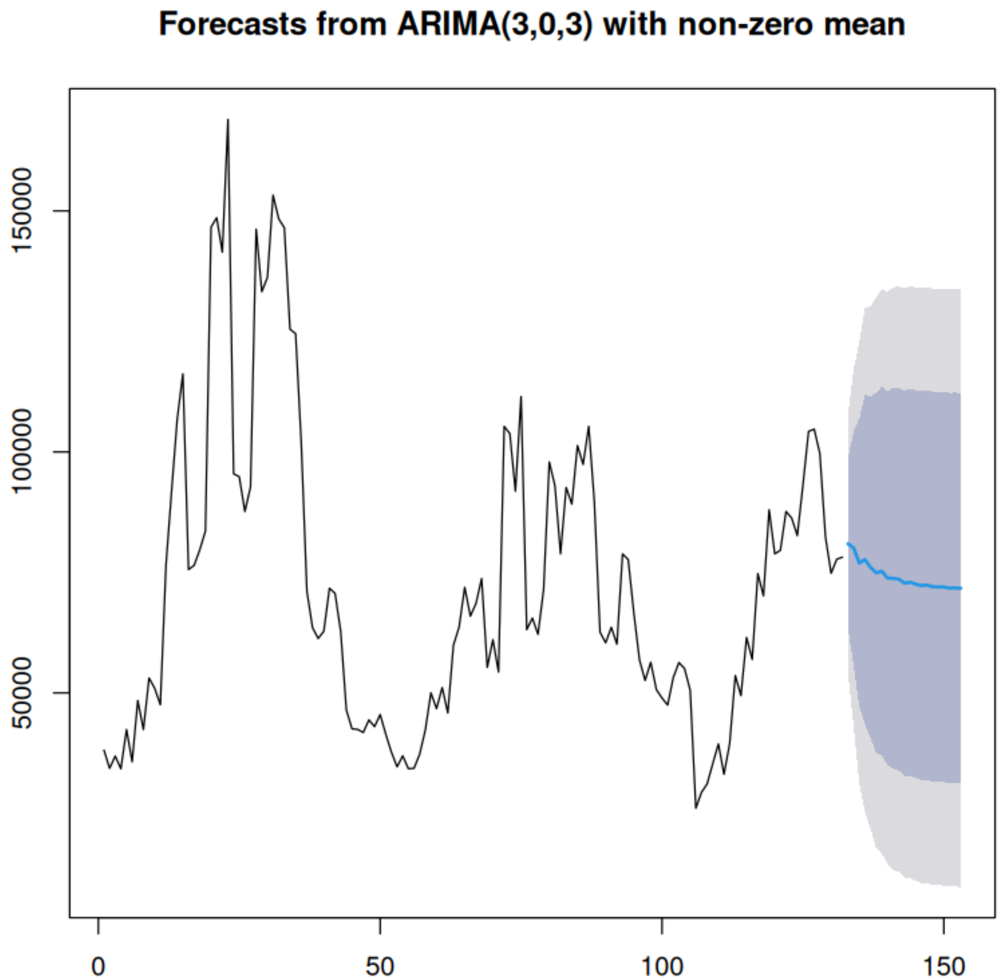
Product no. 29

```
[41]: product_29 <- data[data$Articulo == "Art_29", ] # filter data by product no.
arima_model <- auto.arima(product_29[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



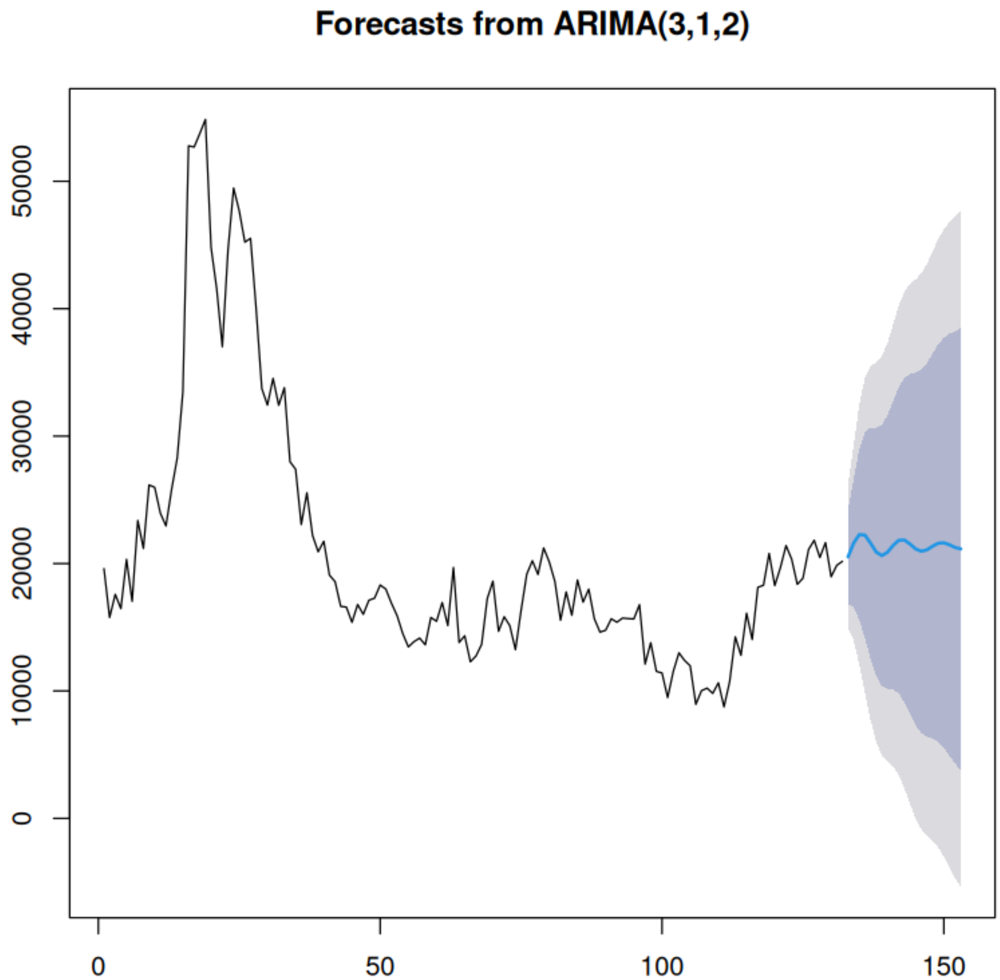
Product no. 30

[42]:
```
product_30 <- data[data$Articulo == "Art_30", ] # filter data by product no.
arima_model <- auto.arima(product_30[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



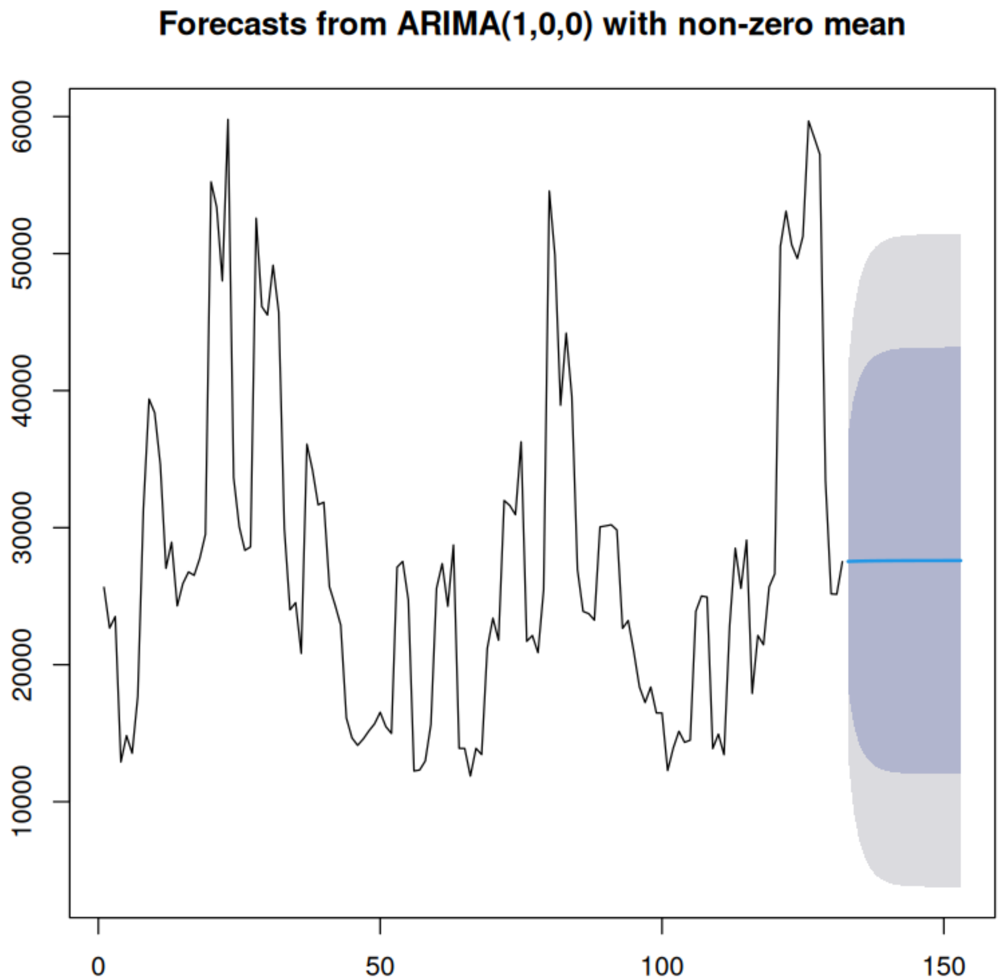
Product no. 31

[43]:
```
product_31 <- data[data$Articulo == "Art_31", ] # filter data by product no.
arima_model <- auto.arima(product_31[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



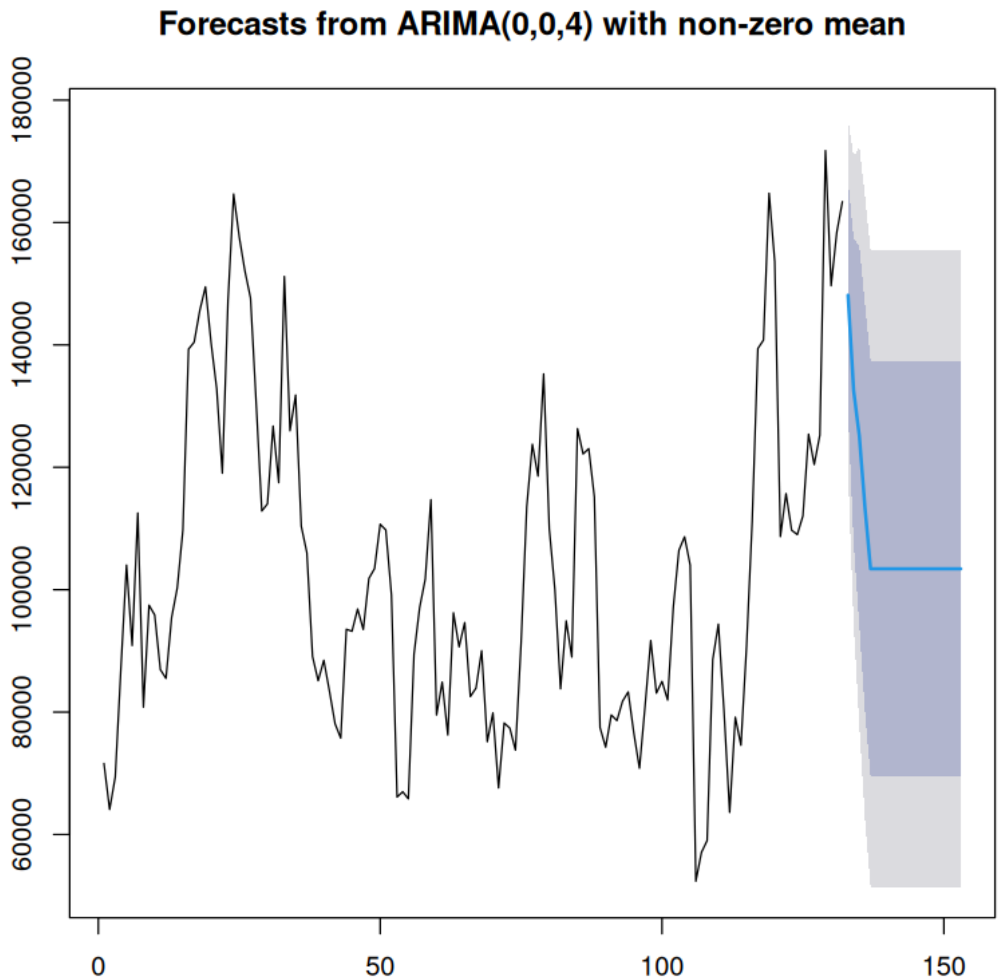
Product no. 32

[44]:
```
product_32 <- data[data$Articulo == "Art_32", ] # filter data by product no.
arima_model <- auto.arima(product_32[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



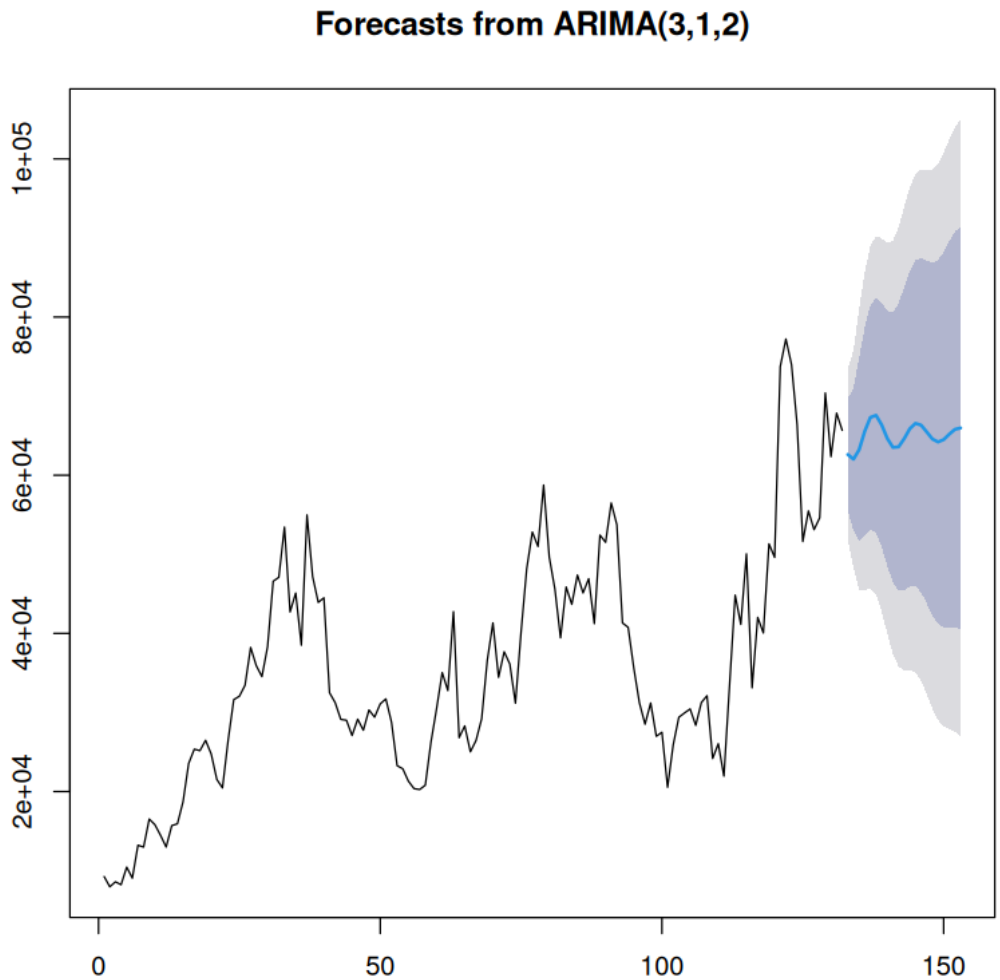
Product no. 33

[45]:
```r
product_33 <- data[data$Articulo == "Art_33", ] # filter data by product no.
arima_model <- auto.arima(product_33[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



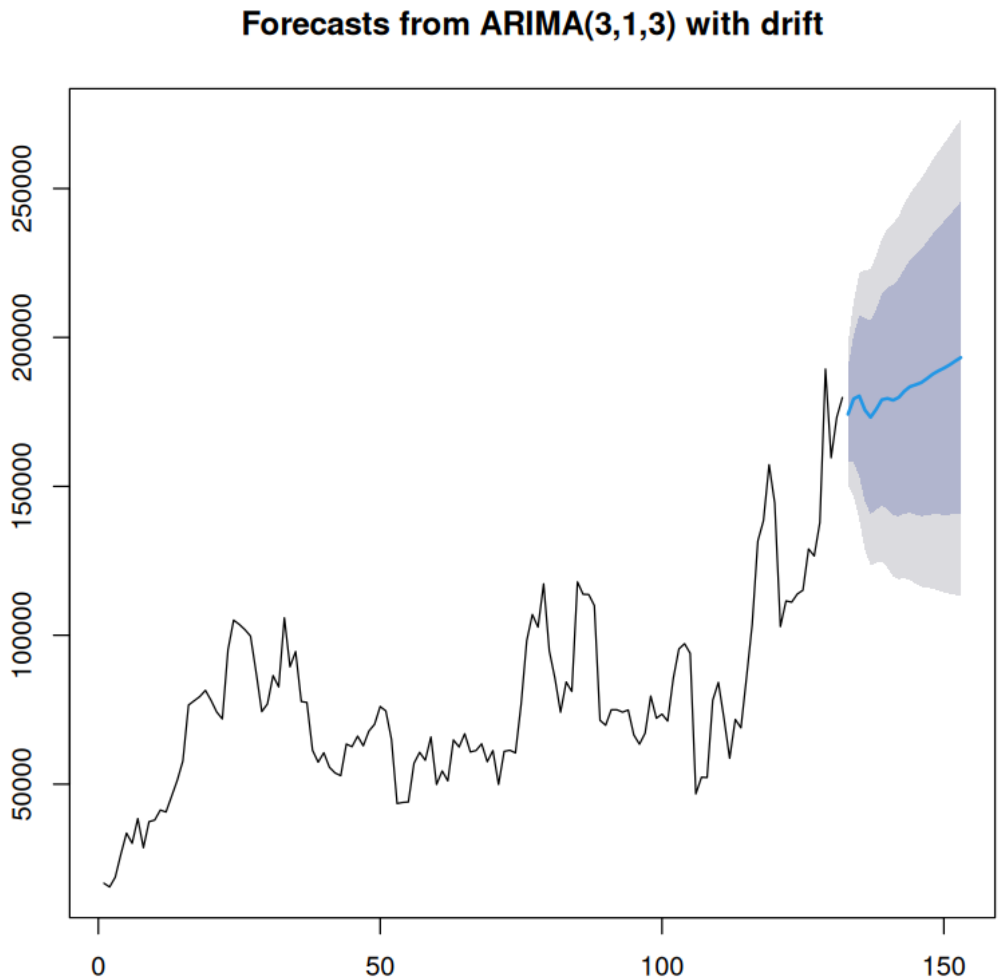
Product no. 34

[47]:
```
product_34 <- data[data$Articulo == "Art_34", ] # filter data by product no.
arima_model <- auto.arima(product_34[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```



Product no. 35

[46]:
```
product_35 <- data[data$Articulo == "Art_35", ] # filter data by product no.
arima_model <- auto.arima(product_35[, 3]) # fit a model on filtered data
fcast <- forecast(arima_model, h = 21) # forecasting time series
plot(fcast)
```


Forecasts from ARIMA(3,1,3) with drift
50000
100000
150000
200000
250000
0
50
100
150