

Pronóstico para serie de tiempo de demanda total de una red espacial

Aarón Ortiz Mendoza

Contents

Motivos	1
Problématica	1
Paquetes	1
Análisis de la serie de tiempo de la demanda total semanal	3
Serie de tiempo de demanda total semanal	3
Análisis de serie de Tiempo de demanda total	4
Gráfica del pronóstico final de la serie de tiempo de la red espacial	14
Extra-Análisis de grafo ponderado	15
Grafo Completo Ponderado	15
Histograma de ponderaciones del grafo	15
Gráfica del grafo con pesos	18
Referencias	19

Motivos

Buscando aplicar las técnicas de pronóstico **Box-Jenkins** a una serie de tiempo de una red espacio-temporal para conocer sus condiciones respecto al tipo de serie **ARIMA**. Se hara un pronóstico a una serie de tiempo de la demanda total semanal del grafo.

Problématica

Se trabaja con un tipo de dato de una red espacial, para el dataset espacio-temporal de las entregas semanales de la aplicación **PedalMe** en Londres, conteniendo una lista de vértices ponderada y una serie de tiempo de las demandas semanales del 2020 al 2021 por nodo. Se busca pronosticar la demanda total semanal de todos los nodos.

Paquetes

Librerías a instalar y utilizadas para realizar el pronóstico.

```
packages <- c(
  "dplyr", "tidyr", "lubridate", "igraph",
  "feasts", "ggplot2", "tsibble", "tseries", "forecast"
)
```

```
# Ver qué paquetes NO están instalados
```

```
missing_packages <- setdiff(packages, rownames(installed.packages()))
```

```
# Instalar solo los faltantes
```

```
if (length(missing_packages) > 0) {  
  install.packages(missing_packages)  
}
```

```
# Cargar todos los paquetes
```

```
invisible(lapply(packages, library, character.only = TRUE))
```

```
##
```

```
## Adjuntando el paquete: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
##   filter, lag
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
##   intersect, setdiff, setequal, union
```

```
##
```

```
## Adjuntando el paquete: 'lubridate'
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
##   date, intersect, setdiff, union
```

```
##
```

```
## Adjuntando el paquete: 'igraph'
```

```
## The following objects are masked from 'package:lubridate':
```

```
##
```

```
##   %--%, union
```

```
## The following object is masked from 'package:tidyr':
```

```
##
```

```
##   crossing
```

```
## The following objects are masked from 'package:dplyr':
```

```
##
```

```
##   as_data_frame, groups, union
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
##   decompose, spectrum
```

```
## The following object is masked from 'package:base':
```

```
##
```

```
##   union
```

```
## Cargando paquete requerido: fabletools
```

```
## Registered S3 method overwritten by 'tsibble':
```

```
##   method          from
```

```
##   as_tibble.grouped_df dplyr
```

```
##
```

```
## Adjuntando el paquete: 'fabletools'
```

```
## The following object is masked from 'package:igraph':
##
##     components
##
## Adjuntando el paquete: 'tsibble'
## The following objects are masked from 'package:igraph':
##
##     difference, union
## The following object is masked from 'package:lubridate':
##
##     interval
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, union
## Registered S3 method overwritten by 'quantmod':
##   method      from
##   as.zoo.data.frame zoo
```

Análisis de la serie de tiempo de la demanda total semanal

Se genera un panel de datos la serie de tiempo, tomando en cuenta la fecha de los valores que esta dados por un año y semana del mismo.

```
panel <- read.csv("pedalme_features.csv") %>%
  mutate(
    date = as.Date(ISOdate(year, 1, 4)) + weeks(week - 1)
  ) %>%
  select(date, year, week, node, demand) %>%
  arrange(date, node)
```

Serie de tiempo de demanda total semanal

Del panel de datos se toma el total de cada semana de las demandas para tener el total semanal.

```
network_weekly <- panel %>%
  group_by(date) %>%
  summarise(total_demand = sum(demand), .groups = "drop") %>%
  arrange(date)

knitr::kable(head(network_weekly, 10),
  caption = "Demanda total por semana")
```

Table 1: Demanda total por semana

date	total_demand
2020-05-30	22
2020-06-06	315
2020-06-13	317
2020-06-20	337
2020-06-27	392
2020-07-04	385

date	total_demand
2020-07-11	424
2020-07-18	439
2020-07-25	427
2020-08-01	481

Ahora los transformamos a un objeto de serie de tiempo para poder trabajar con el mismo.

```
network_ts <- network_weekly %>%
  as_tsibble(index = date)
network_ts <- network_ts %>%
  mutate(week = isoweek(date))
knitr::kable(head(network_ts,20),
  caption = "Demanda total por semana serie de tiempo")
```

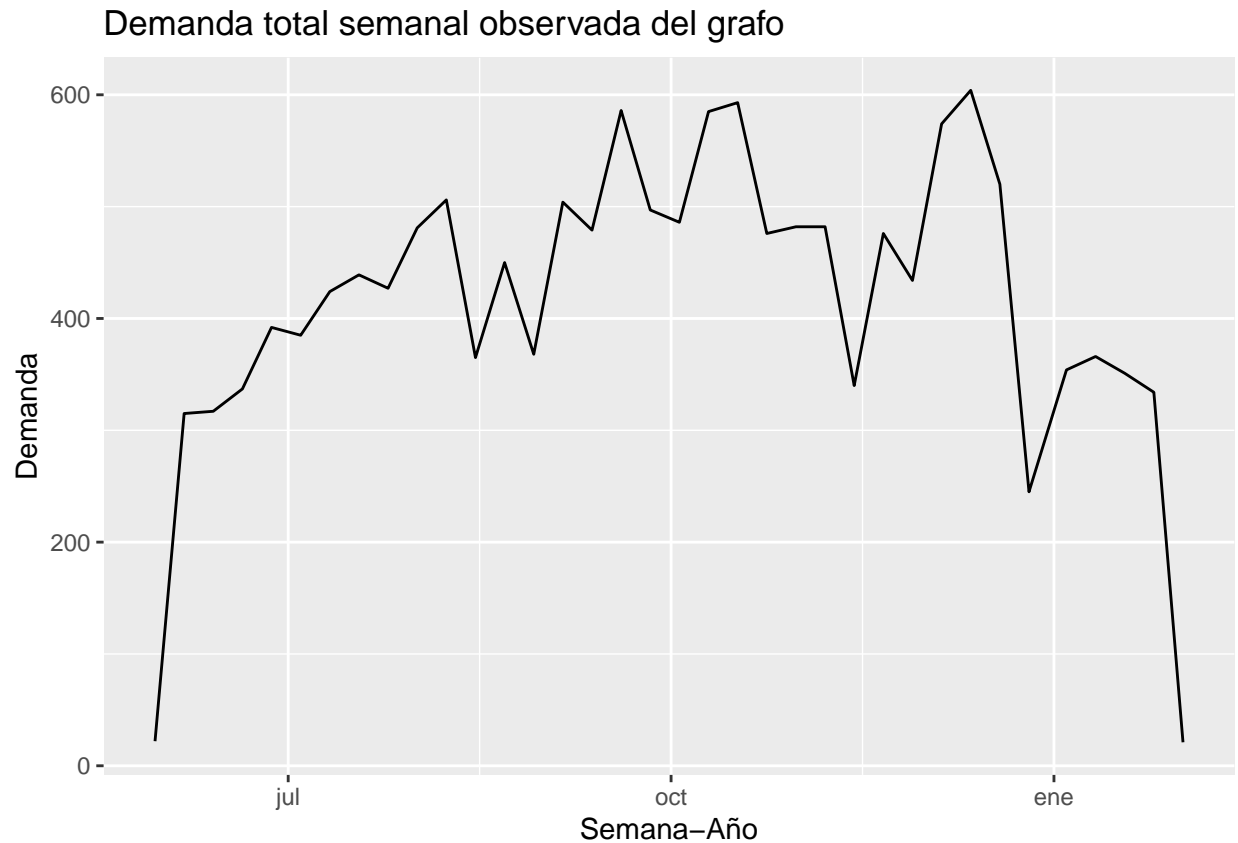
Table 2: Demanda total por semana serie de tiempo

date	total_demand	week
2020-05-30	22	22
2020-06-06	315	23
2020-06-13	317	24
2020-06-20	337	25
2020-06-27	392	26
2020-07-04	385	27
2020-07-11	424	28
2020-07-18	439	29
2020-07-25	427	30
2020-08-01	481	31
2020-08-08	506	32
2020-08-15	365	33
2020-08-22	450	34
2020-08-29	368	35
2020-09-05	504	36
2020-09-12	479	37
2020-09-19	586	38
2020-09-26	497	39
2020-10-03	486	40
2020-10-10	585	41

Análisis de serie de Tiempo de demanda total

Grafcamos nuestra serie de tiempo para conocer la misma y sus tendencias.

```
autoplot(network_ts, total_demand) +
  labs(
    title = "Demanda total semanal observada del grafo",
    x="Semana-Año",
    y = "Demanda"
  )
```

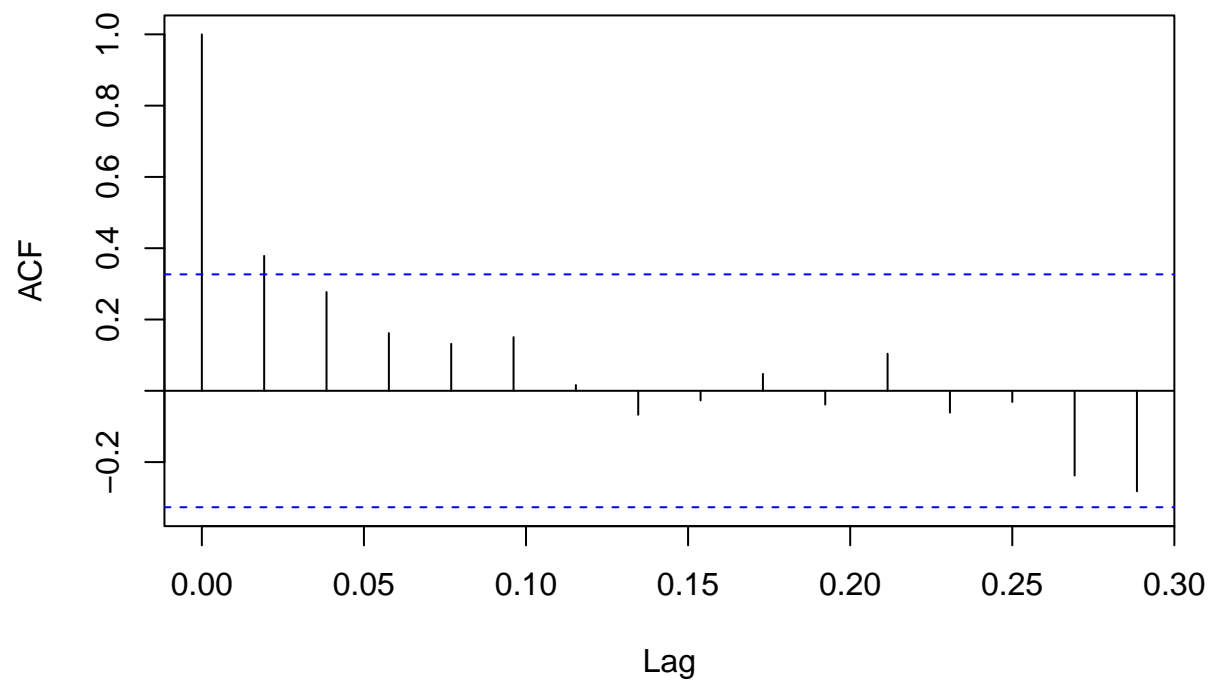


Se acaba realizando una diagnóstico de la serie para conocer los los parámetros a considerar del modelo *ARIMA* del método Box-Jenkins, donde queremos revisar los parámetros: * Autoregresión *P* * Integración *I* * Media Móvil *Q*

Además de realizar una prueba **aumentada de Dickey-Fuller** para conocer si la serie es estacionaria o se necesitara realizar una diferencia para volverla una. Además de las gráficas de las funciones de autoregresión y autoregresión parcial para saber que parámetros ir considerando para el modelo **ARIMA**.

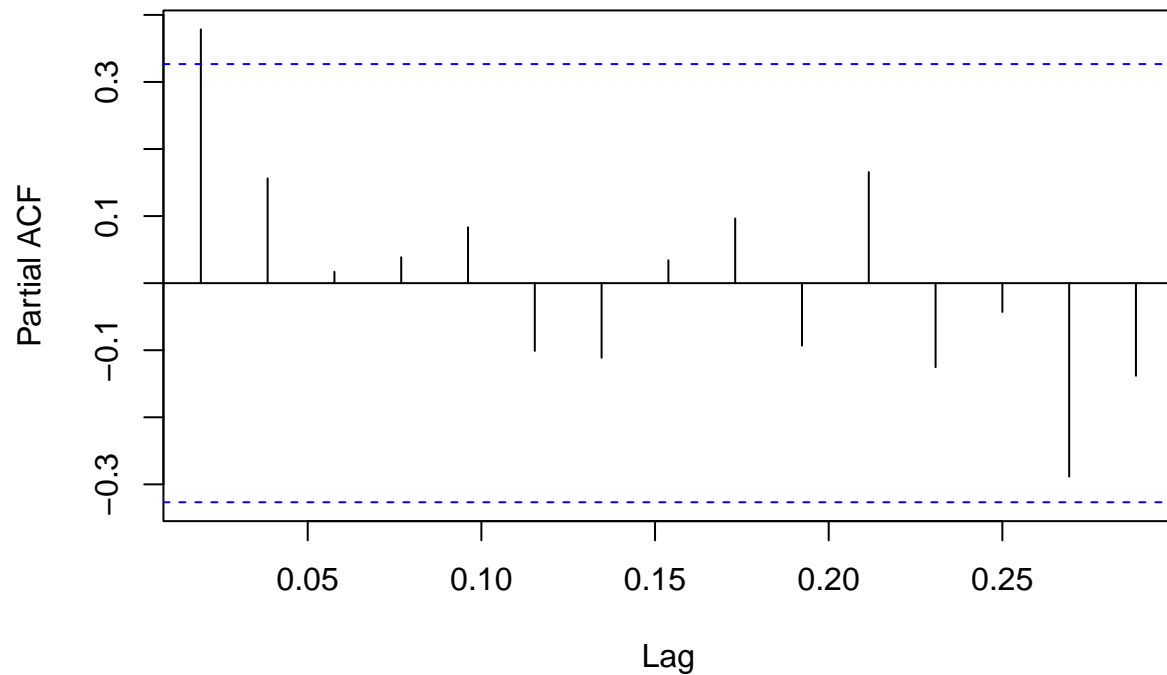
```
ts_network <- ts(  
  network_weekly$total_demand,  
  frequency = 52  
)  
  
acf_plot<-acf(ts_network, lag.max = 15)
```

Series ts_network



```
pacf_plot<-pacf(ts_network, lag.max = 15)
```

Series ts_network



```
adf.test(ts_network)
```

```
##  
## Augmented Dickey-Fuller Test  
##  
## data: ts_network  
## Dickey-Fuller = -0.72887, Lag order = 3, p-value = 0.9584  
## alternative hypothesis: stationary
```

Vemos que la serie debe diferenciarse así que los en base a tener las diferencias entre el valor actual y sus lags.

```
# Diferenciación
```

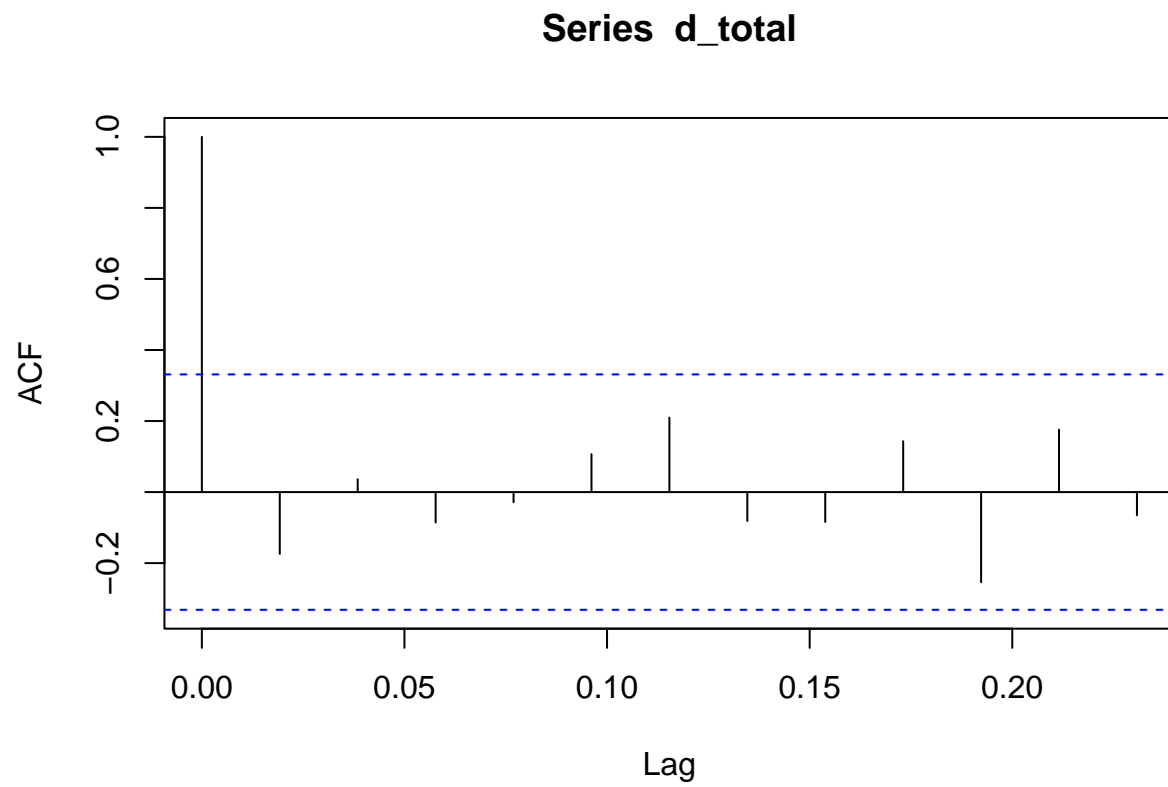
```
d_total <- diff(ts_network)
```

Realizamos de nuevo la prueba **aumentada de Dickey-Fuller** para revisar la estacionalidad como revisar de nuevo las funciones de autoregresión y autoregresión parcial para conocer los parámetros del modelo **ARIMA** siguiendo la metodología **Box-Jenkins**.

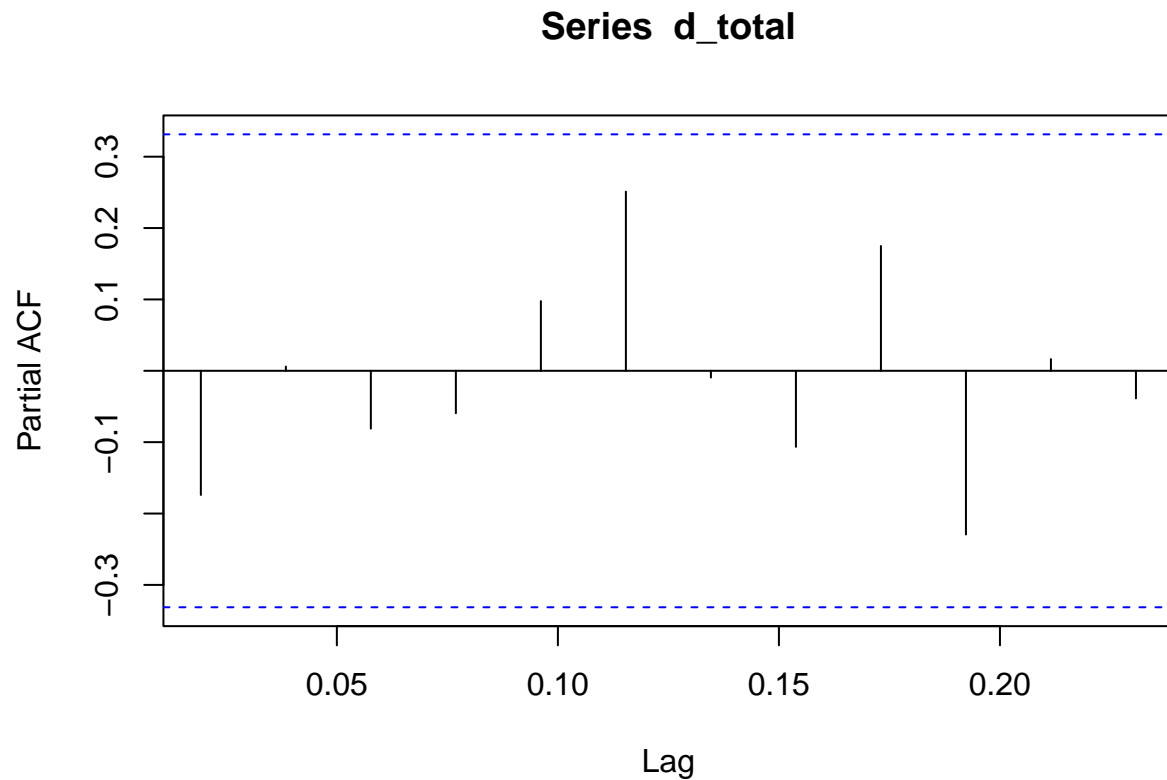
```
adf.test(d_total)
```

```
## Warning in adf.test(d_total): p-value smaller than printed p-value  
##  
## Augmented Dickey-Fuller Test  
##  
## data: d_total  
## Dickey-Fuller = -4.5376, Lag order = 3, p-value = 0.01  
## alternative hypothesis: stationary
```

```
acf_dtotal<-acf(d_total, lag.max = 12)
```



```
pacdf_dtotal<-pacf(d_total, lag.max = 12)
```

Dividiremos en la serie en un 80% para los datos de entrenamiento y el 20% para la prueba para poder ir seleccionando el modelo a escoger en base a lo visto, de la prueba **aumentada de Dickey-Fuller**, función de autoregresión y autoregresión parcial para ir tomando en cuenta que modelo aplicar en base a los parámetros.

```
n <- length(ts_network)
n_train <- floor(0.8 * n)

train_total <- window(ts_network, end = c(1, n_train))
test_total <- window(ts_network, start = c(1, n_train + 1))
```

Probamos tres modelos distintos con los parámetros de: * I(1) y MA(1) * AR(1) y I(1) * AR(1), I(1) y MA(1)

```
# Demanda total
m1_total <- Arima(train_total, order = c(0,1,1))
m2_total <- Arima(train_total, order = c(1,1,0))
m3_total <- Arima(train_total, order = c(1,1,1))
```

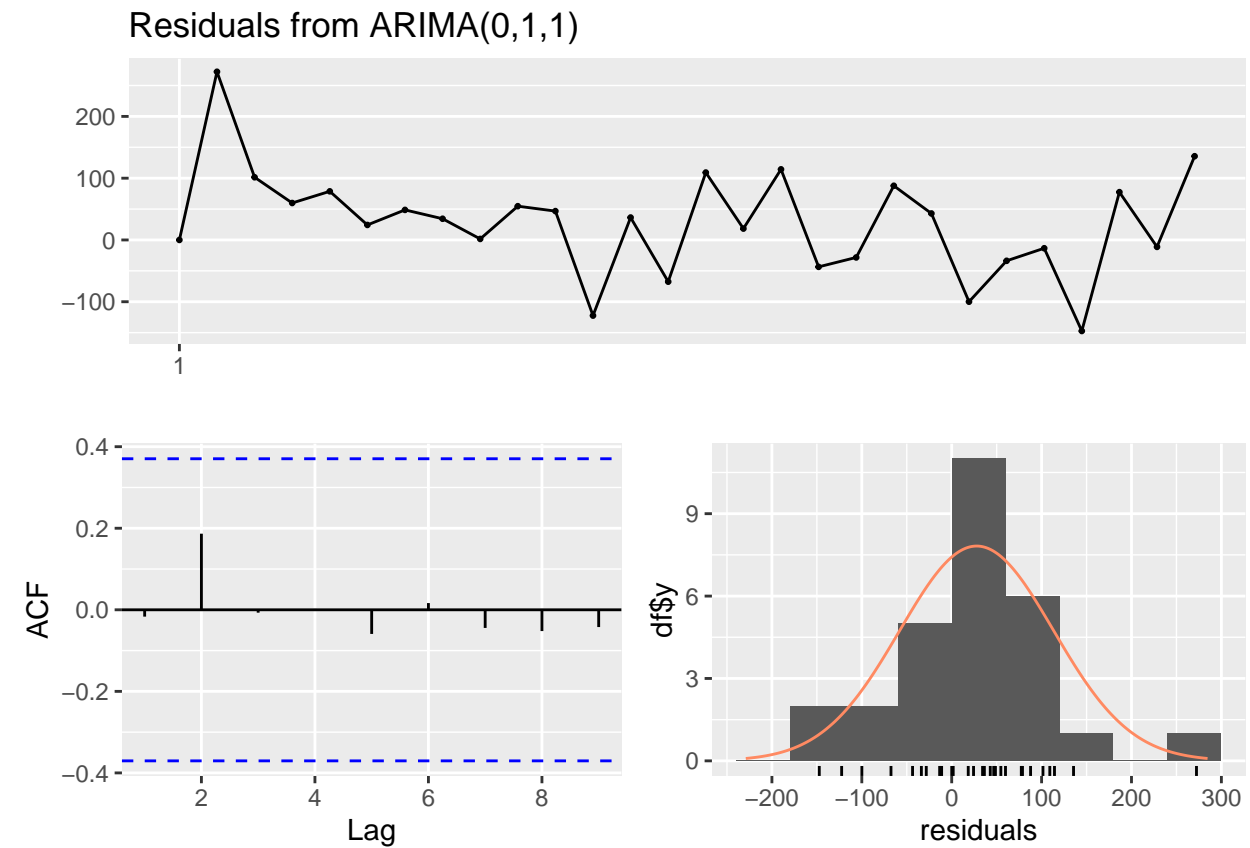
Realizamos un pronóstico con cada de un tiempo de cuatro semanas más.

```
h <- length(test_total)

fc_total_011 <- forecast(m1_total, h = h)
fc_total_110 <- forecast(m2_total, h = h)
fc_total_111 <- forecast(m3_total, h = h)
```

Revisaremos los residuos en base a la prueba Ljung-Box para considerar si alguno de los tres propuestos es mejor a los demás en base a su autoregresión.

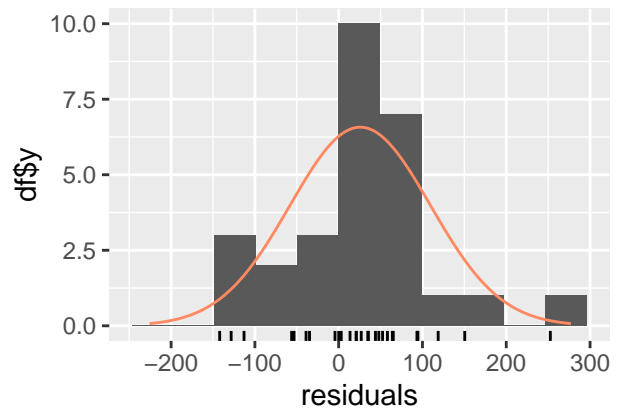
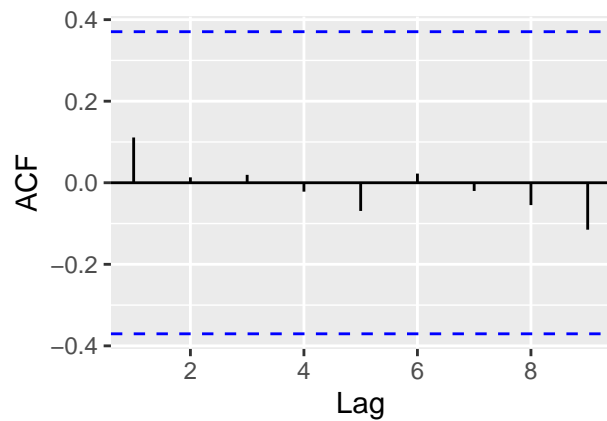
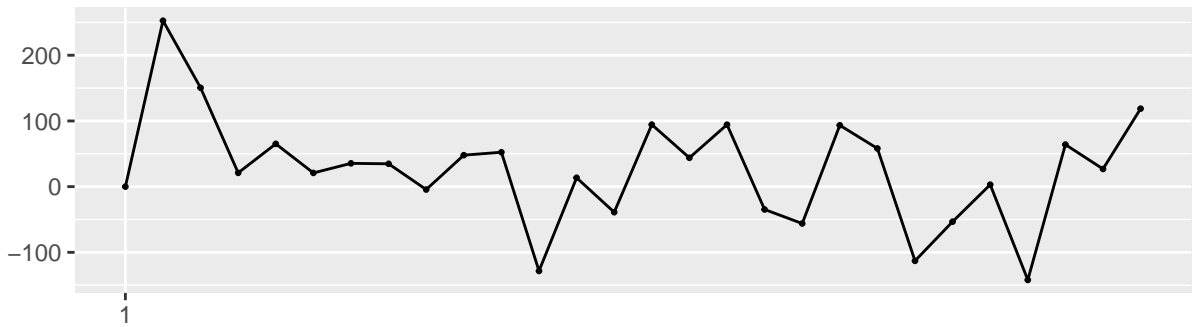
```
checkresiduals(m1_total)
```



```
##  
##  Ljung-Box test  
##  
## data:  Residuals from ARIMA(0,1,1)  
## Q* = 1.2749, df = 5, p-value = 0.9375  
##  
## Model df: 1.   Total lags used: 6
```

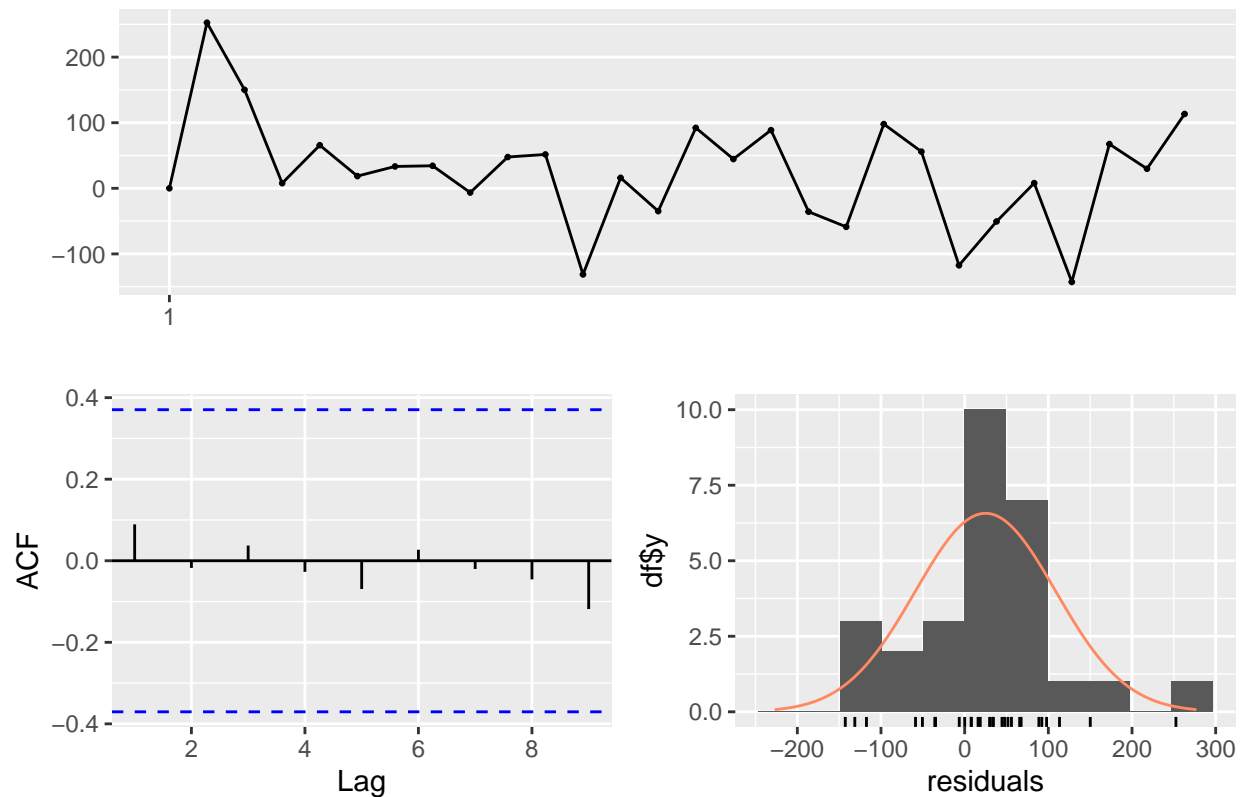
```
checkresiduals(m2_total)
```

Residuals from ARIMA(1,1,0)



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0)
## Q* = 0.61194, df = 5, p-value = 0.9874
##
## Model df: 1.   Total lags used: 6
checkresiduals(m3_total)
```

Residuals from ARIMA(1,1,1)



```
##
## Ljung-Box test
##
## data: Residuals from ARIMA(1,1,1)
## Q* = 0.53295, df = 4, p-value = 0.9702
##
## Model df: 2. Total lags used: 6
```

Tomando en cuenta que contamos con sólo 36 entradas de datos, siendo muy pocas, consideramos un criterio de información más sencillo siendo este el “Akaike Information Criterion” para poder seleccionar en base los errores estándar de alguna de nuestras propuestas, seleccionando para poder escoger con la evaluación más baja.

```
AIC(m1_total, m2_total, m3_total)
```

```
##          df      AIC
## m1_total  2 323.8772
## m2_total  2 322.5759
## m3_total  3 324.5351
```

Tomaremos las medidas en base a los datos de prueba la factibilidad de considerar en base a diferentes criterios de errores.

```
accuracy(fc_total_011, test_total)
```

```
##          ME      RMSE      MAE      MPE      MAPE  MASE
## Training set  27.74776  88.53014  68.28339   6.249306  16.47685  NaN
## Test set     -170.73890 236.22607 191.71043 -333.511691 336.98380  NaN
##
##          ACF1 Theil's U
```

```
## Training set -0.01674025      NA
## Test set      0.14202740  1.627661
```

```
accuracy(fc_total_110, test_total)
```

```
##              ME      RMSE      MAE      MPE      MAPE  MASE
## Training set  25.72814  86.22295  66.49755   5.875975  16.00865  NaN
## Test set     -175.57171 241.72151 200.80674 -338.857622 343.03561  NaN
##              ACF1 Theil's U
## Training set  0.1110011      NA
## Test set      0.1505220  1.666014
```

```
accuracy(fc_total_111, test_total)
```

```
##              ME      RMSE      MAE      MPE      MAPE  MASE
## Training set  24.93282  86.15284  66.18857   5.675558  15.91267  NaN
## Test set     -178.09026 243.53615 203.09229 -341.107202 345.24661  NaN
##              ACF1 Theil's U
## Training set  0.08913092      NA
## Test set      0.15113860  1.681415
```

Acabamos escogiendo el modelo **ARIMA** con los parámetros I(1) y MA(1), donde dado que por el intervalo de confianza sabemos que el pronóstico producido sera uno también con valores menores a cero, transformamos nuestra serie momentaneamente a una con valores logaritmicos para evitar esa cuestión y tener pronósticos con solo valores no negativos.

```
ts_total_log <- log1p(ts_network)
# Modelo ya ajustado
model_log <- Arima(ts_total_log, order = c(0,1,1))

# Pronóstico
fc_log <- forecast(model_log, h = 10, level = c(80, 95))

# Media
mean_fc <- expm1(fc_log$mean)

# Intervalos
lower80 <- expm1(fc_log$lower[, "80%"])
upper80 <- expm1(fc_log$upper[, "80%"])

lower95 <- expm1(fc_log$lower[, "95%"])
upper95 <- expm1(fc_log$upper[, "95%"])
```

Tranformando igualmente los datos para poder tener una orden crónologico como la serie de tiempo original para poder tener una coherencia en nuestra serie de tiempo.

```
future_dates <- seq(
  from = max(network_weekly$date) + weeks(1),
  by = "week",
  length.out = length(mean_fc)
)
fc_df <- data.frame(
  date = future_dates,
  mean = as.numeric(mean_fc),
  lower80 = as.numeric(lower80),
  upper80 = as.numeric(upper80),
  lower95 = as.numeric(lower95),
```

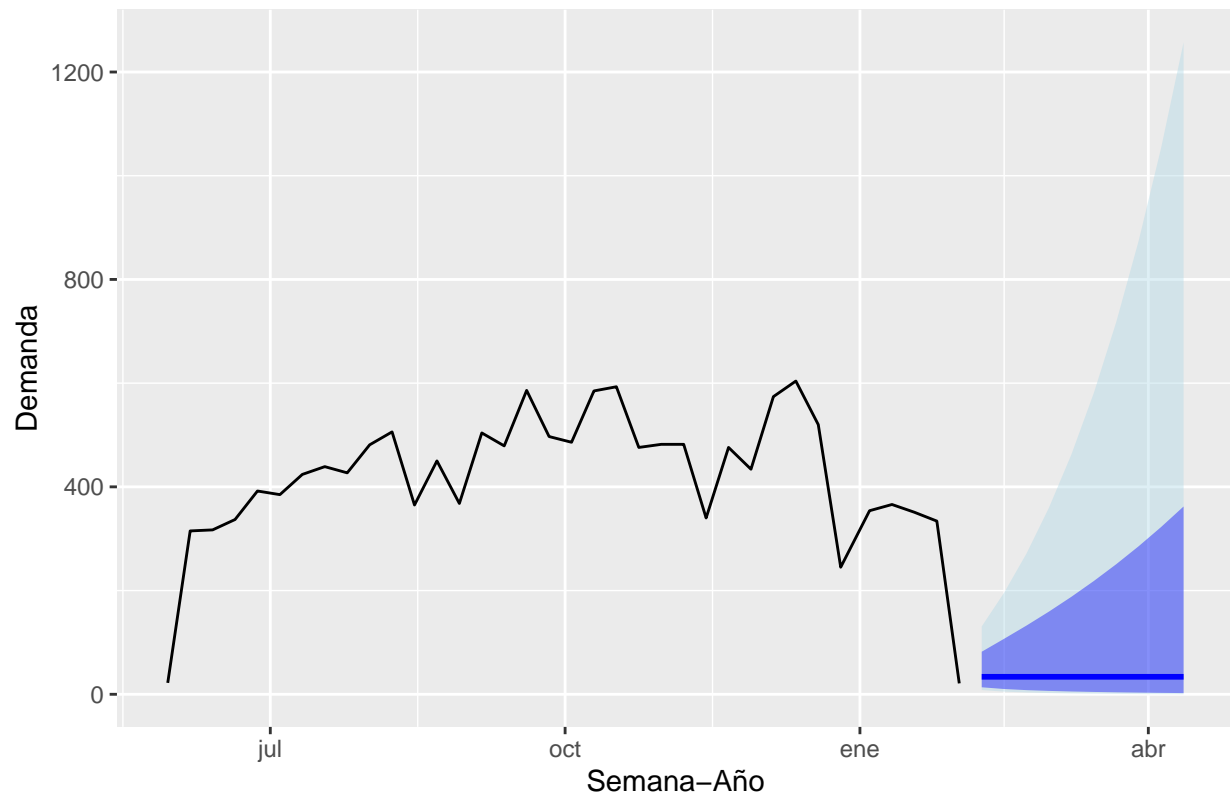
```
upper95 = as.numeric(upper95)
)
```

Gráfica del pronóstico final de la serie de tiempo de la red espacial

Gráfica con los valores presentados de la serie de tiempo de la demanda total semanal de nuestra red espacial.

```
ggplot() +
  geom_line(
    data = network_weekly,
    aes(x = date, y = total_demand),
    color = "black"
  ) +
  geom_ribbon(
    data = fc_df,
    aes(x = date, ymin = lower95, ymax = upper95),
    fill = "lightblue",
    alpha = 0.4
  ) +
  geom_ribbon(
    data = fc_df,
    aes(x = date, ymin = lower80, ymax = upper80),
    fill = "blue",
    alpha = 0.4
  ) +
  geom_line(
    data = fc_df,
    aes(x = date, y = mean),
    color = "blue",
    linewidth = 1
  ) +
  labs(
    title = "Pronóstico semanal de la demanda del grafo",
    x = "Semana-Año",
    y = "Demanda"
  )
```

Pronóstico semanal de la demanda del grafo



Extra-Análisis de grafo ponderado

Grafo Completo Ponderado

Trabajamos con un grafo de 15 nodos y 15 vérticos, no dirigidos con ponderación entre cada uno, unos más grandes que otros donde veremos los 5 primeros nodos con mayor valor ponderado entre cada uno. Crearemos:

- Matriz de adyacencia entre los nodos con su ponderación
- Gráfica de conexión entre los nodos con sus vértices en base a su ponderación

```
edges <- read.csv("pedalme_edges.csv")

g <- graph_from_data_frame(edges, directed = FALSE)
E(g)$weight <- edges$weight

matriz_adyacencia <- as.matrix(
  as_adjacency_matrix(g, attr = "weight", sparse = FALSE)
)
diag(matriz_adyacencia) <- 0
```

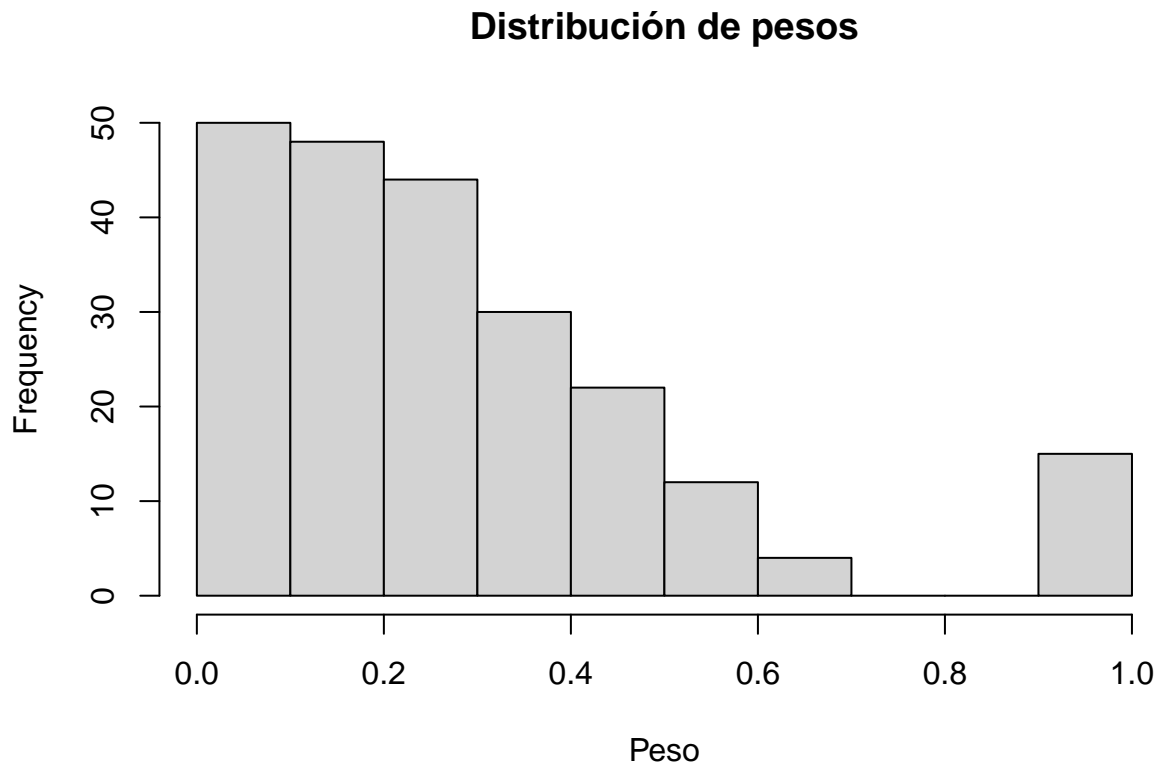
Histograma de ponderaciones del grafo

Se muestra la frecuencia de los valores ponderados de los vértices entre los nodos, dado que en la red se conectan los nodos con todos los de la misma red, se considera al grafo uno completo, los valores de uno son los de los nodos conectados consigo mismos. Además de un resumen de los valores de las ponderaciones de los vértices de la red.

```
summary(E(g)$weight)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.007879 0.120345 0.224675 0.287179 0.382945 1.000000
```

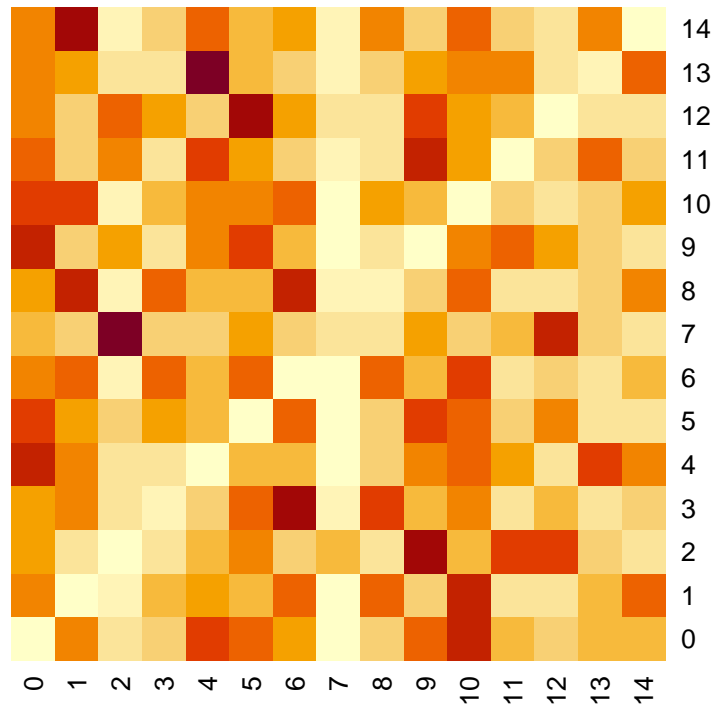
```
hist(E(g)$weight, main = "Distribución de pesos", xlab = "Peso")
```



Mapa de calor del grafo completo ponderado Como en el caso anterior se muestra en base a un color más oscuro a claro, llenando de un valor entre 0 a 1 de los vértices ponderados entre los vértices.

```
heatmap(matriz_adyacencia,
        Rowv = NA, Colv = NA,
        scale="row",
        main = "Mapa de calor del grafo ponderado")
```


Mapa de calor del grafo ponderado



Dimensiones del grafo Dimensiones de la matriz de adyacencia además de los valores de las conexiones entre los nodos a través de la lista de vértices.

```
# Dimensiones y estructura
dim(edges)
```

```
## [1] 225  3
```

```
str(edges)
```

```
## 'data.frame':  225 obs. of  3 variables:
## $ from : int  0 0 0 0 0 0 0 0 0 0 ...
## $ to   : int  0 1 2 3 4 5 6 7 8 9 ...
## $ weight: num  1 0.425 0.157 0.207 0.564 ...
```

```
# Primeras filas
```

```
knitr::kable(head(edges, 10),
  caption = "Lista de vértices ponderados del grafo")
```

Table 3: Lista de vértices ponderados del grafo

from	to	weight
0	0	1.0000000
0	1	0.4254590
0	2	0.1573554
0	3	0.2065002
0	4	0.5641481
0	5	0.5185980

from	to	weight
0	6	0.4004257
0	7	0.0266818
0	8	0.2246753
0	9	0.5012188

Gráfica del grafo con pesos

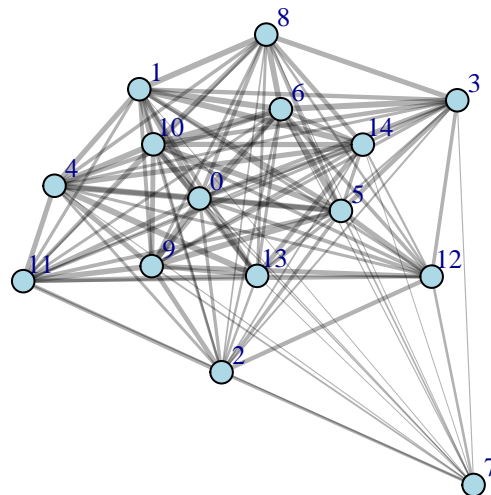
Gráfica de toda la red donde se presenta los nodos y vértices, donde los pesos se representan a través del grosor del vértice que conecta los nodos.

```
# Crear grafo
g <- graph_from_adjacency_matrix(matriz_adyacencia, mode = "undirected", weighted = TRUE)

layout_fr <- layout_with_fr(g, weights = E(g)$weight)

plot(g,
     layout = layout_fr,
     vertex.size = 10,
     vertex.color = "lightblue",
     vertex.label.dist = 1.5,
     vertex.label.cex = 0.8,
     vertex.label.color = "darkblue",
     edge.width = sqrt(E(g)$weight) * 3,
     edge.color = rgb(0,0,0,0.3),
     main = "Grafo ponderado de bicicletas")
```

Grafo ponderado de bicicletas



Referencias

- GeeksforGeeks. 2025. “Box-Jenkins Methodology for ARIMA Models.” <https://www.geeksforgeeks.org/machine-learning/box-jenkins-methodology-for-arima-models/>.
- Moffatt, Mike. 2024. “An Introduction to Akaike’s Information Criterion (AIC).” <https://www.thoughtco.com/introduction-to-akaike-information-criterion-1145956>.
- Rozemberczki, Benedek, Paul Scherer, Yixuan He, George Panagopoulos, Alexander Riedel, Maria Astefanoaei, Oliver Kiss, et al. 2021. “PyTorch Geometric Temporal: Spatiotemporal Signal Processing with Neural Machine Learning Models.” In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*.
- Scott, Gordon. 2025. “Box-Jenkins Model Explained: Uses, Methodology, and Forecasting.” <https://www.investopedia.com/terms/b/box-jenkins-model.asp>.