Predicción del monto total que se va a pagar por remesas en dolares que se originan en un día.

Análisis de Series de Tiempo - Grupo 5

2024-10-20

Las remesas son transferencias de dinero realizadas principalmente por emigrantes a sus familiares o relacionados en su país de origen. En este proceso participan los siguientes actores: remitente (Persona que desde el país de origen, se acerca a un agente para enviar un determinado monto de dólares), el agente en el lugar de origen (oficina donde se inicia la operación financiera), la empresa de transferencia de dinero (empresa que hace el traslado del monto del país de origen al país destino), el agente en el lugar de pago (oficina donde se realiza el desembolso del dinero) y el receptor (persona que en el país de destino recibe el dinero). Las transferencias entre la empresa de transferencia y los agentes se realizan en dólares, y las tasas de cambio se fijan en la fecha de origen de la transacción con una tasa de referencia establecida para cada país. Esto significa que todas las transacciones iniciadas en el día se pagarán según la tasa de referencia correspondiente a ese día.

Esto genera un desafío para los agentes en el lugar de pago, dado que la tasa de pago asociada a las operaciones es informada al remitente y se fija según la tasa de referencia vigente en la fecha de origen y se mantiene inalterada hasta que el giro es cobrado (con un tope máximo de 30 días calendario). Existe una exposición a la volatilidad de la tasa de cambio entre el día inicial de la transacción y la fecha de cobro. Estas transacciones se identifican el día en que se realiza el pago al receptor, momento en que la tasa de mercado a la que el agente de pago convierte a moneda local los dólares recibidos por la empresa de transferencia de dinero puede haber variado con relación a la tasa de cambio con la que se fijó la conversión al momento de la originación de la remesa.

En resumen, hay transacciones pagadas en el día que tienen como referencia una tasa de cambio de un día diferente, y el agente del lugar de pago puede monetizar (convertir de dólares a pesos colombianos) a la tasa del día . De esta forma, en escenarios de revaluación del peso colombiano, es decir, caídas en la tasa de cambio, se generarían menores ingresos o pérdidas por concepto de monetización de dólares, materializándose el riesgo cambiario. Este proceso se ve afectado por las celebraciones especiales en cada uno de los países, tales como día de amor y amistad, día de la madre, navidad, etc.

Según lo expuesto anteriormente, se identifican las siguientes variables para el conjunto de datos provisto por una entidad que realiza remesas en Colombia bajo acuerdo de confidencialidad:

- **Fecha de pago.**
- **Pais de Origen.**
- **Fecha de origen.**
- **Monto en Pesos.**
- **Canal.**
- **Numero de transacciones.**
- **Monto en dolares.**
- **Tasa pactada.**

En relación con la importancia de analizar esta información enfocado en la variable tiempo se puede decir que; el análisis de las variables de tipo fecha (Fecha de la operación y Fecha de cobro) es crucial debido a la volatilidad de las tasas de cambio entre el día en que se origina la remesa y el día en que se paga. Por otra parte, el pronóstico del monto total a pagar en remesas es esencial para mitigar este riesgo, permitiendo anticipar posibles pérdidas por variaciones en la tasa de cambio y tomar decisiones informadas sobre cobertura y gestión del riesgo.

Documento Bookdown - Propuesta avance 2

En este momento deberemos retomar la Unidad 1 en la cual se creó un minilibro que contiene el entregable de dicha unidad. Este documento tiene como repositorio GitHub (elaborado desde Markdown). Ahora, en esta Unidad 2, se debe continuar con los datos presentados en dicho entregable y se debe evidenciar, en una de las variables en el tiempo, la aproximación en promedio móvil, en rezagos y en estacionalidad. Todo lo anterior, a través de funciones y gráficas que permitan detectar patrones y ciclos de la variable.

```
#install.packages("forecast")
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(tseries)
```

```
##
##     'tseries' version: 0.10-58
##
##     'tseries' is a package for time series analysis and computational finance.
##
##     See 'library(help="tseries")' for details.
```

```
#install.packages("timsac")
library(timsac)
library(ggplot2)
library(changepoint)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Successfully loaded changepoint package version 2.2.4
##  See NEWS for details of changes.
```

```
library(readxl)
```

```
#install.packages(c('quantmod','ff','foreign','R.matlab'),dependency=T)
suppressPackageStartupMessages(library(tidyverse))
```

```
## Warning: package 'readr' was built under R version 4.3.3
```

```
## Warning: package 'dplyr' was built under R version 4.3.3
```

```
library(tidyverse)
Datos <- read_csv('F:/Base_resumida_2023_2024.csv',show_col_types = FALSE)
head(Datos, 50)
```

```
## # A tibble: 50 x 7
##    Fecha_Pago Pais_Origen Fecha_Origen Canal     Transacciones_USD Monto_USD Monto_COP
##    <date>     <chr>       <date>       <chr>                 <dbl>     <dbl>     <dbl>
##  1 2023-01-01 AE          2022-12-31   APN                       1      369.    1765896
##  2 2023-01-01 AE          2023-01-01   APN                       3      471.    2252157
##  3 2023-01-01 AT          2023-01-01   APN                       3     1146.    5480357
```

```
##  4 2023-01-01 AU          2022-12-31   APN                     3    1781.    8518037
##  5 2023-01-01 AU          2023-01-01   APN                    18   13946.   66679489
##  6 2023-01-01 BE          2022-12-30   RedPropia               1      48.2    230602
##  7 2023-01-01 BE          2023-01-01   APN                     1     418.    2000000
##  8 2023-01-01 BR          2022-12-31   APN                    12    2206.   10549058
##  9 2023-01-01 BR          2022-12-31   RedPropia               2     415.    1982944
## 10 2023-01-01 BR          2023-01-01   APN                    35   10959.   52399857
## # i 40 more rows
```

```
Indice.ts<-ts(Datos$Monto_USD, start =c(2023,1), frequency = 12 )
#(Indice.ts)
```

```
class(Indice.ts)
```

```
## [1] "ts"
```

```
start(Indice.ts)
```

```
## [1] 2023    1
```

```
end(Indice.ts)
```

```
## [1] 46144    12
```

Estos valores indican Diciembre del año 46144 lo que no es corecto ni lógico dado que los datos van hasta octubre de 2024.

En este caso, se procede a agrupar y sumar el monto en dolares por dia y se creará una serie de tiempo sobre estas varables usando ts.

```
Datos$Fecha_Pago <- as.Date(Datos$Fecha_Pago, format = "%Y-%m-%d")
Datos_agg <- aggregate(Monto_USD ~ Fecha_Pago, data = Datos, sum)
```

```
# Crear la serie de tiempo
serie_ts <- ts(Datos_agg$Monto_USD, start = c(as.numeric(format(min(Datos_agg$Fecha_Pago), "%Y")), as.nu
(serie_ts)
```

```
## Time Series:
## Start = c(2023, 1)
## End = c(2024, 284)
## Frequency = 365
##   [1]   314473.1  4700346.1  6069974.1  6164948.9  6986643.0  6616416.0  5392530.2  1368008.4
##   [9]  2248875.5  8307518.8  6546327.3  4509599.3  5737621.1  4668733.8  1329182.5  6681862.6
##  [17]  5926830.6  5664752.1  5235327.5  5645884.9  4690341.0  1282273.5  4082641.5  6148264.7
##  [25]  5876090.1  4762893.9  5861068.1  4659864.5  1183509.1  6693741.0  7450724.1  7297537.1
##  [33]  7278922.2  7021275.3  5930574.6  1506607.6  8058198.8  7973355.0  6715875.0  6143883.4
##  [41]  6418863.8  5695236.5  1433886.2  6939748.9  7026277.5  6409925.2  7535383.3  8464289.8
##  [49]  6327532.8  1681515.1  6674906.8  6175714.9  6615088.6  6371841.1  6515140.9  5399052.6
##  [57]  1485372.5  7218453.8  7396809.3  8092041.5  8052039.2  8198663.8  6516882.6  1604140.3
##  [65]  8109699.7  6841650.8  6177331.8  5732340.7  6759066.8  5560840.4  1497907.2  7083909.6
##  [73]  6713222.0  6961578.6  7263762.1  7951493.4  6163578.0  1581605.8  3274220.4  9256305.6
##  [81]  7101678.3  6046199.3  6496190.2  5168790.3  1525396.4  6925066.6  6124443.4  5968335.6
##  [89]  5845369.7  7966852.4  7291303.0  1856617.1  9236108.9  8164548.3  8004949.1  2514662.1
##  [97]  1531244.7  5243993.1  1346539.0  8317608.5  7707540.7  6643078.9  6074091.7  6853682.1
## [105]  6035473.8  1607960.8  7528328.4  7096677.8  6498093.5  6536824.6  6507232.2  6033553.3
## [113]  1518827.9  7264795.5  6518810.2  5994862.4  7536306.0  8613419.5  7255698.5  1931756.8
## [121]  2739327.3 10420013.2  8867168.5  8711491.7  7761317.1  6417886.4  1753203.2  7282059.0
## [129]  7188483.7  6740174.6  6162021.7  8206571.8  7256215.3  1777446.6  7446557.6  6784442.4
## [137]  6238983.8  5692616.8  6505164.7  4137160.6  1728970.3  2242886.5  7926913.4  5759638.4
```

```
## [145]  5164065.6  3726434.0  5236303.7  1389816.8  4149486.0  6474502.6  6433081.4  7332152.1
## [153]  7726673.8  6316966.1  1669507.6  8378369.8  6947498.7  7751210.2  5911044.2  6783155.9
## [161]  5135652.0  1729162.2  2613712.8  6966285.5  8041686.3  7063770.0  7008665.8  6614357.2
## [169]  1749712.4  2791168.2  8360321.4  7535034.1  6178470.9  6415784.0  4836203.1  1376986.1
## [177]  6978628.0  6428276.0  4732962.8  7617789.5  7883338.6  7490169.5  1918182.7  4200116.4
## [185] 11189788.8  8895365.7  7740735.5  7803764.5  6688634.7  1802593.2  7572181.5  7069739.7
## [193]  6424353.8  5826196.6  6357001.8  5747147.0  1631454.8  8131152.1  6783472.1  6470289.7
## [201]  2916365.9  8100681.1  5402015.9  1296905.1  7179008.9  7038431.7  6397416.2  6072333.8
## [209]  6966187.2  5751397.8  1715854.5  8789056.8  9137281.1  8529185.8  8076552.9  8954585.1
## [217]  7406210.7  2025980.7  3667874.1  9764655.7  7650382.8  6512355.0  6668420.6  5257432.0
## [225]  1460387.6  7058204.5  6928458.6  7704196.7  7276921.0  7270269.7  5835270.1  1648115.7
## [233]  2865195.2  8713377.0  7058716.0  5893077.2  6704749.3  5301743.4  1552663.5  7740846.0
## [241]  7355943.0  7113233.7  7263892.3  8774866.0  6910845.9  1934002.5  8732485.4  8416985.2
## [249]  7896627.5  5447209.1  7914513.4  5410397.4  1662772.2  7095546.7  6830140.7  5800521.3
## [257]  5884305.7  6742415.0  5848987.1  1599481.0  7188551.3  6354637.6  6076460.7  6028572.8
## [265]  6566439.1  4728961.0  1689781.7  7303868.8  6836161.9  6775506.6  6793959.9  7538577.6
## [273]  6711792.7  2235300.3  9862087.6 10150399.5  9042188.4  8973380.3 10216431.7  7277818.0
## [281]  2264959.5  8664824.5  7775003.1  6831189.6  5440913.6  6862378.1  5478542.4  1782013.1
## [289]  3401187.1  9710719.0  6663916.6  6307454.3  6710536.6  6095149.9  1775294.7  7319210.8
## [297]  6650116.9  6022527.2  5406713.8  5825848.7  5072293.2  1436303.6  6945282.7  7363174.1
## [305]  6919058.9  7096878.9  8023584.9  6032896.2  1895507.5  3347429.2 10045582.4  7248597.2
## [313]  7488471.2  7629943.1  6042572.3  1518509.5  1567225.3 10487917.0  7561326.6  7262709.1
## [321]  8459269.0  7065156.6  2175513.4  8197145.0  7269695.3  7201726.7  6472907.9  6698727.9
## [329]  5667686.3  1805457.7  4729286.5  6749008.0  6320072.4  7717661.6  9234264.2  7428004.4
## [337]  2457826.9 10555811.0 10006196.0  7952353.8  8072306.7  3251750.8  7222188.9  2123057.1
## [345]  9479107.8  8553992.4  7838233.8  7709394.8  8708028.2  7611722.2  2738259.1  9759560.5
## [353]  9487945.9  8902300.6  8930031.7 10257492.0  8155962.3  2857351.1   652542.2  6512200.7
## [361]  6814595.7  6842928.0  8035830.1  7045530.7  2145334.0   505280.1  6820057.7  7404593.7
## [369]  7283681.0  7770615.8  5655610.4  1719917.4  3401593.2  8728032.2  7773876.2  7094922.4
## [377]  6996094.8  5655475.2  1924773.1  7823617.0  7616251.0  7271742.1  6812172.4  6885921.2
## [385]  5596082.2  1819458.5  7347963.9  6600582.3  6398968.0  5926189.2  6555209.9  5955498.9
## [393]  1911514.3  7524929.7  7491055.8  7886142.6  8173076.6  8931807.1  7434563.6  2020806.6
## [401]  9868185.5  9149629.3  8074062.0  7033425.3  7301704.2  6059231.0  1972543.8  7380608.9
## [409]  5982010.2  6598916.4  7011886.4  7218439.0  7163912.2  2145529.2  8028294.3  7506031.8
## [417]  7091657.3  6663895.3  7283606.4  6152975.9  2059568.2  8065926.5  7737183.0  7264272.9
## [425]  7867347.9  9606888.4  8012018.5  2633559.6 10793763.1 10137218.4  8225675.2  7670511.2
## [433]  8241422.5  6519580.7  2140786.2  8612884.5  7532119.8  6929378.0  6421291.2  7860819.5
## [441]  6500610.1  2040849.0  8288016.0  7072724.4  6952442.5  6680275.8  7335655.1  6093044.1
## [449]  2024866.9  3829516.6  9276807.9  7479829.8  2864832.2  2077830.2  6126951.6  1799359.7
## [457]  9997138.2 10734783.0  9109931.4  8230222.4  8772731.8  6728156.9  1794215.4  8298332.9
## [465]  7718244.4  7228161.3  6757319.4  7665366.8  6905481.6  2292912.9  8898029.6  8760086.2
## [473]  7772849.5  7030226.8  8243643.9  7258300.3  2481931.3  8481686.2  7153537.3  7221872.1
## [481]  7208226.0  7964935.5  6352901.2  2304322.3  7907723.0  9375000.6  4000410.6 10042036.4
## [489] 10866159.4  7532502.0  2142311.6 10826730.7  9142704.6  8299629.7  7678764.3  9462356.7
## [497]  8203611.4  2396505.2  3576427.4  9766012.8  8033861.1  7457808.9  7918157.2  6560897.9
## [505]  2158533.0  7783438.2  5981235.8  7567613.3  6197493.3  7595541.5  6309355.5  2113557.6
## [513]  7198510.9  7886957.0  7262750.7  7445462.3  9312907.3  8450534.7  2717512.6  4655279.3
## [521] 12766038.5 11949525.1 10002477.3 10391649.8  8355321.9  2666593.5  4738287.1 11764712.2
## [529]  8869733.3  9050971.9 11821833.5 10368110.5  3421866.4 10685880.5  9465422.0  8222003.1
## [537]  7624057.1  9081307.0  6972872.1  2272084.3  8281610.5  8281622.4  7665341.4  7719037.3
## [545]  9397663.3  7925224.0  2718277.5  5677020.3 13866706.5 10450637.5  9160963.2  9840751.7
## [553]  7592222.9  2395039.4 10038907.6  8870238.3  7840748.6  6956062.7  8046026.1  7083918.1
## [561]  2267103.9  7615605.0  8695692.9  8237349.6  7908726.8  8318918.2  5342450.4  2560095.8
## [569]  9366633.3  7983002.4  7279315.1  7264004.5  7953469.3  6300788.2  2074476.9  7744144.0
```

```
## [577]  8295330.0 10093150.1  9929350.0 11369795.6  9410426.4  2625448.6 12998358.1 12894179.2
## [585]  6413917.0 10884938.7  9125900.4  7309564.9  2252089.3  9081629.9  8420543.1  7865148.2
## [593]  7494347.8  9448721.5  6802063.0  2314490.9  3855175.5 10618757.8  8119515.2  7762111.5
## [601]  8573579.2  7191506.7  2503958.8  8746232.9  8191339.4  7728205.2  8807574.6 11107618.3
## [609]  9201504.2  3639657.9 12077136.4 12109944.5 11203024.1 10479236.3 10308405.4  7875153.2
## [617]  2548740.1  9588905.0  9984227.0  9855453.2  9551353.1  9491011.4  7120767.0  2467621.5
## [625]  9481797.9  9533537.3  8456907.1  7546943.5  8491834.9  6920966.3  2333227.9  8622631.9
## [633]  8088456.9  7773169.4  8424197.9  8793350.0  6891522.9  2598720.4 10280423.8 11751475.7
## [641] 11100253.9 11024750.3 10558256.6  8718527.4  2948235.6  9838745.8  9609911.5  8971786.3
## [649]  8293481.3
```

```
class(serie_ts)
```

```
## [1] "ts"
```

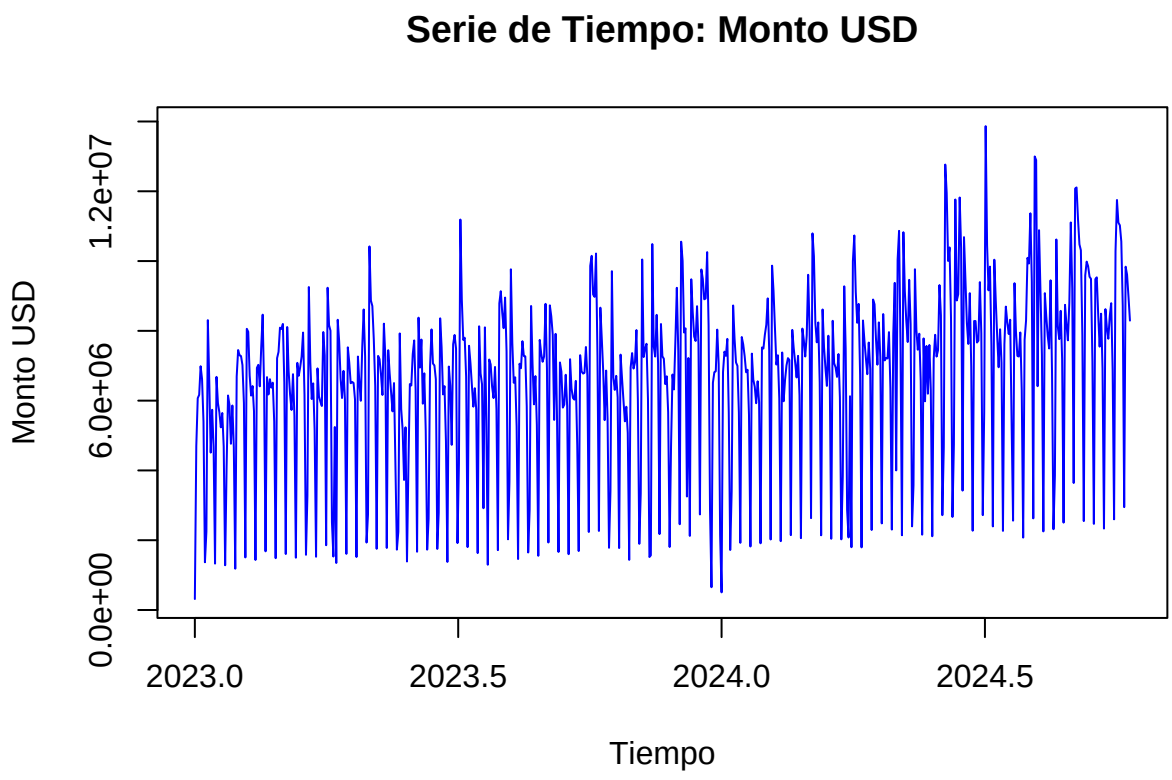
```
start(serie_ts)
```

```
## [1] 2023    1
```

```
end(serie_ts)
```

```
## [1] 2024  284
```

Se tiene entonces que la serie finaliza el dia 284 del año 2024.



Serie de tiempo usando xts.Si las fechas no son consecutivas o hay algunos días sin datos, se recomienda usar xts() del paquete xts

```
# Instalar y cargar el paquete xts
#install.packages("xts")
```

```r
#install.packages("zoo")
library(zoo)
library(xts)
```

```
## Warning: package 'xts' was built under R version 4.3.3
```

```
##
## ############################ Warning from 'xts' package ##########################
## #                                                                                #
## # The dplyr lag() function breaks how base R's lag() function is supposed to  #
## # work, which breaks lag(my_xts). Calls to lag(my_xts) that you type or       #
## # source() into this session won't work correctly.                               #
## #                                                                                #
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can add #
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop           #
## # dplyr from breaking base R's lag() function.                                   #
## #                                                                                #
## # Code in packages is not affected. It's protected by R's namespace mechanism #
## # Set `options(xts.warn_dplyr_breaks_lag = FALSE)` to suppress this warning.  #
## #                                                                                #
## ##################################################################################
```

```
##
## Attaching package: 'xts'
```

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

```r
# Crear la serie de tiempo con fechas irregulares
serie_xts <- xts(Datos_agg$Monto_USD, order.by = Datos_agg$Fecha_Pago)
(serie_xts)
```

```
##            m.c.seq.row..seq.n...seq.col..drop...FALSE.
## 2023-01-01                                    314473.1
## 2023-01-02                                   4700346.1
## 2023-01-03                                   6069974.1
## 2023-01-04                                   6164948.9
## 2023-01-05                                   6986643.0
## 2023-01-06                                   6616416.0
## 2023-01-07                                   5392530.2
## 2023-01-08                                   1368008.4
## 2023-01-09                                   2248875.5
## 2023-01-10                                   8307518.8
##        ...
## 2024-10-01                                  11751475.7
## 2024-10-02                                  11100253.9
## 2024-10-03                                  11024750.3
## 2024-10-04                                  10558256.6
## 2024-10-05                                   8718527.4
## 2024-10-06                                   2948235.6
## 2024-10-07                                   9838745.8
## 2024-10-08                                   9609911.5
## 2024-10-09                                   8971786.3
## 2024-10-10                                   8293481.3
```

```r
plot(serie_xts, col = "blue", main = "Serie de Tiempo: Monto USD", ylab = "Monto USD")
```

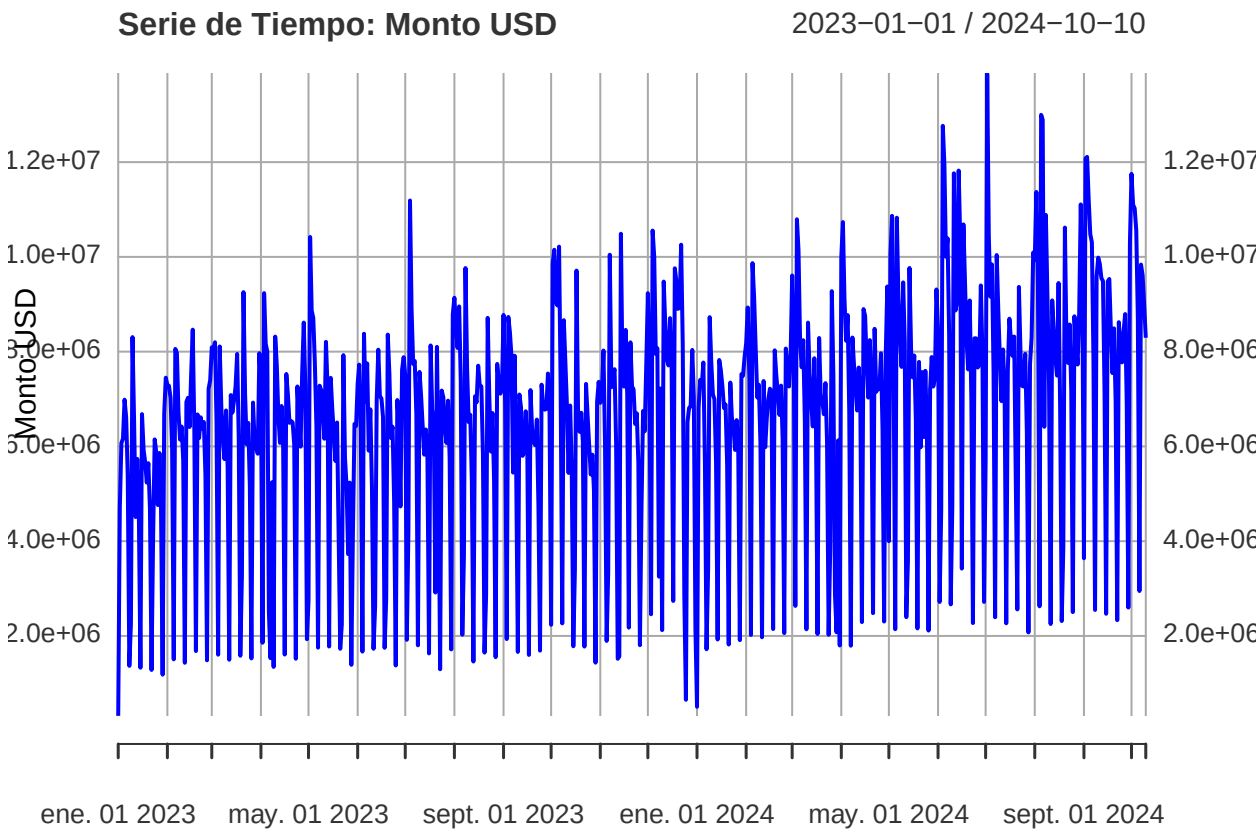
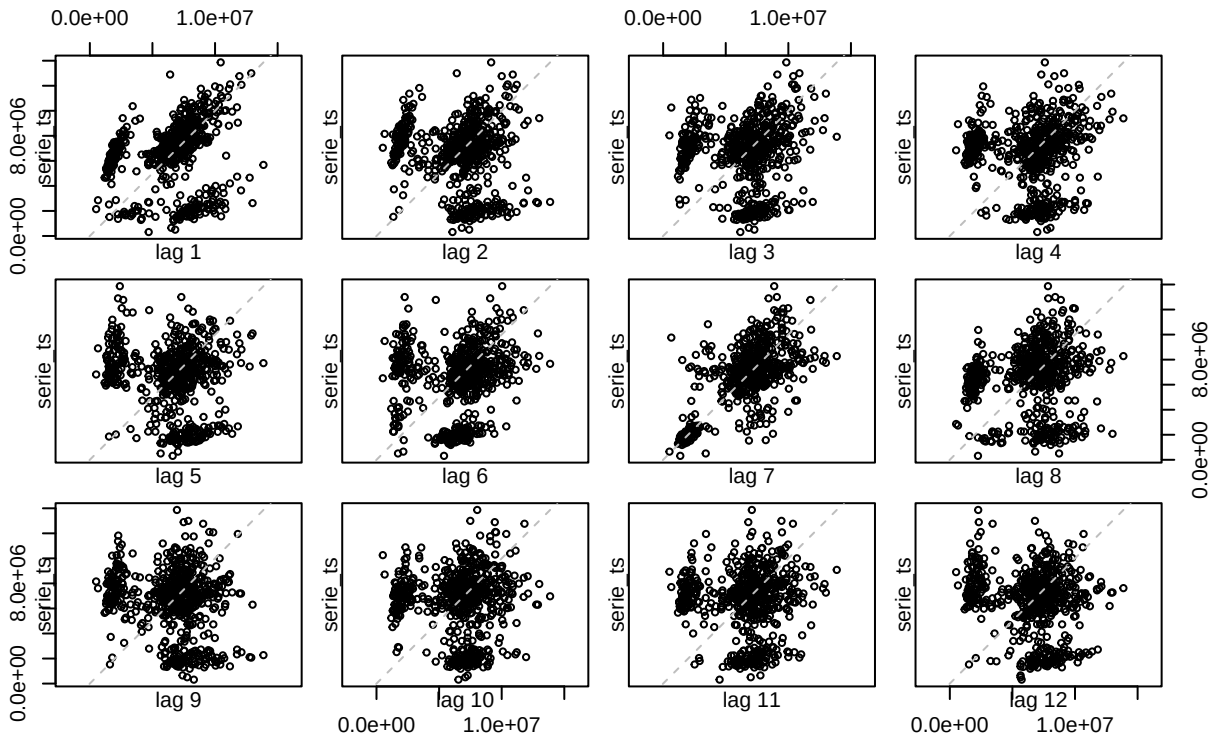


Gráfico de resago para la serie ts

```r
# Gráfico de rezago para la serie 'ts'
lag.plot(serie_ts, lags = 12, layout = c(3, 4), main = "Gráfico de Rezago para la Serie de Tiempo")
```



Según la gráfica no hay comportamiento que se repita en un cierto periodo.

Aproximación en promedio móvil

Promedio Móvil Simple (SMA)

El promedio móvil simple es una técnica estadística utilizada para suavizar series de tiempo mediante la creación de un promedio de los valores en un intervalo específico de tiempo. Permite facilitar la identificación de tendencias subyacentes al hacer que las variaciones estacionales y aleatorias sean menos evidentes.

Una de sus ventajas es que proporciona una visión clara de la tendencia general de los datos a lo largo del tiempo.

SMA se usa principalmente en análisis financiero, como en el seguimiento de precios de acciones, proyecciones de ventas, y otras métricas de negocio.

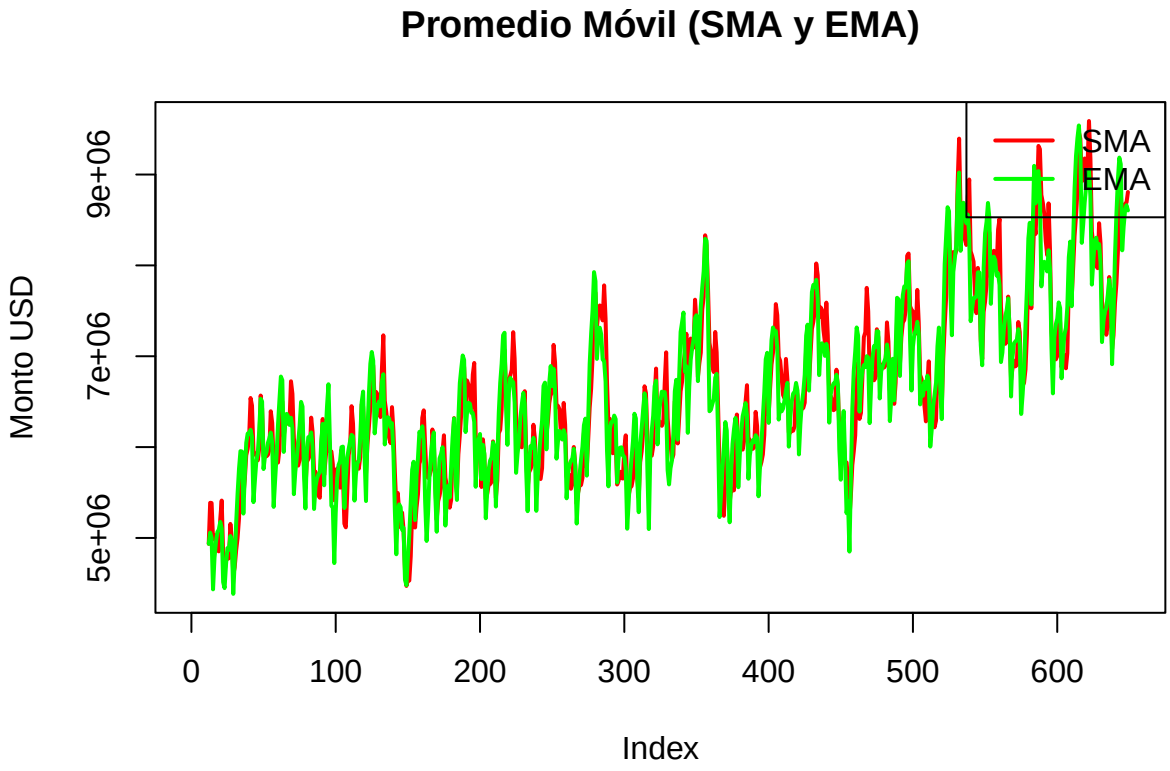
Promedio Móvil Exponencial (EMA).

Es similar al promedio móvil simple (SMA), pero con un enfoque que da más peso a los datos más recientes.Este puede ser más representativo de la tendencia actual en series de tiempo con alta volatilidad.

Se aplica en diversas áreas de análisis de datos, como en el monitoreo de ventas, pronósticos de demanda, y otros indicadores de rendimiento.

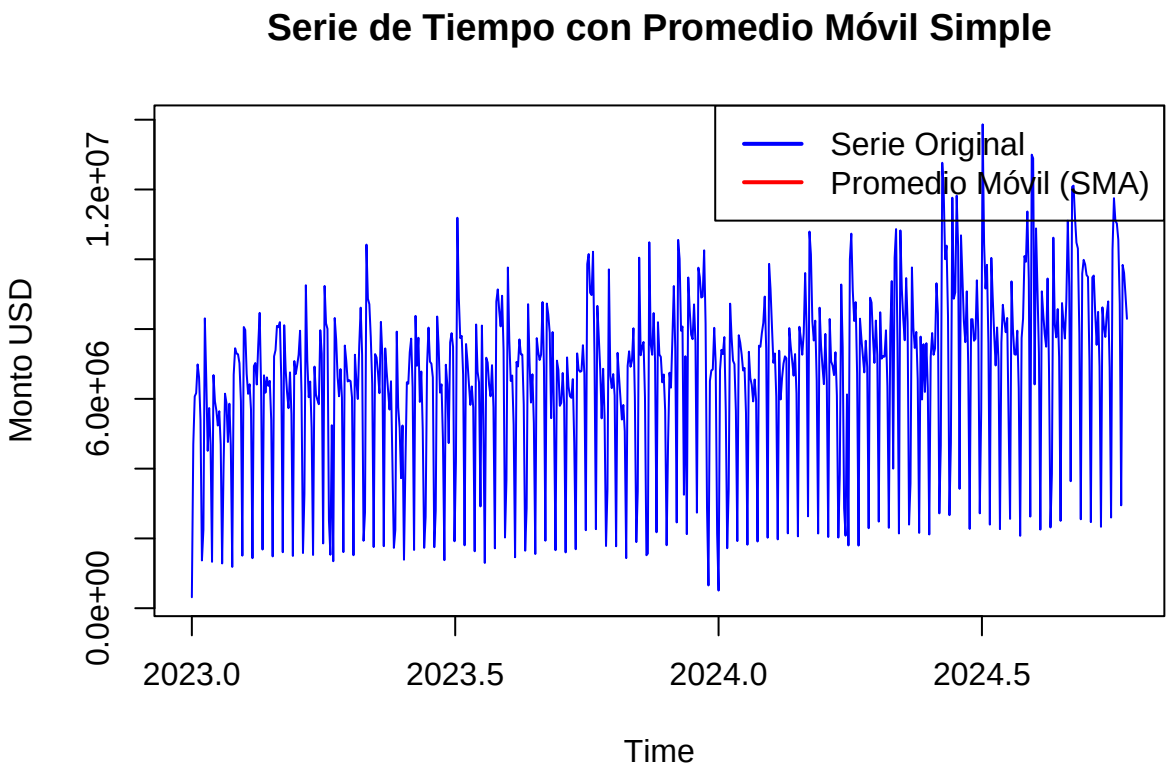
```
# Graficar solo los promedios móviles, sin la serie original
plot(sma_serie_ts, type = "l", col = "red", lwd = 2, ylim = range(c(sma_serie_ts, ema_serie_ts), na.rm =
     main = "Promedio Móvil (SMA y EMA)", ylab = "Monto USD")
```

```r
# Agregar la línea del EMA
lines(ema_serie_ts, col = "green", lwd = 2)

# Añadir una leyenda para identificar las líneas
legend("topright", legend = c("SMA", "EMA"), col = c("red", "green"), lty = 1, lwd = 2)
```



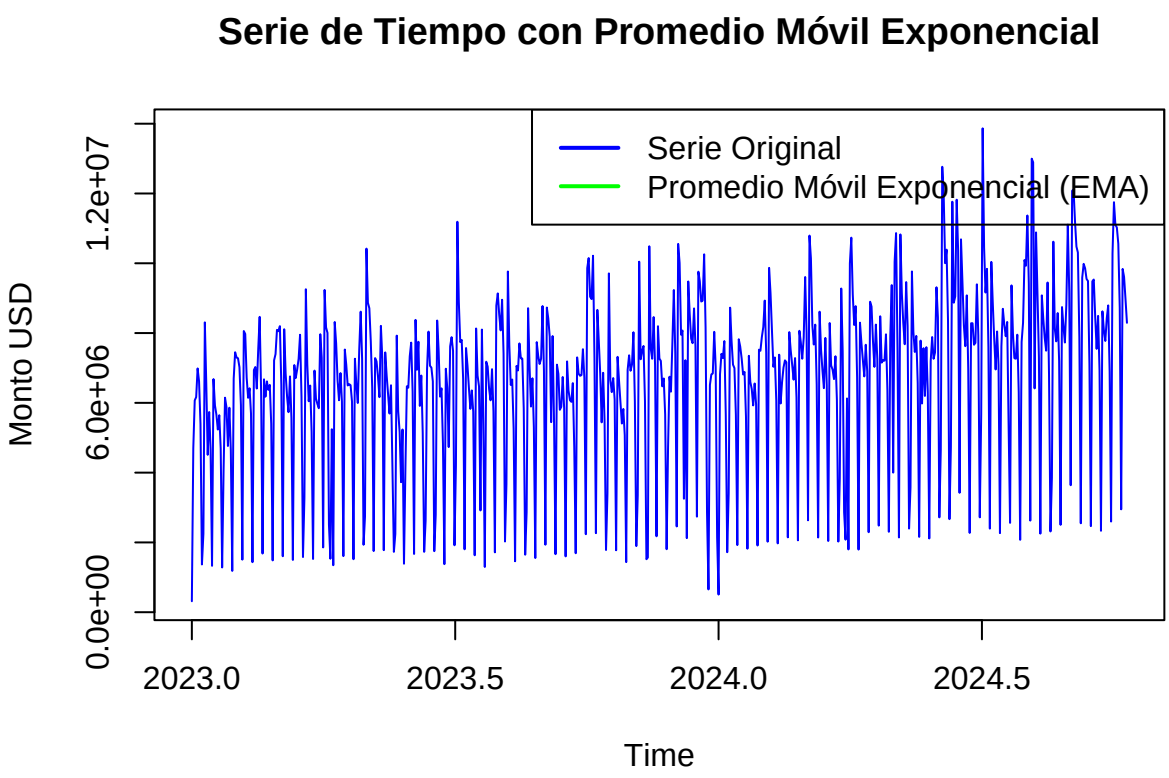
Promedio Móvil Simple (SMA) con serie de tiempo original

```r
library(TTR)

# Promedio Móvil Simple para la serie 'ts'
sma_serie_ts <- SMA(serie_ts, n = 12)  # Aquí n = 12 indica la ventana de 12 periodos (por ejemplo, men

# Graficar la serie original y el promedio móvil
plot(serie_ts, type = "l", col = "blue", main = "Serie de Tiempo con Promedio Móvil Simple", ylab = "Mo
lines(sma_serie_ts, col = "red", lwd = 2)  # Agregar la línea del promedio móvil
legend("topright", legend = c("Serie Original", "Promedio Móvil (SMA)"), col = c("blue", "red"), lty =
```



Promedio Móvil Exponencial (EMA) con serie de tiempo original

```r
# Promedio Móvil Exponencial para la serie 'ts'
ema_serie_ts <- EMA(serie_ts, n = 12)  # Usamos una ventana de 12 periodos

# Graficar la serie original y el promedio móvil exponencial
plot(serie_ts, type = "l", col = "blue", main = "Serie de Tiempo con Promedio Móvil Exponencial", ylab =
lines(ema_serie_ts, col = "green", lwd = 2)  # Agregar la línea del promedio móvil exponencial
legend("topright", legend = c("Serie Original", "Promedio Móvil Exponencial (EMA)"), col = c("blue", "g
```



Al usar la serie de tiempo original, los promedios moviles tanto simple, como exponencial no son visibles, esto sugiere que la serie temporal puede tener poca volatilidad, lo que significa que los valores no cambian drásticamente de un período a otro. Esto resulta en promedios móviles que siguen de cerca la tendencia general.

Estacionalidad.

La estacionalidad se refiere a patrones que se repiten en intervalos regulares de tiempo (por ejemplo, mensualmente, trimestralmente, etc.)

Descomposición de Series de Tiempo

Esta técnica separa la serie en tres componentes: tendencia, estacionalidad y error

```
# Descomponer la serie de tiempo
#serie_decomp <- stl(serie_ts, s.window = "periodic")

# Graficar la descomposición
#plot(serie_decomp)
```

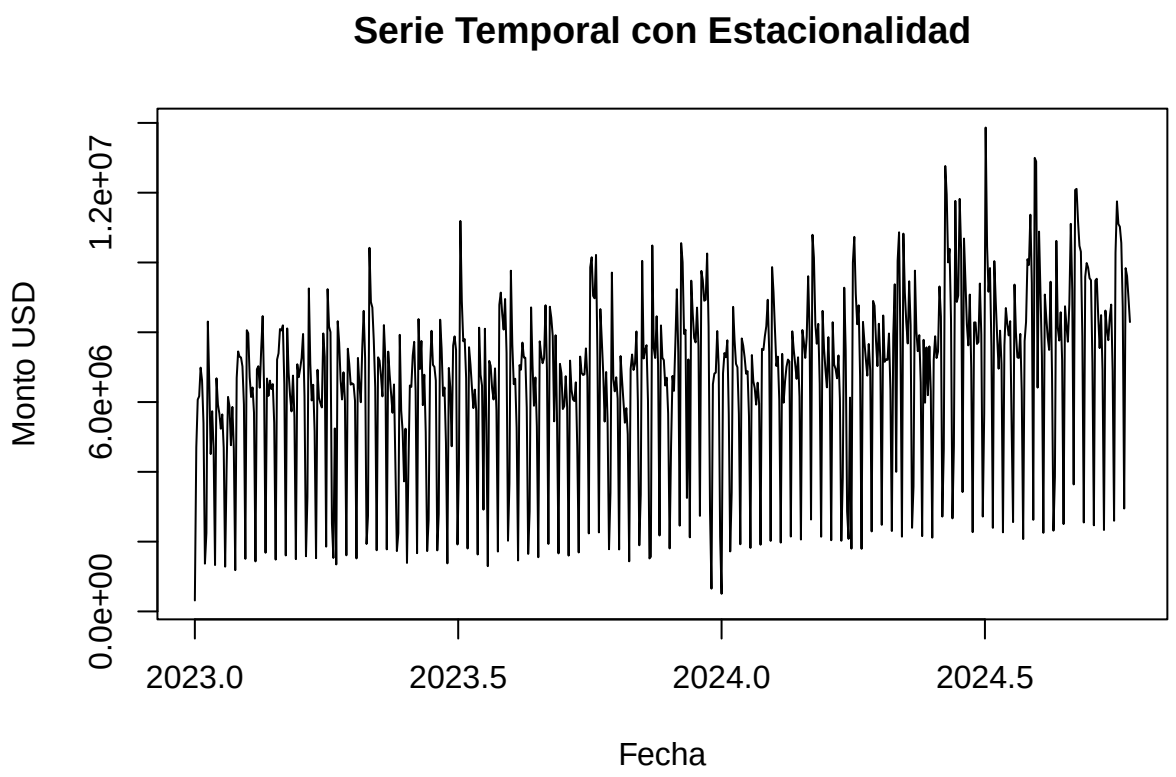
La ejecucion del codigo anterior generar el siguiente error:

Error in stl(serie_ts, s.window = “periodic”) : series is not periodic or has less than two periods

En este caso, el error se debe a que en el set de datos, es necesario al menos dos ciclos completos de los datos estacionales.

Gráfico de Serie Temporal

```r
# Gráfico de la serie temporal
plot(serie_ts, main = "Serie Temporal con Estacionalidad", ylab = "Monto USD", xlab = "Fecha")
```



La serie tiene un patrón claro que se repite a lo largo del tiempo, eso es una indicación de estacionalidad.