Pronósticos del índice Nacional de Precios al Consumidor (INPC)

Proyecto

Pronósticos del INPC con modelos SARIMA, VAR/VEC y MFD

Tema: Econometría
Escribe: Martha Aguilar Jiménez

Fecha: Mayo de 2023

Resumen

El Índice Nacional de Precios al Consumidor se usa como medida de inflación, es por ello por lo que tiene gran importancia en la economía del país. En este sentido, se abordan los modelos SARIMA, VAR/VEC y de Factores Dinámicos, con el objetivo de pronosticar doce periodos de este índice, correspondientes al año 2022, a través de datos (y variables) del Indicador Oportuno de la Actividad Económica en México (IOAE). A lo largo de este análisis, se presenta el desarrollo de cada uno de los modelos mencionados con anterioridad para compararlos a partir del cálculo de sus respectivos errores de predicción mediante el error cuadrático medio (MSE por sus siglas en inglés); para con ello, finalmente elegir el mejor modelo y concluir si a mayor cantidad de información valiosa en los modelos ajustados, se obtiene una mejor precisión en los pronósticos.

Índice

1. Estado del arte

1.1 Objetivo general

El objetivo de este trabajo es generar pronósticos de doce periodos correspondientes al año 2022 del Índice Nacional de Precios al Consumidor (INPC), a través de variables del Indicador Oportuno de la Actividad Económica en México (IOAE).

1.2 Antecedentes

Un Índice de Precios al Consumidor es un indicador económico que mide, a lo largo del tiempo, la variación promedio de los precios de una canasta de bienes y servicios representativa del consumo de los hogares del país.

El INPC se ha consolidado como uno de los principales indicadores del desempeño económico del país; sus aplicaciones son numerosas y de gran importancia en los ámbitos económico, jurídico y social. La estimación de su evolución en el tiempo permite contar con una medida de la inflación general en el país, la cual es confiable y oportuna gracias a la aplicación de una metodología basada en las recomendaciones de buenas prácticas internacionales, y la sistematización y mejora continua de los procesos.

Además de su utilización como medida de la inflación, cabe destacar su uso como:

- Factor de actualización de los créditos fiscales.
- Determinante del valor de la Unidad De Inversión (UDI).
- Factor para actualizar la Unidad de Medida y Actualización (UMA).
- Referente en negociaciones contractuales.
- Factor de actualización de valores nominales y como deflactor del Sistema de Cuentas Nacionales de México.
- Auxiliar en la determinación de los incrementos salariales, los montos de las jubilaciones y las prestaciones de seguridad social.
- Auxiliar en el cálculo de pagos de intereses, montos de alquiler, contratos privados y precios de los bonos que suelen estar indexados al INPC.
- Auxiliar para las autoridades financieras y hacendarias del país en el diseño y evaluación de las políticas monetarias y fiscales, orientadas a procurar la estabilidad del poder adquisitivo de la moneda nacional y unas finanzas públicas sanas.
- Herramienta estadística para empresas e investigadores.

La elaboración y publicación del INPC se inició en 1969 por parte del Banco de México. A partir de esa fecha el INPC se ha actualizado en seis ocasiones, por lo que el Cambio de Año Base de 2010 a 2018, fue la séptima actualización. La sexta actualización, realizada en 2013, consideró solamente el cambio de la estructura de ponderaciones con la información reportada por la ENIGH 2010.

2. Marco teórico

2.1 Modelo SARIMA

Para explicar los modelos SARIMA, es necesario abordar los modelos autorregresivos y de medias móviles.

AR: Autorregresivos

Estos modelos imitan directamente a los modelos tradicionales de regresión, expresando a Y_t como una función lineal de sus valores en el pasado. Es de esperarse entonces, que una gran parte de las propiedades estadísticas que se conocen para el caso de regresión se apliquen directamente en este caso.

Un modelo AR(p) puede ser escrito como

$$Y_t = \phi_1 Y_{t-1} + ... + \phi_p Y_{t-p} + Z_t$$

Entonces, Y_t se dice que sigue un proceso $AR(p)$ si

- Es estacionario.
- Satisface $\phi(B)Y_t = Z_t$, para toda t.
- Con media μ si $Y_t - \mu$ sigue un proceso $AR(p)$.

MA: Promedios Móviles

Es un modelo estacionario de series temporales que describe la dependencia lineal entre una observación y un término de error residual generado por una media móvil de orden q. En otras palabras, el modelo $MA(q)$ se utiliza para describir la relación entre una observación actual y los errores pasados del modelo y para modelar la autocorrelación en los residuos de un modelo.

El modelo MA(q) se representa

$$Y_t = \theta(B)Z_t$$

donde $\theta(B) = 1 + \theta_1 B + ... + \theta_q B^q$ con $BZ_t = Z_{t-1}$

ARMA: Autorregresivos de Promedios Móviles

Una representación parsimoniosa es siempre deseable y ésta se consigue a través de los modelos ARMA.

Y_t se dice que sigue un proceso $ARMA(p, q)$ si

- Es estacionario.
- Para toda t, $\psi(B)Y_t = \theta(B)Z_t$, donde $Z_t \backsim WN(0, \sigma^2)$
- Con media μ si $Y_t - \mu$ sigue un proceso $ARMA(p, q)$.

Modelo SARIMA

El modelo SARIMA es un modelo autorregresivo estacional multiplicativo integrado de media móvil que se basa en la aplicación de los modelos ARMA (integrados) a una serie temporal transformada donde se ha eliminado el comportamiento estacional y no estacionario.

Formalmente se expresa

$$\Phi_P(B^s)\phi(B)\nabla_s^D \nabla^d x_t = \alpha + \Theta_Q(B^s)\theta(B)w_t$$

donde

$\phi(B)$ y $\theta(B)$ son las componentes autorregresivas y de media móvil de órdenes p y q, respectivamente.

$\Phi(B)$ y $\Theta(B)$ son las componentes autorregresivas y de medias móviles estacionales.

$\nabla^d = (1-B)^d$ y $\nabla_s^D = (1-B^s)^D$ son las diferencias ordinarias y estacionales, respectivamente.

El modelo general, simplemente se denota como

$$SARIMA(p,d,q) \times (P,D,Q)s$$

2.2 Modelo VAR

Un Vector Autorregresivo (VAR) es la extensión natural de los modelos AR(p) desde una perspectiva multivariada, se utiliza cuando se buscan interacciones simultáneas entre dos o más variables, que tienen el mismo grado de integración, pero que no se encuentran cointegradas, y así, el modelo VAR es un marco general para describir la interrelación dinámica entre variables estacionarias.

El modelo se representa

$$Y_t = \sum_{i=1}^{p} \Phi_i Y_{t-i} + \epsilon_t$$

2.3 Modelo VEC

El modelo de Vectores de Corrección de Error (VEC) es un tipo de VAR definido para variables que no son estacionarias y que están cointegradas, en éste, las variables son estacionarias en primeras diferencias. La aplicación del modelo VEC significa la existencia de una relación estable de equilibrio en el largo plazo entre las variables; sin embargo, ello no significa necesariamente que también lo exista en el corto plazo.

Este modelo permite examinar qué tanto cambiará la variable de respuesta a un cambio en la variable explicativa (la parte de cointegración y_t), como también de la velocidad de cambio (la parte del error de corrección Δy_t).

El modelo VEC es de la forma

$$\Delta Y_t = \Pi Y_{t-1} + \sum_{i=1}^{p-1} \Phi_i^* \Delta Y_{t-i} + \epsilon_t$$

2.4 Modelo de Factores Dinámicos (MFD)

El modelo de factores dinámicos generaliza el análisis factorial, apoyándose en su filosofía, técnicas de estimación para los parámetros e interpretación.

El objetivo de los MFD es representar la dinámica del sistema a través de un pequeño número de factores comunes ocultos que son utilizados principalmente para la elaboración de pronósticos y políticas macroeconómicas.

Además, el MFD puede simplificar el problema de predicción para una variable de interés a partir de un vector de covariables de alta dimensión modelizando las dinámicas de cada serie en términos de un número reducido de factores latentes inobservables más un término de error de media cero correspondiente al movimiento idiosincrático de cada serie.

El modelo se puede escribir

$$y_{it} = P_i F_t + \epsilon_{it}$$

3. Contexto de los datos

El Indicador Oportuno de la Actividad Económica (IOAE) en México, es un índice económico que busca medir la evolución de la actividad económica del país en tiempo real, para esto se usan datos de exportaciones no petroleras, índices de producción industrial, muestra representativa de ventas al por menor de la ANTAD, índices de la Bolsa Mexicana de Valores, índices de confianza empresarial y del consumidor, remesas, y ventas de vehículos; estos datos son proporcionados por diferentes instituciones o dependencias como el Instituto Nacional de Estadística y Geografía (INEGI), la Secretaría de Energía (SENER), Instituto Mexicano del Seguro Social (IMSS), la Asociación Nacional de Tiendas de Autoservicio y Departamentales (ANTAD) o el Banco de México.

3.1 Variables

La información que se utiliza en este estudio se describe a continuación:

- IAI: Índice de producción industrial.
- TDU: Tasa de desocupación.
- M: Importaciones totales.
- CONF_MAN: Indicador de confianza manufacturera.
- IPC: Índice de precios y cotizaciones de la Bolsa Mexicana de Valores.
- TC: Tipo de cambio nominal promedio.
- TIIE_28: Tasa de interés interbancaria de equilibrio a 28 días.
- SP_500: Índice Standard & Poor's.
- IPI_EUA: Índice de producción industrial de los Estados Unidos.
- ANTAD: Ventas totales de la ANTAD.
- PROD_VEH: Producción de vehículos automotores.
- OCUP_HOT: Ocupación hotelera en corredores y agrupamientos.
- GASOLINAS: Demanda de combustibles.
- IMSS: Asegurados permanentes y eventuales del Seguro Social.
- REMESAS: Remesas familiares.
- M4: Agregado monetario M4.
- X: Exportaciones totales.
- PEDIDOS_MANU: Indicador de pedidos manufactureros.
- IGAE: Indicador Global de la Actividad Económica.
- IMEF: Indicador del Instituto Mexicano de Ejecutivos de Finanzas.
- INPC: Índice Nacional de Precios al Consumidor.
- PRECIO: Precio del petróleo, mezcla mexicana.
- U_US: Tasa de desempleo abierto, U3 (Estados Unidos).

- TOTAL_NONFARM: Total de empleo, no agrícola, en miles, Estados Unidos.
- TOTAL_CONSTR: Total de empleo, construcción, en miles, Estados Unidos.
- TOTAL_MANUF: Total de empleo, manufacturas, en miles, Estados Unidos.
- TOTAL_SERV: Total de empleo, servicios, en miles, Estados Unidos.
- MANUF_USA: Índice de manufacturas EUA.

3.2 Marco conceptual

En este apartado se abordarán los conceptos relacionados con la información utilizada en el análisis.

- **Agregado monetario M4:** Agregado monetario compuesto por instrumentos altamente líquidos en poder de los sectores residentes tenedores de dinero (M1), más los instrumentos monetarios a plazo en poder de los sectores residentes tenedores de dinero (M2), más los valores públicos en poder de los sectores residentes tenedores de dinero y que fueron emitidos por el Gobierno Federal, Banco de México (BREMS) y el IPAB (M3), más la tenencia por parte de no residentes de todos los instrumentos incluidos en M3 (M4).
- **ANTAD:** Es la Asociación Nacional de Tiendas de Autoservicio y Departamentales, la cual está conformada por 93 cadenas, de las cuales 25 son de autoservicio, 13 departamentales y 55 especializadas.
- **Indicador Global de la Actividad Económica (IGAE):** Es un indicador que permite conocer y dar seguimiento a la evolución mensual del sector real de la economía. Para su cálculo se utilizan: el esquema conceptual, los criterios metodológicos, la clasificación de actividades económicas y las fuentes de información, que se emplean en los cálculos anuales y trimestrales del Producto Interno Bruto. Este cálculo se alinea con las cifras anuales utilizando la técnica Denton. Incorpora a las Actividades Primarias, Secundarias y Terciarias, a excepción de: la pesca, el aprovechamiento forestal, los corporativos y otras actividades de servicios.
- **Indicador IMEF:** Es un indicador de difusión que busca medir el clima empresarial en torno al ambiente económico, es decir, sirve como un indicador económico adelantado, que anticipa la trayectoria o dirección de la actividad económica en el muy corto plazo. Es el primer indicador del sector privado mexicano que cuenta con el apoyo técnico y normativo del INEGI.
- **Indicador Oportuno de la Actividad Económica (IOAE):** Muestra estimaciones oportunas mensuales de la actividad económica de México obtenidas a través de modelos de nowcasting, los cuales generan estimaciones de las variaciones porcentuales anuales y niveles del Indicador Global de la Actividad Económica (IGAE). De esta forma, se obtienen estimaciones suficientemente precisas que ayudan a adelantar las señales económicas.
- **Índice:** Es un indicador que tiene por objeto medir las variaciones de un fenómeno económico o de otro orden referido a un valor que se toma como base en un momento dado. Relación de precios, de cantidades, de valores entre periodos dados.
- **Índice Nacional de Precios al Consumidor (INPC):** Indicador económico que mide a través del tiempo, la variación de precios de una canasta fija de bienes y servicios representativa del consumo de los hogares del país. Es elaborado y difundido quincenal y mensualmente por el INEGI.
- **Industria manufacturera:** Es el conjunto de unidades económicas que, en una ubicación única, delimitada por construcciones e instalaciones fijas, combina recursos bajo un solo propietario o control para desarrollar por cuenta propia o ajena (maquila) actividades de ensamble, procesamiento y transformación total o parcial de materias primas que derivan en la producción de bienes nuevos y servicios afines.

- **Tasa de interés interbancaria de equilibrio (TIIE):** Se determina por el Banco de México con base en cotizaciones presentadas por las instituciones de crédito, teniendo como fecha de inicio la publicación en el Diario Oficial de la Federación. El procedimiento de cálculo de dicha tasa se establece en el Título Tercero, Capítulo IV, de la Circular 3/2012 emitida por el Banco de México y el Diario Oficial de la Federación del 2 de marzo de 2012.
- **Tipo de cambio:** Es una referencia que se usa en el mercado cambiario para conocer el número de unidades de moneda nacional que deben pagarse para obtener una moneda extranjera, o similarmente, el número de unidades de moneda nacional que se obtienen al vender una unidad de moneda extranjera.

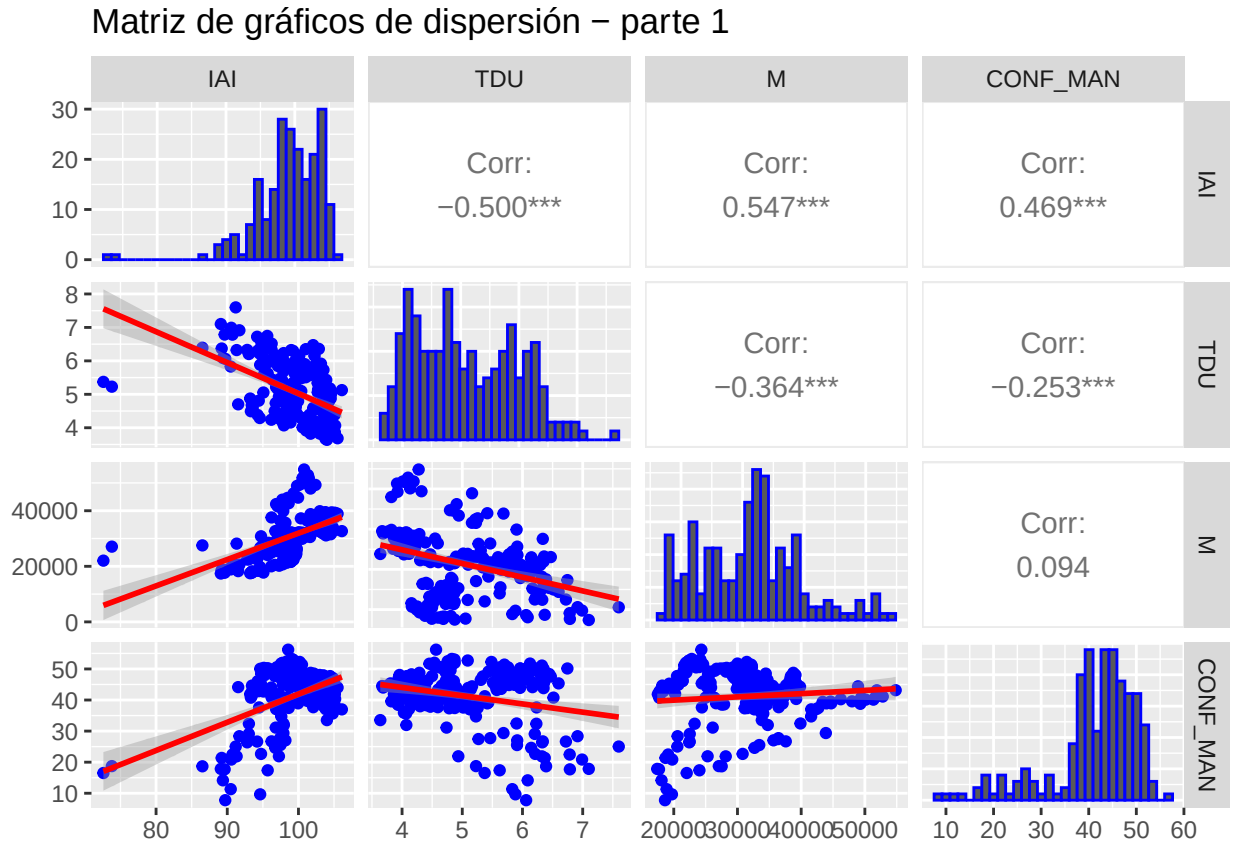
4. Análisis exploratorio de los datos

Antes de abordar el análisis de series de tiempo, se realiza un análisis exploratorio para ver cómo se comportan los datos y así contar con un panorama general de las variables.

4.1 Gráficos de dispersión

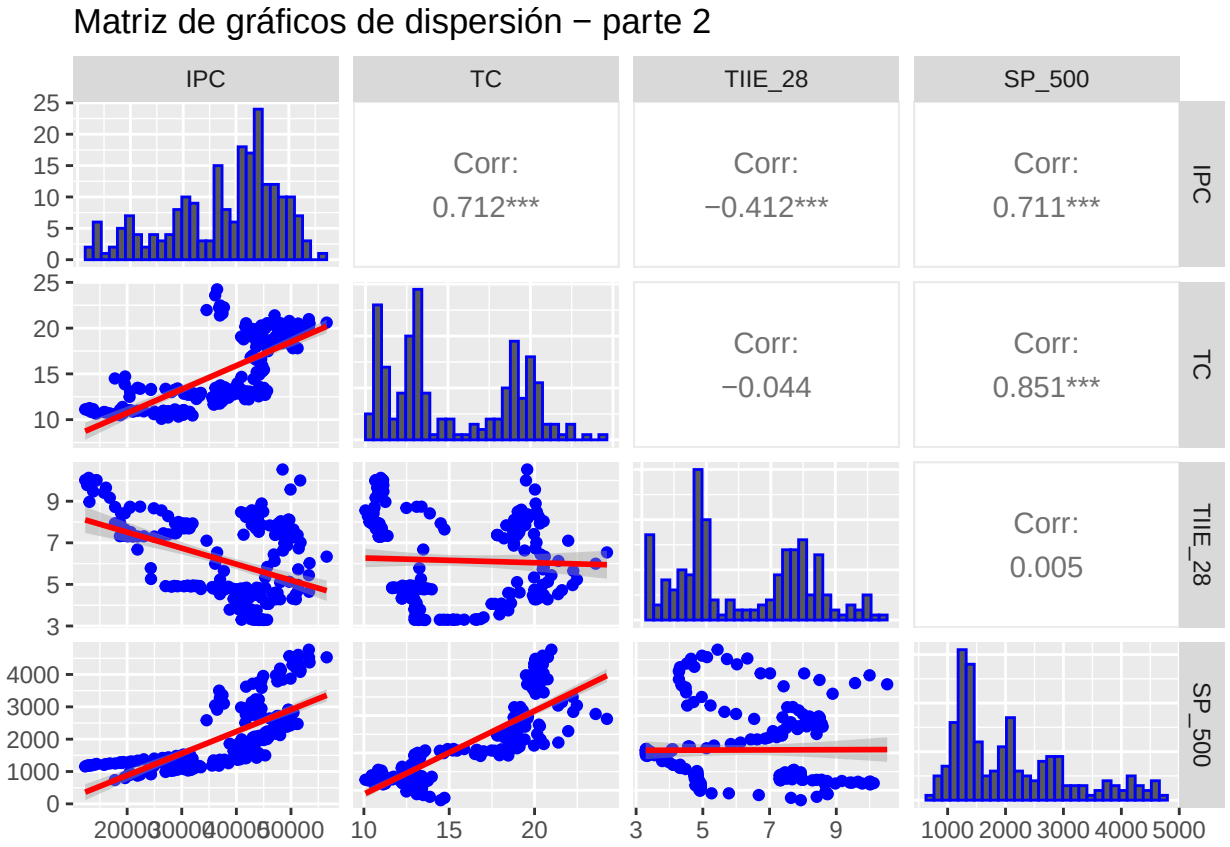
Como parte del análisis exploratorio, se muestran gráficos de dispersión, los cuales, fueron generados a partir de una función donde se incluye la recta de regresión entre variables y se organizan por partes, para una mejor visualización.

```
####################################
# Análisis exploratorio de los datos
####################################

# Limpiar espacio de trabajo
rm(list = ls())
invisible(gc(reset = TRUE))

# Se cargan las librerías
library(ggplot2)  # Gráficos avanzados
library(GGally)   # Gráficos de dispersión
library(car)      # Modelos de regresión
library(corrplot) # Matriz gráfica de correlaciones

# Se cargan los datos de interés
datos<-read.csv("C:/Proyecto de series/IOAE.csv")

# Función para gráficos de dispersión
lowerFn<-function(data, mapping, method = "lm", ...){
p<-ggplot(data = data, mapping = mapping) +
   geom_point(colour = "blue") +
   geom_smooth(method = method, color = "red", ...)
p}

# Matriz de gráficos de dispersión parte 1
ggpairs(datos[, 2:5], lower = list(continuous = wrap(lowerFn, method = "lm")),
        diag = list(continuous = wrap("barDiag", colour = "blue")),
        upper = list(continuous = wrap("cor", size = 4)),
        title = "Matriz de gráficos de dispersión - parte 1")
```



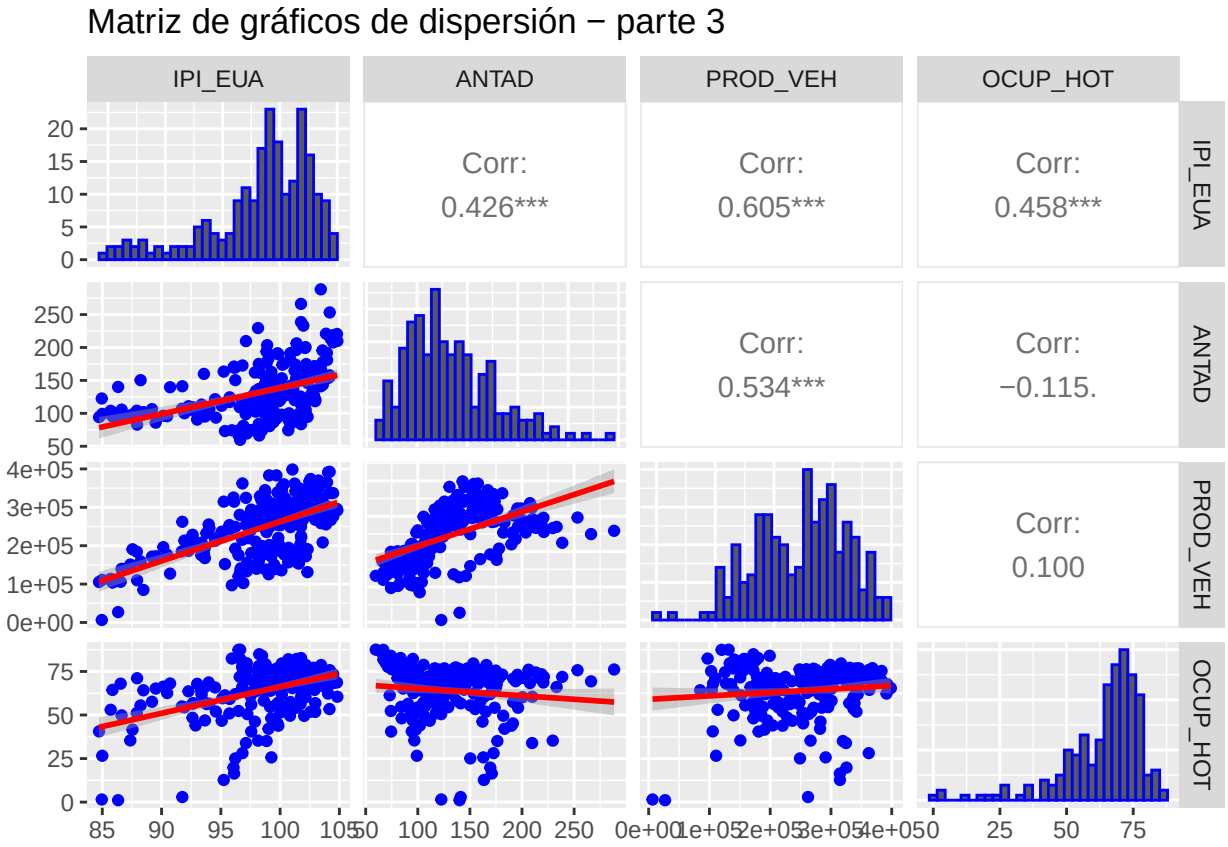
De acuerdo con la salida anterior, se aprecia una correlación positiva importante entre las importaciones totales (M) y el índice de producción industrial (IAI); por otro lado, existe una correlación negativa entre el índice de producción industrial (IAI) y la tasa de desocupación (TDU). En lo que respecta a la dispersión, los datos IAI presentan un sesgo izquierdo o negativo, mientras que, los datos TDU muestran un ligero sesgo positivo. Particularmente, las observaciones del indicador de confianza manufacturera (CONF_MAN), se ven con un sesgo negativo importante.

```
# Matriz de gráficos de dispersión parte 2
ggpairs(datos[, 6:9], lower = list(continuous = wrap(lowerFn, method = "lm")),
        diag = list(continuous = wrap("barDiag", colour = "blue")),
        upper = list(continuous = wrap("cor", size = 4)),
        title = "Matriz de gráficos de dispersión - parte 2")
```



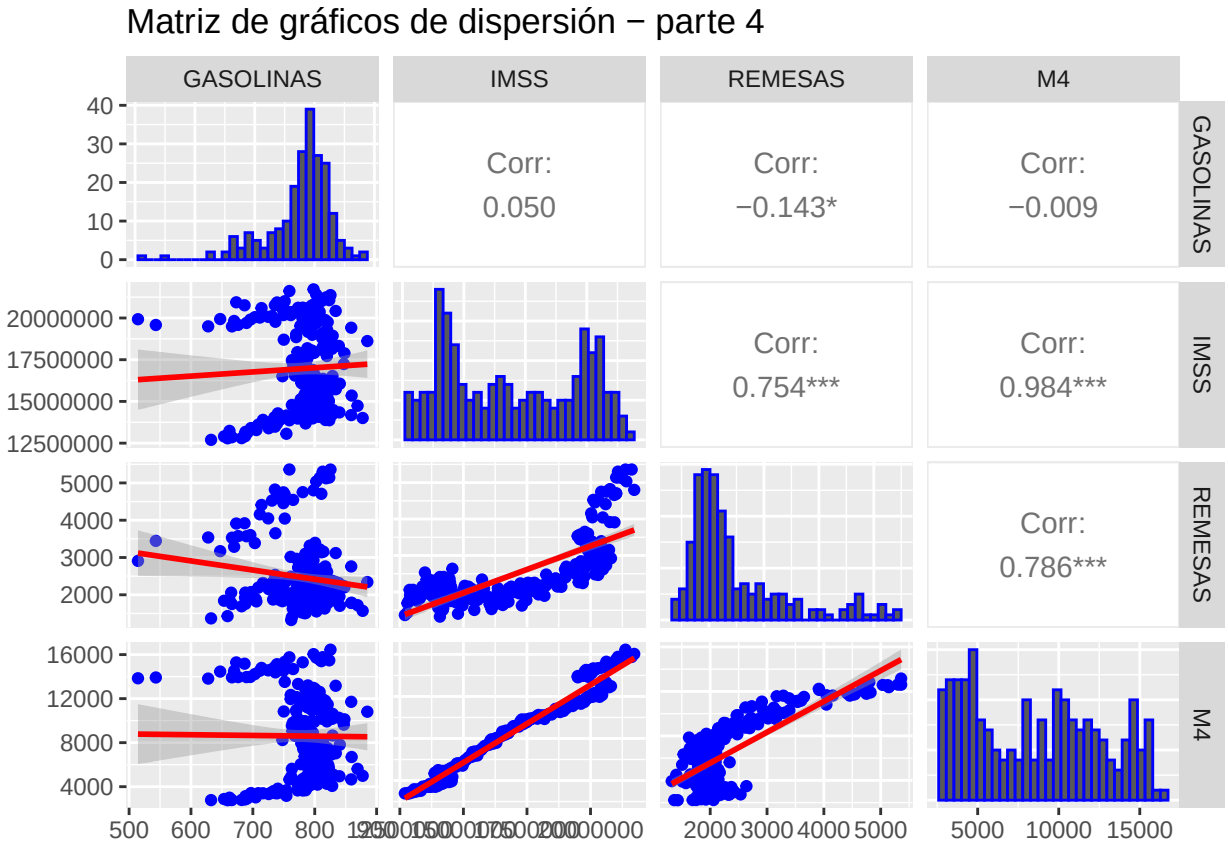
En los gráficos previos, pudiera parecer la existencia de una tendencia lineal entre el índice de precios y cotizaciones (IPC) y el índice Standard & Poor's (SP_500), así como entre el tipo de cambio (TC) y el SP_500; sin embargo, entre las variables TC y TIIE_28 o TIIE_28 y SP_500 se aprecia un patrón un poco extraño con cierta aleatoriedad. Con respecto a la correlación lineal, la más alta ocurre entre las variables SP_500 y TC, con un coeficiente de 0.851 y la más baja entre las variables SP_500 y TIIE_28 con 0.005.

```
# Matriz de gráficos de dispersión parte 3
ggpairs(datos[, 10:13], lower = list(continuous = wrap(lowerFn, method = "lm")),
        diag = list(continuous = wrap("barDiag", colour = "blue")),
        upper = list(continuous = wrap("cor", size = 4)),
        title = "Matriz de gráficos de dispersión - parte 3")
```



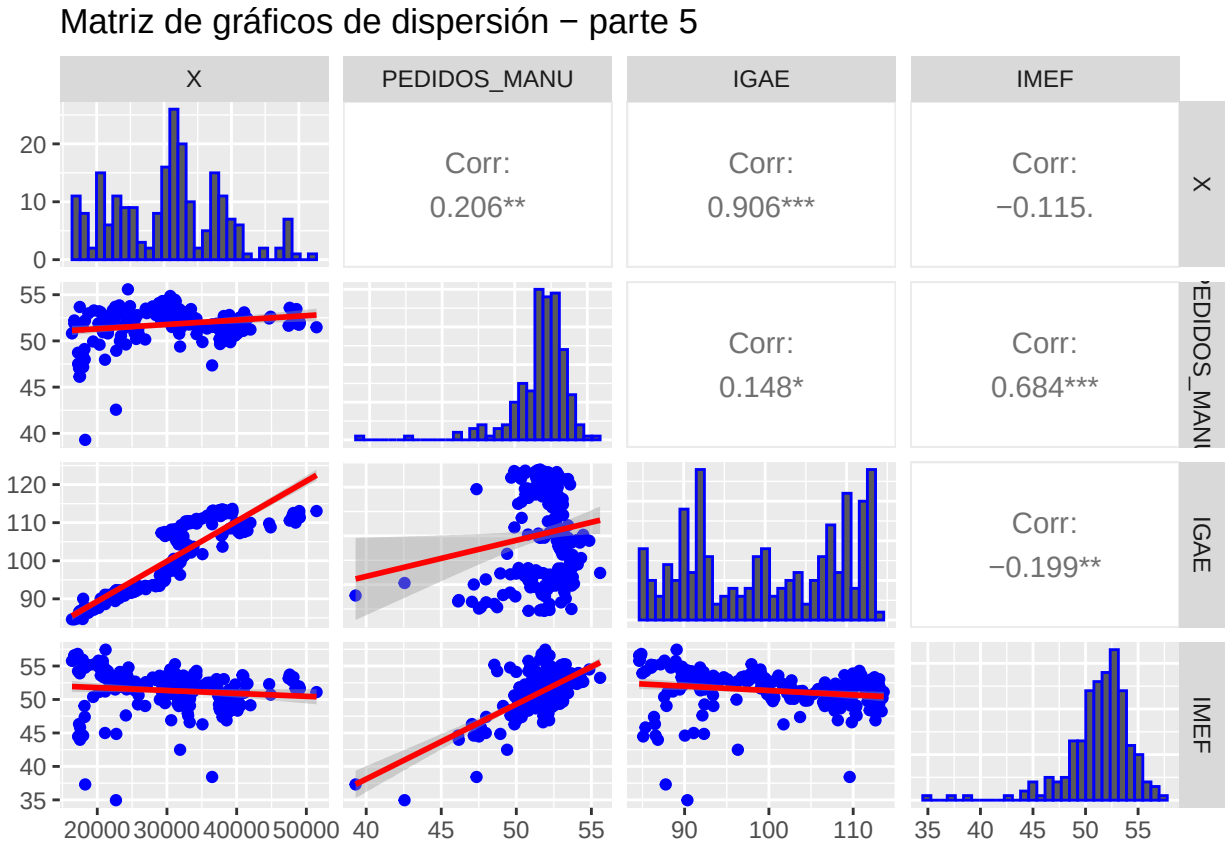
Para la parte 3, en la dispersión de los datos donde interviene la producción industrial de Estados Unidos (IPI_EUA) y la producción vehicular (PROD_VEH), se nota una ligera tendencia lineal; mientras que en la ocupación hotelera (OCUP_HOT) contra las ventas de la ANTAD y la producción vehicular se nota un patrón más aleatorio. Las observaciones de IPI_EUA y OCUP_HOT presentan un sesgo izquierdo importante, en contraste con la ANTAD cuyo sesgo es hacia la derecha. En este sentido, las correlaciones lineales fuertes ocurren en presencia de las variables IPI_EUA y PROD_VEH.

```
# Matriz de gráficos de dispersión parte 4
ggpairs(datos[, 14:17], lower = list(continuous = wrap(lowerFn, method = "lm")),
        diag = list(continuous = wrap("barDiag", colour = "blue")),
        upper = list(continuous = wrap("cor", size = 4)),
        title = "Matriz de gráficos de dispersión - parte 4")
```



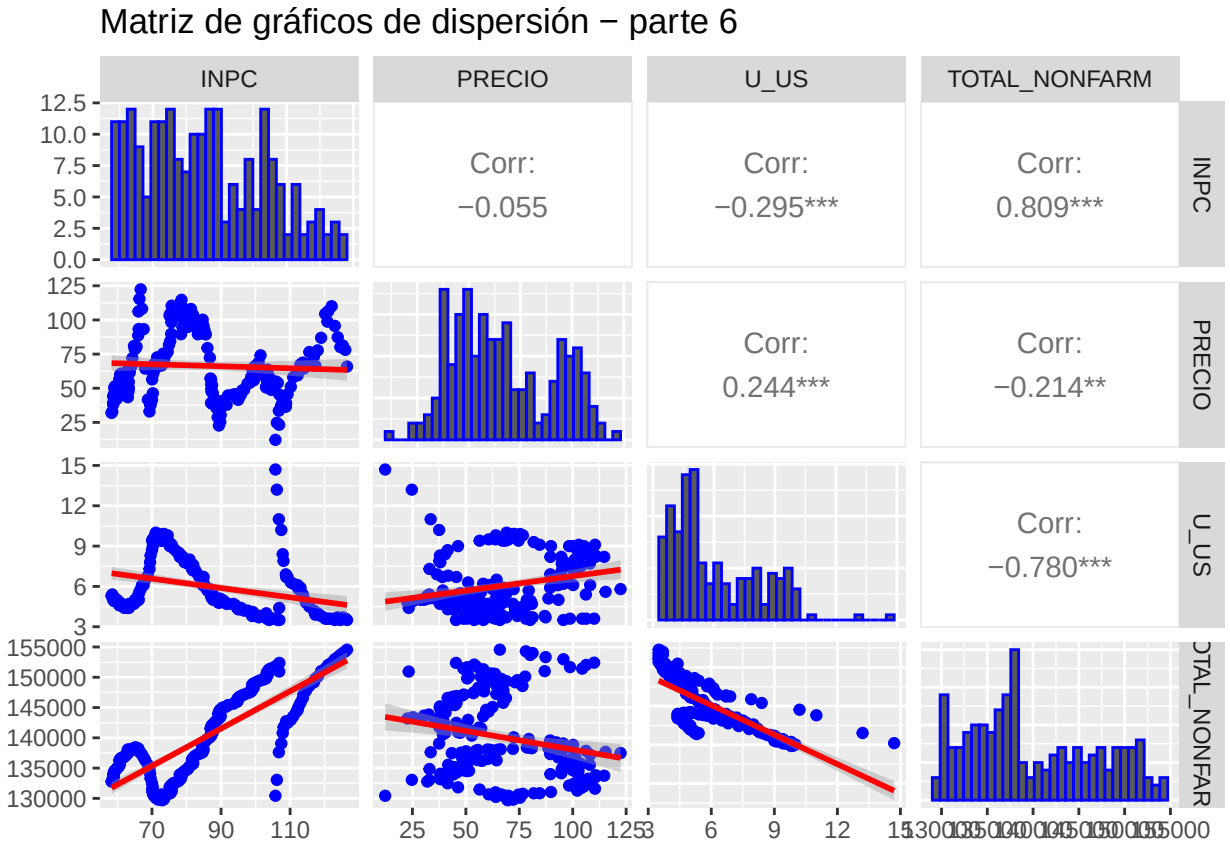
En la matriz anterior, se aprecia una correlación muy fuerte entre los asegurados del IMSS y el agregado monetario M4, con un coeficiente de 0.984; por consiguiente, hay una evidente tendencia lineal entre estas variables y la recta de regresión se ajusta bastante bien. Los datos con la variable GASOLINAS presentan una forma de nube aleatoria y se aprecia un sesgo importante hacia la izquierda. Por otra parte, las remesas presentan un sesgo hacia la derecha; esta variable cuenta con una correlación de 0.754 con la variable IMSS.

```
# Matriz de gráficos de dispersión parte 5
ggpairs(datos[, 18:21], lower = list(continuous = wrap(lowerFn, method = "lm")),
        diag = list(continuous = wrap("barDiag", colour = "blue")),
        upper = list(continuous = wrap("cor", size = 4)),
        title = "Matriz de gráficos de dispersión - parte 5")
```



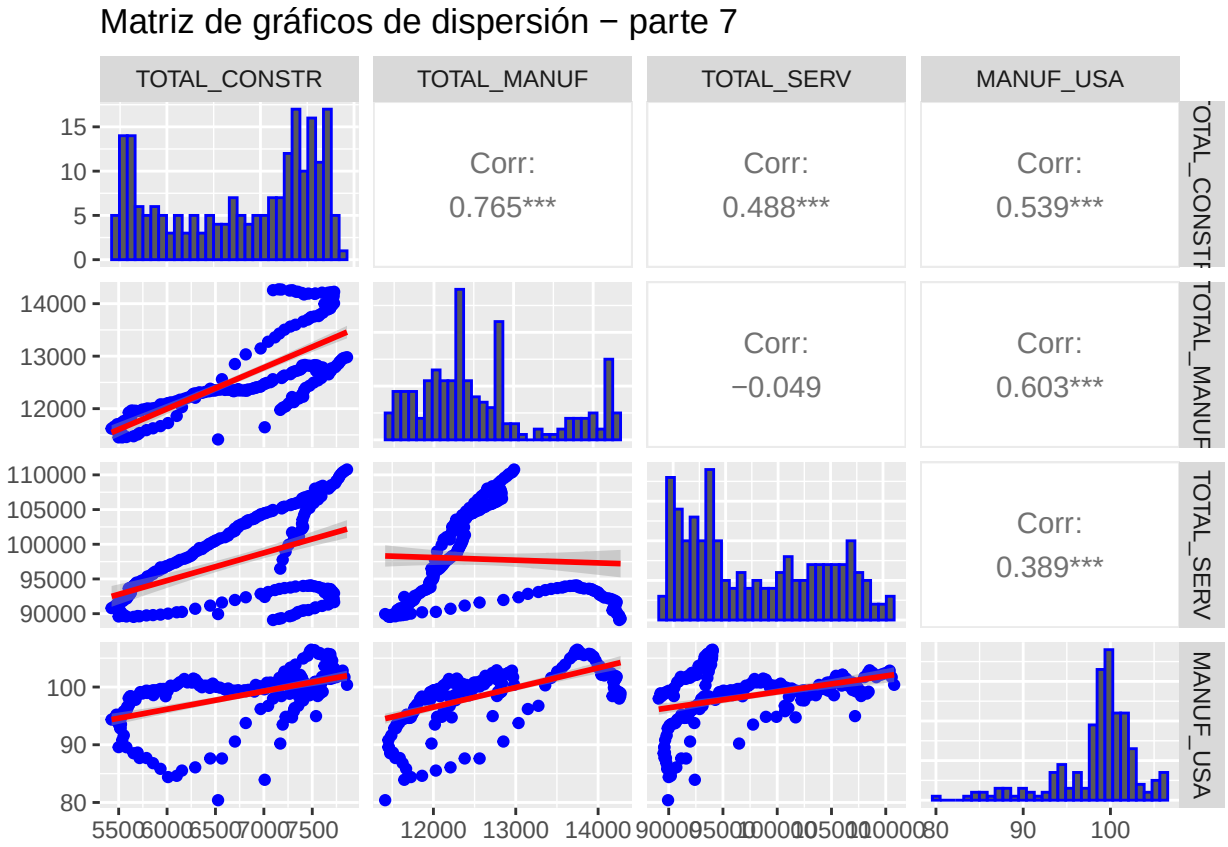
Derivado de lo anterior, se aprecia una evidente relación lineal entre las exportaciones totales (X) y el IGAE, cuya correlación es de 0.906; también entre las variables IMEF y PEDIDOS_MANU se nota una tendencia lineal; entre los demás datos, se nota un patrón más aleatorio. La variable X no presenta un sesgo tan pronunciado, contrario a PEDIDOS_MANU e IMEF, cuyo sesgo es izquierdo.

```
# Matriz de gráficos de dispersión parte 6
ggpairs(datos[, 22:25], lower = list(continuous = wrap(lowerFn, method = "lm")),
        diag = list(continuous = wrap("barDiag", colour = "blue")),
        upper = list(continuous = wrap("cor", size = 4)),
        title = "Matriz de gráficos de dispersión - parte 6")
```



De manera general, algunos de los patrones en las dispersiones de la gráfica anterior no son tan habituales; no obstante, presentan una relación lineal, como es el caso del INPC contra TOTAL_NONFARM, cuya correlación es de 0.809; o también entre las variables U_US y TOTAL_NONFARM, que su correlación es de -0.780. En las demás variables se nota un patrón más aleatorio. Particularmente, en el histograma de la variable INPC, se aprecia que entre más se avance en el eje horizontal, la frecuencia del INPC va disminuyendo.

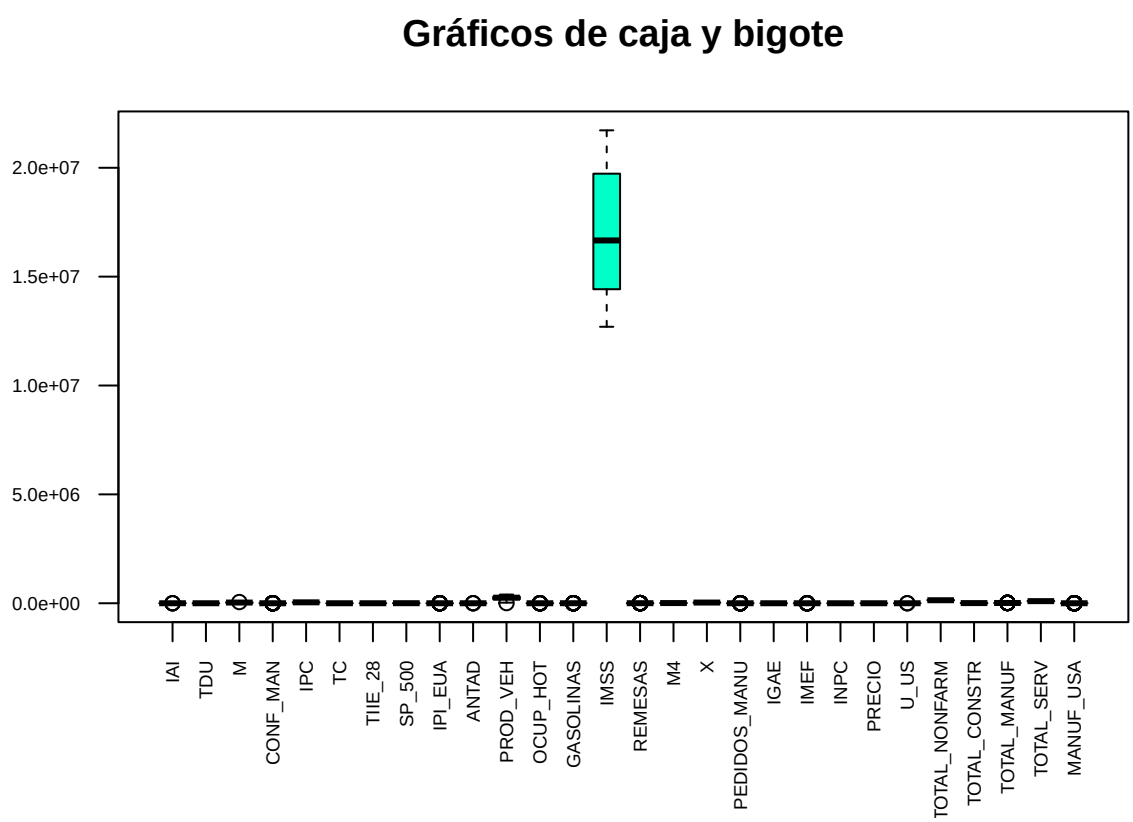
```
# Matriz de gráficos de dispersión parte 7
ggpairs(datos[, 26:29], lower = list(continuous = wrap(lowerFn, method = "lm")),
        diag = list(continuous = wrap("barDiag", colour = "blue")),
        upper = list(continuous = wrap("cor", size = 4)),
        title = "Matriz de gráficos de dispersión - parte 7")
```



En esta última gráfica, se puede apreciar que los datos donde la variable TOTAL_CONSTR interviene, aparece cierta tendencia lineal y por consiguiente, los coeficientes de correlación (positivos todos) se consideran como fuertes. La dispersión entre TOTAL_MANUF y TOTAL_SERV forma un patrón con una trayectoria sin forma y su correlación es de -0.049. En particular, el histograma de MANUF_USA está sesgado hacia la izquierda y su correlación con las demás variables, también es importante.

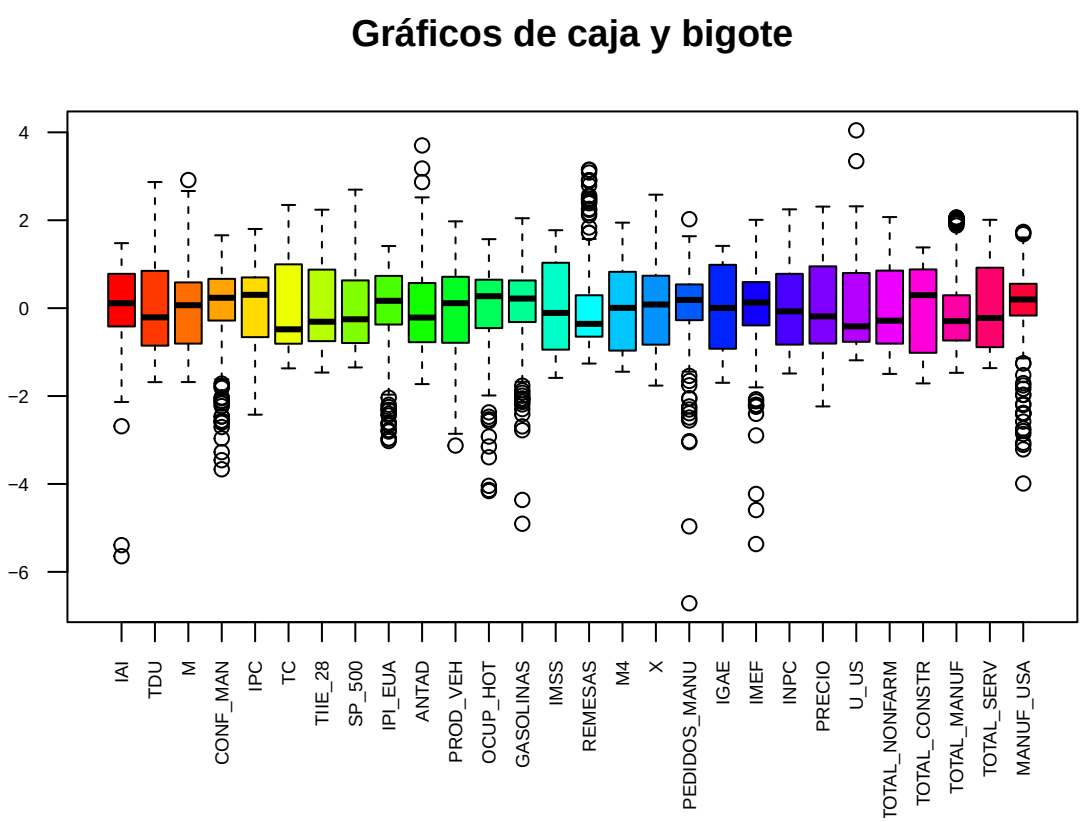
4.2 Gráficos de caja y bigote

```
# Gráfico de caja y bigote
boxplot(datos[, -1], las = 2, cex.axis = 0.6, col = rainbow(28),
        main = "Gráficos de caja y bigote")
```



De acuerdo con el diagrama de caja y bigote, se nota que la variable IMSS es muy diferente del resto respecto a su escala, por lo que estandarizar los datos parece conveniente de realizar, por lo menos para que todas las variables sean comparables de manera visual.

```
# Se estandarizan los datos
datos_std<-scale(datos[, -1], center = TRUE, scale = TRUE)
# Gráfico de caja y bigote
boxplot(datos_std, las=2, cex.axis = 0.6, col = rainbow(28),
        main = "Gráficos de caja y bigote")
```



Ya con los datos estandarizados, se aprecia mejor la presencia de datos atípicos, que cabe resaltar, son varias variables las que tienen importantes cantidades de éstos, como lo son CONF_MAN, IPI_EUA, OCUP_HOT, GASOLINAS, REMESAS, PEDIDOS_MANU, IMEF y MANUF_USA.

5. Modelo SARIMA

Como se mencionó con anterioridad, el Índice Nacional de Precios al Consumidor (INPC) es un indicador cuya finalidad es estimar la evolución de los precios de ciertos bienes y servicios que consumen las familias en México. El INPC mide la variación de los precios de una canasta básica de bienes y servicios representativa del consumo regular en los hogares. La serie de tiempo del INPC tiene regularmente tendencia al alza; sin embargo, también experimenta tendencias a la baja y períodos en los que su variación es mínima.

El INEGI tiene la facultad exclusiva de elaborar y publicar los índices nacionales de precios a partir del 15 de julio de 2011, según lo publicado en el sitio de este Instituto http://www.inegi.org.mx.

Análisis de la serie temporal del INPC

Para el modelo SARIMA que se elaborará para generar pronósticos del INPC, se tomarán los datos de las series del IOAE, de la que se cuenta con datos mensuales de enero del 2005 a diciembre del 2022.

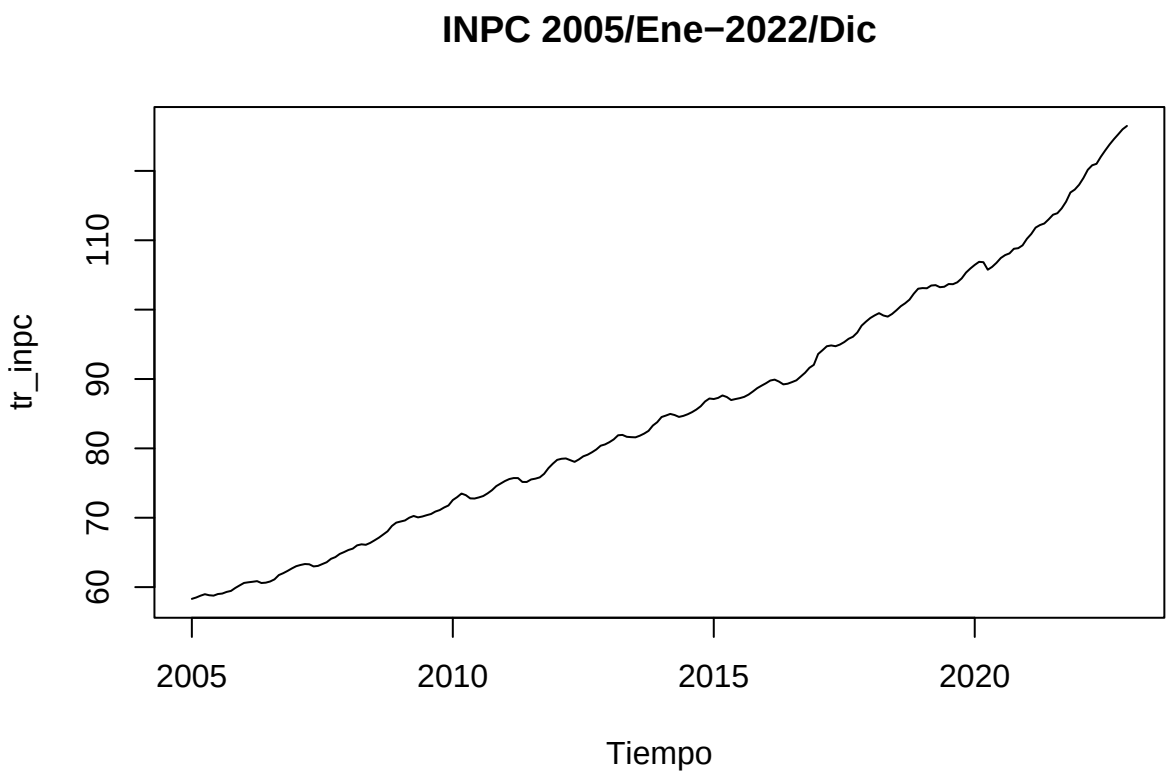
En caso de que la serie muestre problemas de heterocedasticidad, se analizarán los datos mediante una transformación, que regularmente es la logarítmica, con la intención de observar si este problema mejora. Posteriormente, se realizará un análisis exploratorio breve, se analizarán las gráficas ACF y PACF y se eliminará la tendencia y estacionalidad, en caso de que exista mediante las diferenciaciones que correspondan. Durante el modelado, se tratará de aplicar un número óptimo de veces el operador de diferenciación, de manera que no represente una sobre diferenciación. Después, se propondrá un modelo SARIMA, se verificará normalidad y estacionariedad de los residuos y se procederá a la generación de los pronósticos.

5.1 Análisis exploratorio de la serie del INPC

La serie de tiempo consta de 216 registros de observaciones mensuales sin datos omitidos desde enero del 2005 a diciembre del 2022. Los datos de la serie están desestacionalizados según la fuente que publica; visualmente se aprecia con una clara tendencia determinística y parecerían observarse ciclos repetitivos con períodos constantes de incremento y ligeros decrementos durante casi todo el período de observación. Gráficamente se comporta de la siguiente manera.

```
####################################
# Modelo SARIMA
####################################

# Limpiar espacio de trabajo
rm(list = ls())
invisible(gc(reset = TRUE))

# Librerías
library(astsa)      # Gráfico de retrasos
library(tseries)    # Prueba de ADF
library(forecast)   # Pronósticos
library(seasonal)   # Estacionalidad
library(kableExtra) # Edición de tablas

# Establecer dirección de trabajo
ruta<-("C:/Proyecto de series/")
# Se cargan las funciones
source(paste(ruta, "functions.r", sep = ""))

# Se leen los datos
datos<-read.csv(file = paste0(ruta, 'IOAE.csv'))
```

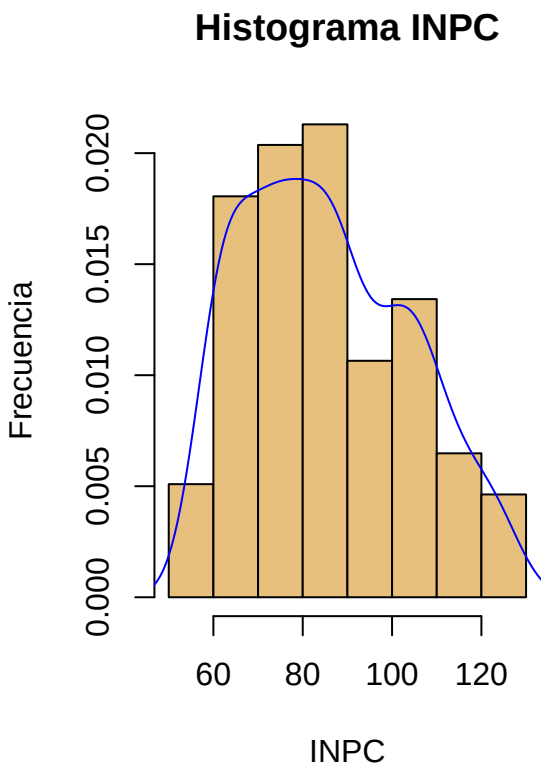
```r
# Se convierten a series de tiempo
tr_inpc<-ts(datos[, "INPC"], start = c(2005, 1), frequency = 12)
# Número de series
N<-ncol(tr_inpc)
# Gráfico temporal
plot(tr_inpc, main = "INPC 2005/Ene-2022/Dic", xlab = "Tiempo", cex.lab = 0.75)
```

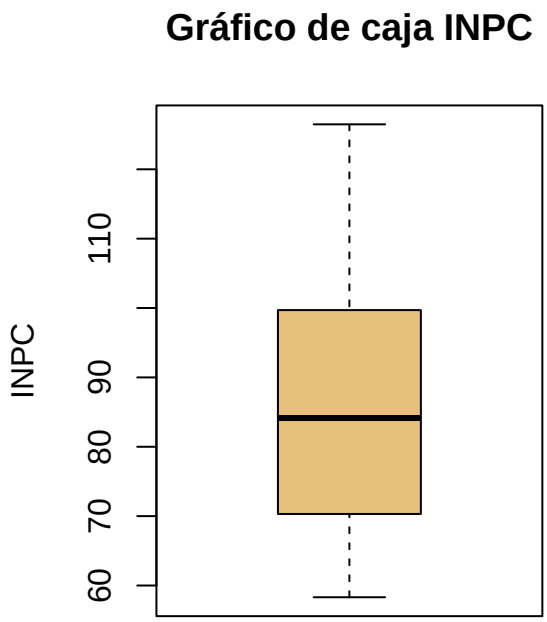


Como primer acercamiento a los datos, se observa en la gráfica que durante el período de enero del 2005 a diciembre del 2022 existe una clara tendencia al alza, con pequeñas oscilaciones estacionales de incremento y decremento (varianza constante) durante casi todo el período de análisis del indicador.

Se procederá con la elaboración de un histograma y un gráfico de cajas que permitan hacer una exploración y conocer de forma general la distribución de los datos.

```r
# Análisis exploratorio
par(mfrow = c(1, 2))
# Creación del histograma
hist(tr_inpc, main = "Histograma INPC", col = "#E8C07D", ylab = "Frecuencia",
     xlab = "INPC", prob = TRUE)
# Densidad del histograma
lines(density(tr_inpc), col = "blue")

# Gráfico de caja y bigote
boxplot(tr_inpc, main = "Gráfico de caja INPC", col = "#E8C07D", ylab = "INPC")
```





Se puede observar que el histograma tiene un comportamiento parecido a una distribución gaussiana, pero con un pequeño sesgo a la derecha, en la que el promedio ronda por los 85 puntos porcentuales; se observa que se acumulan más valores por encima del valor de la mediana; marcando un pequeño sesgo a la derecha de este valor de tendencia central.

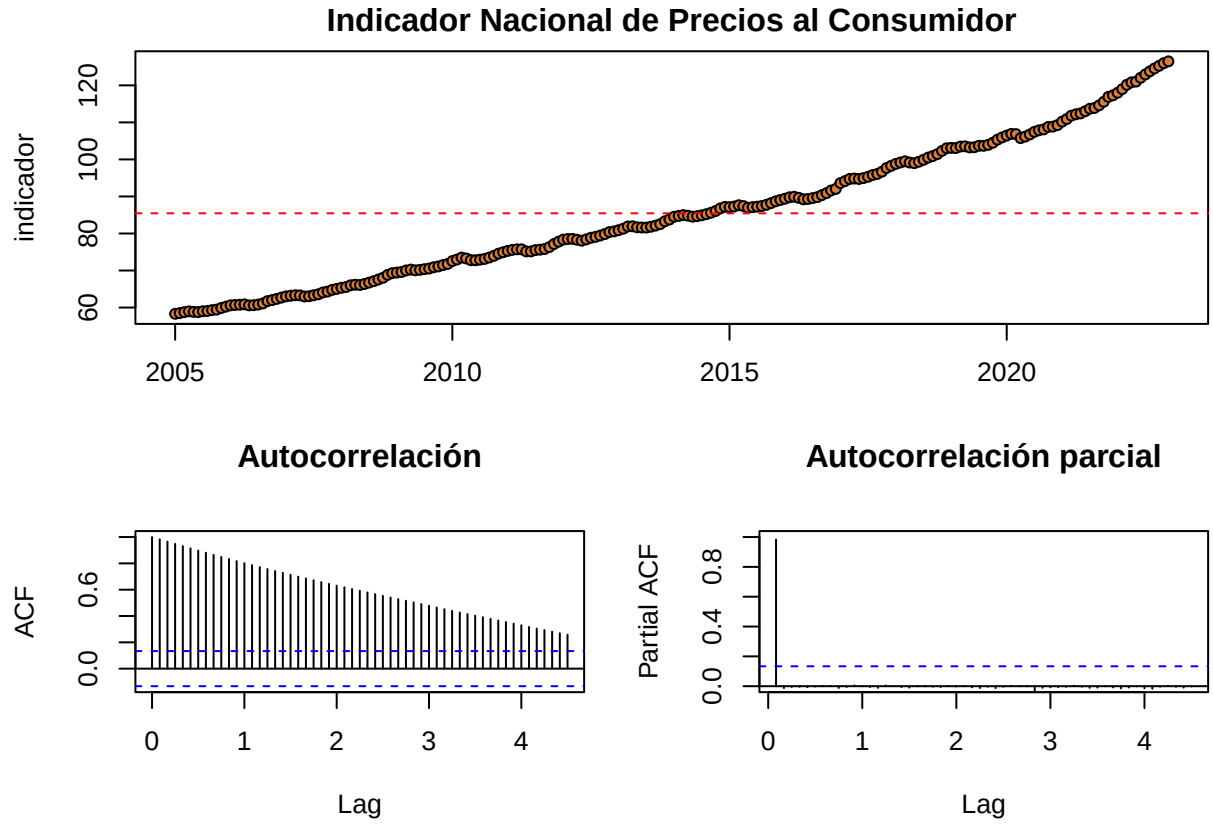
Respecto a la identificación de datos atípicos, se tiene la prueba de Tukey, que identifica valores fuera del rango intercuartílico IQR. En este caso, en el diagrama de caja no se aprecian valores atípicos; sin embargo, la mayor concentración de los datos sucede por encima de la mediana, mostrando un poco de asimetría en los datos (sin sobresalir el rango de valores de 1.5 veces el IQR según la longitud de las líneas de bigote).

Gráficas de Autocorrelación

Enseguida, se presentan los respectivos gráficos de autocorrelación (ACF) y autocorrelación parcial (PACF), mismos que permitirán analizar los coeficientes de autocorrelación entre los períodos de la serie de tiempo del INPC.

```
# Alineación de las gráficas ACF y PACF.
layout(mat = matrix(c(1, 1, 2, 3), nrow = 2, ncol = 2, byrow = T))
# Márgenes del primer gráfico
par(mar = c(3, 4.5, 2, 1))
# Serie temporal
plot(tr_inpc, main = "Indicador Nacional de Precios al Consumidor",
     ylab = "indicador", xlab = "Año", type = "o", pch = 21, bg = "#D67D3E")
abline(h = mean(tr_inpc), col = "red", lty = 2)
# Márgenes del segundo gráfico y tercer gráfico
par(mar = c(5, 4.5, 3.5, 1))
# Gráfico de autocorrelación
```

```r
acf(tr_inpc, main = "Autocorrelación", lag = 54)
# Gráfico de autocorrelación parcial
pacf(tr_inpc, main = "Autocorrelación parcial", ylim=c(0,1), lag = 54)
```

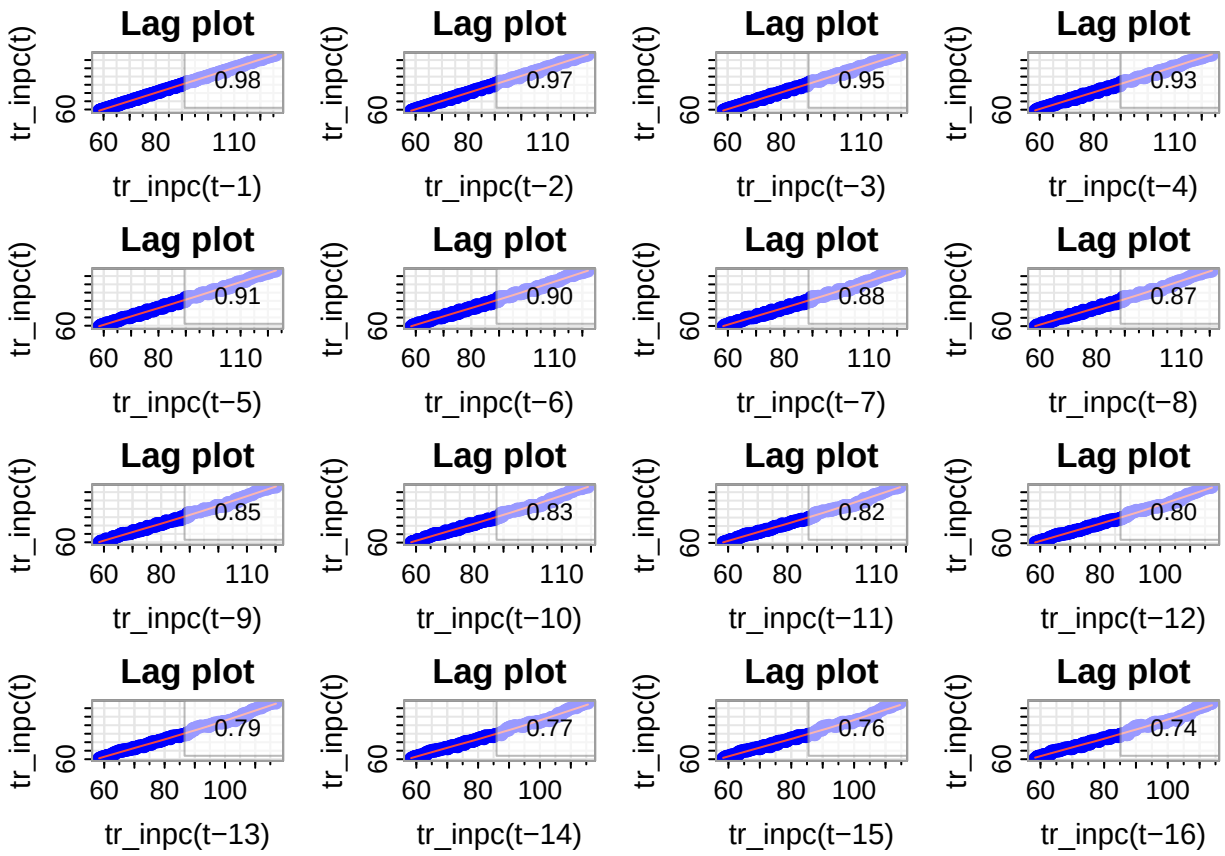


En el primer gráfico, correspondiente a la serie temporal, se observa claramente un proceso no estacionario, es decir, una tendencia al alza del INPC a lo largo de todo el período, por lo que el análisis se centrará en primer lugar en eliminar el efecto de la tendencia y hacer de la serie un proceso estacionario. Es conveniente también mencionar que la evolución de la tendencia presenta variaciones homogéneas, esto significa que no hay cambios explosivos de la varianza a lo largo de la serie.

Con relación al correlograma de autocorrelación ACF, se puede intuir que la característica dominante es la tendencia; eso se observa por las barras que van descendiendo lentamente; esto indicaría la presencia de una raíz unitaria, por lo que antes de ajustar un modelo, la tendencia debería removerse. En el gráfico de autocorrelación parcial, la primera barra cuenta con un valor de 0.92, lo que da indicios de que será necesario diferenciar para eliminar esa tendencia.

Adicionalmente, se presenta el gráfico de rezagos (*lag.plot*), que claramente muestra que la serie tiene una dependencia o correlación de su valor actual $tr_{inpc}(t)$ con su rezago anterior $tr_{inpc}(t-1)$. El patrón de tendencia lineal positiva que es predominante en cada gráfica, indica la correlación entre los valores de la serie y sus rezagos. Cuanto más juntos los puntos se sitúen sobre la diagonal mayor es la autocorrelación. Los datos al estar perfectamente autocorrelacionados, se agruparán en una sola línea diagonal que es justo como se aprecian estas gráficas.

```r
#Gráfica de rezagos
lag1.plot(tr_inpc, max.lag = 16, main = "Lag plot", pch = 19, col = "blue")
```

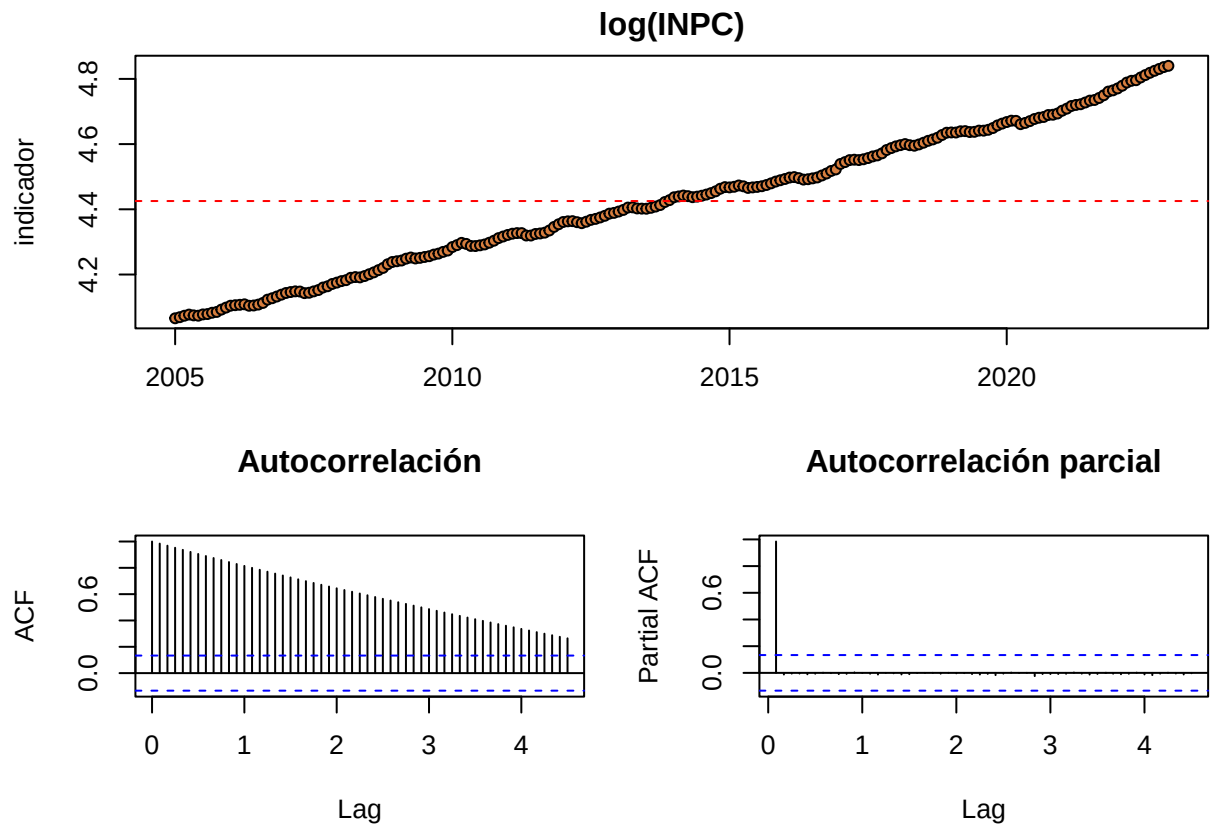


En el gráfico se observa que en el primer rezago (primer recuadro superior izquierdo) la correlación es de 0.98, lo que indica que es necesario diferenciar para eliminar esta tendencia. Adicionalmente, se observa que la varianza no tiene aumentos o decrementos abruptos en el tiempo.

Transformación logarítmica

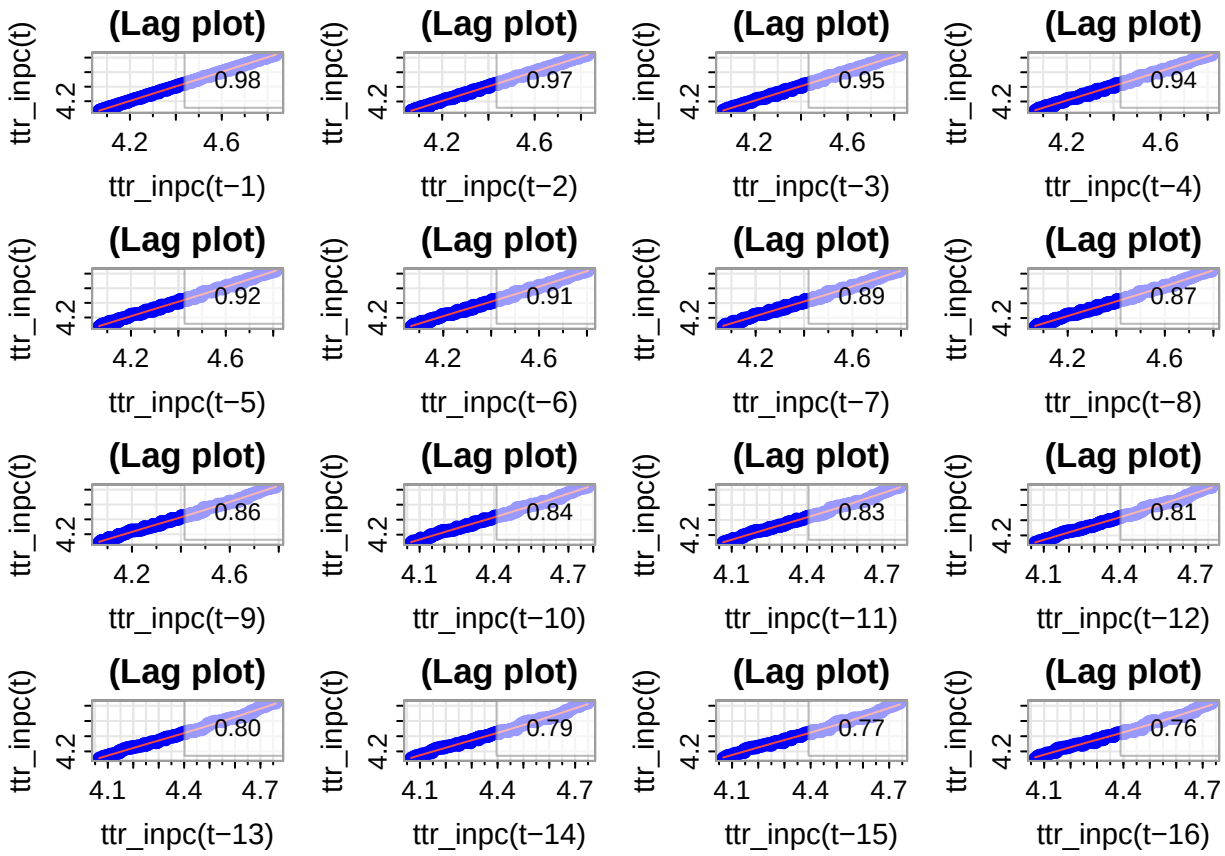
Dado lo anterior, se realizará una transformación logarítmica a los datos antes de diferenciar, con la intención de observar si hay mejora en su varianza. A continuación, se desarrolla la transformación y se muestran las gráficas con los datos transformados.

```
# Análisis inicial de los datos transformados
# Datos transformados. Aplicación de la función logaritmica
ttr_inpc<-log(tr_inpc)
# Matriz de gráficos
layout(mat = matrix(c(1, 1, 2, 3), nrow = 2, ncol = 2, byrow = T))
# Márgenes del primer gráfico
par(mar = c(3, 4.5, 2, 1))
# Serie temporal
plot(ttr_inpc, main = "log(INPC)",
     ylab = "indicador", xlab = "Año", type = "o", pch = 21, bg = "#D67D3E")
abline(h = mean(ttr_inpc), col = "red", lty = 2)
# Márgenes del segundo gráfico y tercer gráfico
par(mar = c(5, 4.5, 3.5, 1))
# Gráfico de autocorrelación
acf(ttr_inpc, main = "Autocorrelación", lag = 54)
# Gráfico de autocorrelación parcial
pacf(ttr_inpc, main = "Autocorrelación parcial", lag = 54)
```



Como se puede observar, al aplicar la transformación logarítmica no se percibe un cambio significativo sobre el comportamiento de los datos. Además, se realizó el gráfico de rezagos con la intención de observar si la transformación logarítmica mejora la varianza, pero no se aprecia ningún cambio significativo.

```
#Gráfica de rezagos
lag1.plot(ttr_inpc, max.lag = 16, main = "(Lag plot)", pch = 19, col = "blue")
```



En la gráfica previa se puede observar que la transformación logarítmica no cambia de forma importante la varianza de los datos, por lo tanto, se decidió seguir el análisis con la serie original.

Prueba Dicky Fuller de raíz unitaria

A continuación, se procede a aplicar la prueba aumentada de Dickey Fuller, empleando las funciones implementadas por el Dr. Francisco de Jesús Corona Villavicencio, para evaluar la posible existencia de una raíz unitaria ante la tendencia observada en la gráfica. La prueba de hipótesis es la siguiente:

H_o : La serie de tiempo es no estacionaria (es necesario diferenciar por tendencia)
H_a : La serie de tiempo es estacionaria (no es necesario diferenciar por tendencia)

La regla de decisión es rechazar la hipótesis nula (H_o) si el valor p es menor al nivel de significancia de 0.05 ($\alpha = 0.05$). La salida de la prueba se muestra a continuación.

```
# Prueba aumentada de Dickey-Fuller
adf(tr_inpc, type = "none", alternative = "stationary")
```

```
#
#   Augmented Dickey-Fuller Test
#
# data:  tr_inpc
# Dickey-Fuller = 6.5889, Lag order = 5, p-value = 0.99
# alternative hypothesis: stationary
```

De acuerdo con la salida anterior, el valor p fue de $0.99 > 0.05$, por lo tanto, no se rechaza la H_o, es decir, la serie es no estacionaria, por lo tanto, es necesario remover la tendencia mediante una diferenciación antes de ajustar el modelo.

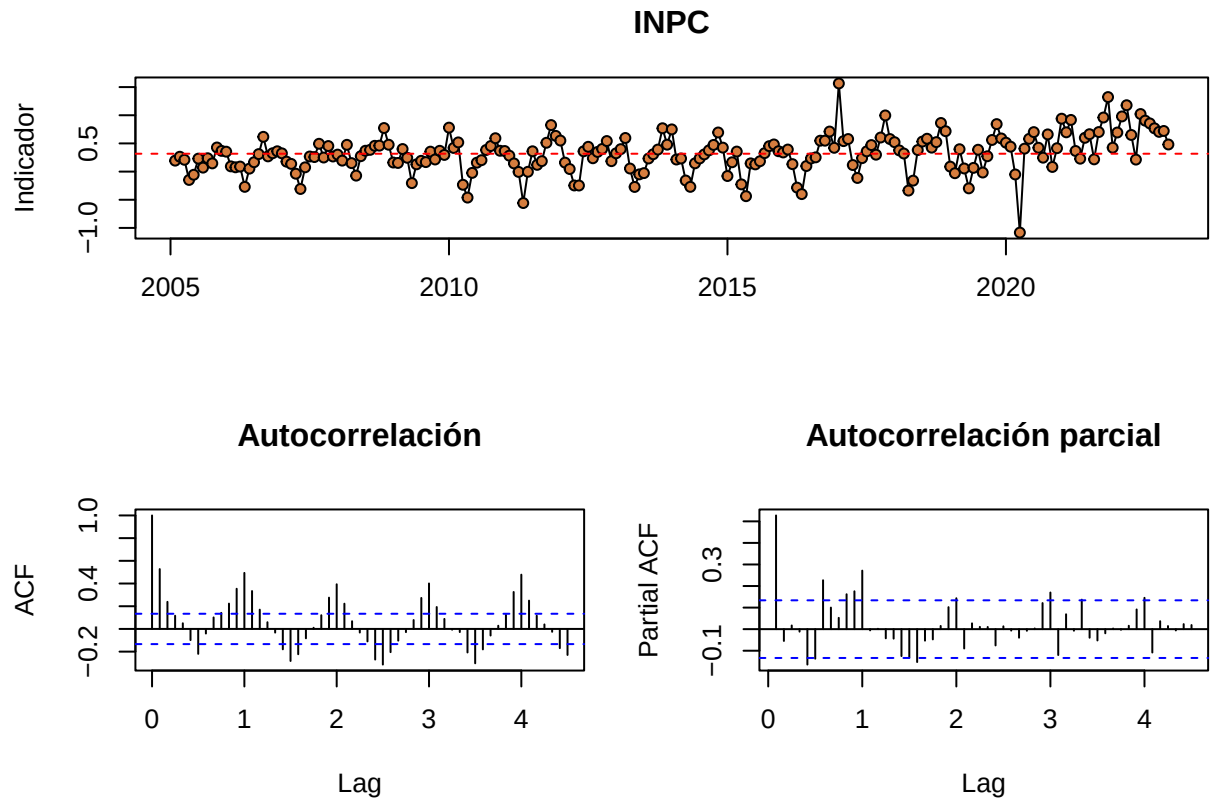
5.1 Diferenciación para remover tendencia en la serie temporal

Se procede a realizar una primera diferenciación $X_t - X_{t-1}$ a la serie temporal, misma que presenta tendencia y que al diferenciarse puede resultar en una serie estacionaria.

```
# Eliminar tendencia con diferenciación
dtr_inpc<-diff(tr_inpc)
```

Enseguida, se genera el gráfico de autocorrelación (ACF) y autocorrelación parcial (PACF) para los datos diferenciados.

```
# Matriz de gráficos
layout(mat = matrix(c(1, 1, 2, 3), nrow = 2, ncol = 2, byrow = T))
# Márgenes del segundo y tercer gráfico
par(mar = c(5, 4.5, 3.5, 1))
# Serie temporal
plot(dtr_inpc, main = "INPC",
     ylab = "Indicador", xlab = "", type = "o", pch = 21, bg = "#D67D3E")
abline(h = mean(dtr_inpc), col = "red", lty = 2)
# Gráfico de autocorrelación
acf(dtr_inpc, main = "Autocorrelación", lag = 54)
# Gráfico de autocorrelación parcial
pacf(dtr_inpc, main = "Autocorrelación parcial", lag = 54)
```

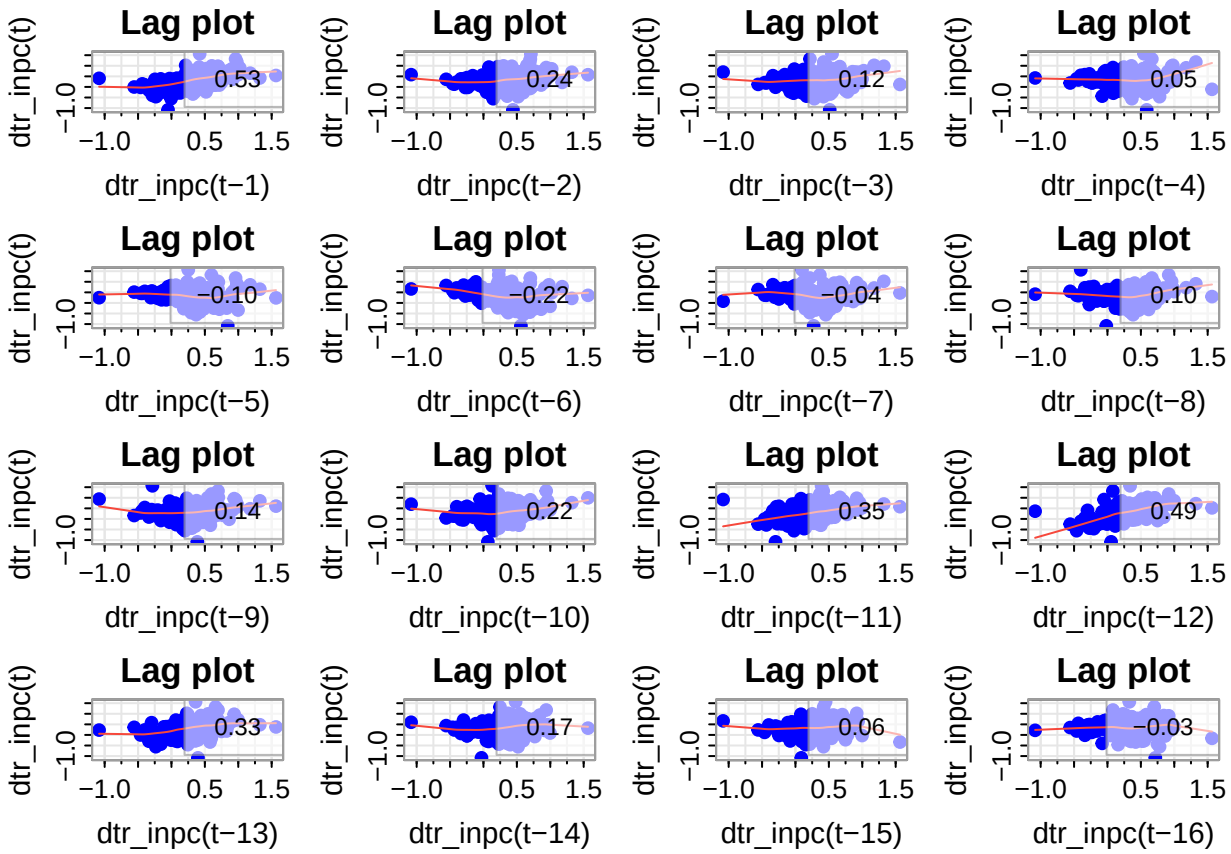


En la gráfica de la serie temporal, se observa que con la diferenciación realizada se logra una estabilización en la serie, misma que elimina la tendencia de los datos originales. Es evidente que se descubren algunos

datos atípicos, uno positivo durante enero del 2007, un valor atípico negativo en abril del 2020 (seguramente por motivo de la pandemia por COVID-19) y uno más, positivo, en noviembre del 2021, entre algunos otros. Sin embargo, para efectos de capturar toda la variabilidad posible en el modelo, se decide no eliminar dichos atípicos y continuar con el conjunto de datos completo. Asimismo, se puede apreciar un patrón que se repite cada 12 datos, lo que da indicios de estacionalidad.

Con respecto a los correlogramas con datos diferenciados, se observan los coeficientes de autocorrelación (ACF), diferentes de cero y sobrepasando el nivel de significancia; se observa que los coeficientes de la ACF no recaen rápidamente y que además presentan un comportamiento sinusoidal que se repite cada que se cumple un ciclo (12 meses), lo que da indicios de estacionalidad. Como propuesta de validación, se realizó el gráfico de rezagos de los datos diferenciados.

```
# Gráfica de rezagos
lag1.plot(dtr_inpc, max.lag = 16, main = "Lag plot", pch = 19, col = "blue")
```



En el diagrama se observa, por ejemplo, el gráfico para el rezago número 12, en el que el nivel de correlación (0.49) indica cierto grado de estacionalidad anual. Debido a esto puede ser necesario aplicar una diferenciación cada 12 meses para atenuar la estacionalidad.

5.2 Prueba de Estacionalidad

Para la realización de esta prueba, se empleará un método que consiste en incluir variables ficticias estacionales o dummy y comprobar si tienen valores p significativos al calcular la regresión. Se recuerda que estas variables dummy se pueden utilizar como variables explicativas para modelar el efecto de la estacionalidad sobre una variable dependiente. Si los coeficientes de las variables para los meses individuales

resultan significativos, la serie de tiempo presenta problemas de estacionalidad. El resultado se presenta a continuación.

```
# Modelo para determinar estacionalidad
estacionalidad<-lm(dtr_inpc ~ c(1:length(dtr_inpc)) + seasonaldummy(dtr_inpc))
# Resumen del modelo
summary(estacionalidad)
```

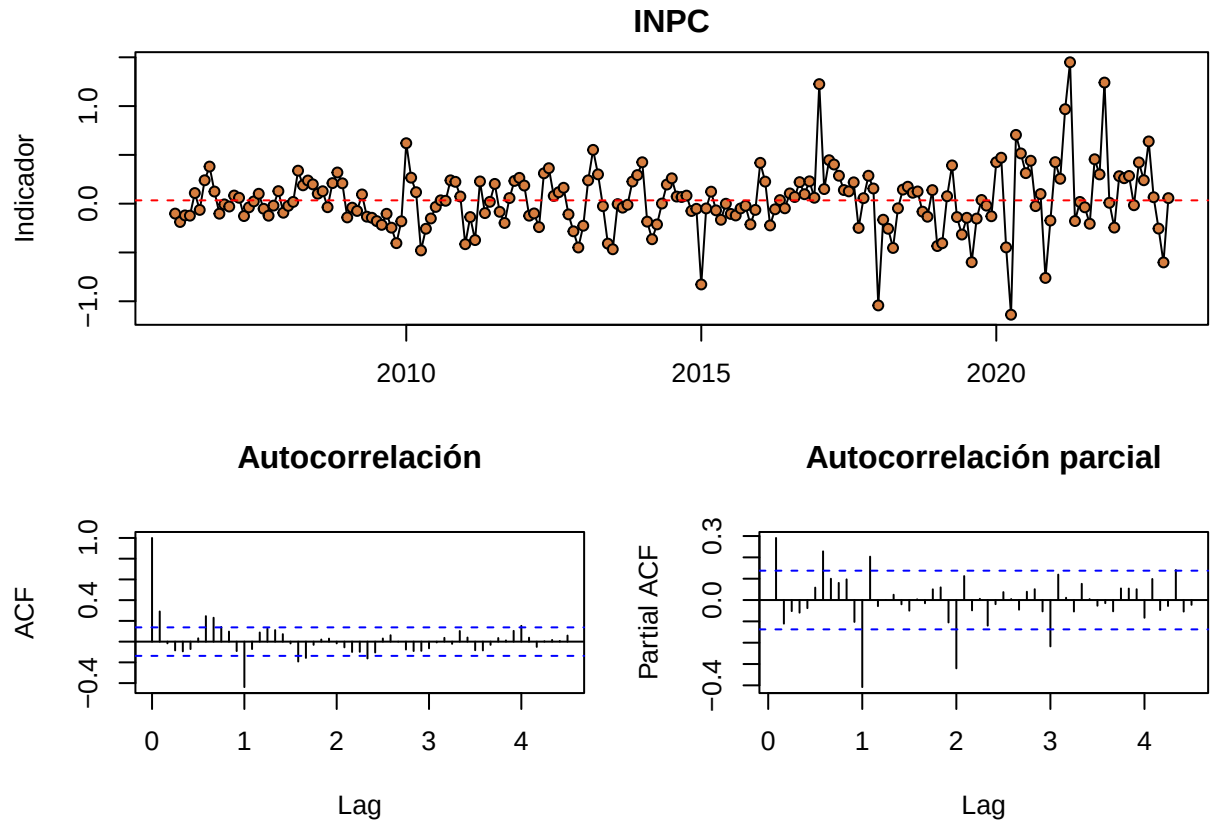
```
#
# Call:
# lm(formula = dtr_inpc ~ c(1:length(dtr_inpc)) + seasonaldummy(dtr_inpc))
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -1.17920 -0.12106 -0.01735  0.12064  1.00359
#
# Coefficients:
#                              Estimate Std. Error t value Pr(>|t|)
# (Intercept)                 0.2418280  0.0635343   3.806 0.000187 ***
# c(1:length(dtr_inpc))       0.0017174  0.0002623   6.547 4.74e-10 ***
# seasonaldummy(dtr_inpc)Jan  0.0721188  0.0806449   0.894 0.372239
# seasonaldummy(dtr_inpc)Feb -0.0938705  0.0795173  -1.181 0.239188
# seasonaldummy(dtr_inpc)Mar -0.0473327  0.0795091  -0.595 0.552301
# seasonaldummy(dtr_inpc)Apr -0.4599130  0.0795017  -5.785 2.73e-08 ***
# seasonaldummy(dtr_inpc)May -0.6113276  0.0794952  -7.690 6.32e-13 ***
# seasonaldummy(dtr_inpc)Jun -0.2013187  0.0794896  -2.533 0.012081 *
# seasonaldummy(dtr_inpc)Jul -0.0773531  0.0794848  -0.973 0.331627
# seasonaldummy(dtr_inpc)Aug -0.1369841  0.0794809  -1.723 0.086331 .
# seasonaldummy(dtr_inpc)Sep -0.0223928  0.0794779  -0.282 0.778425
# seasonaldummy(dtr_inpc)Oct  0.0444908  0.0794757   0.560 0.576233
# seasonaldummy(dtr_inpc)Nov  0.2210778  0.0794744   2.782 0.005919 **
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 0.2384 on 202 degrees of freedom
# Multiple R-squared:  0.5313,  Adjusted R-squared:  0.5035
# F-statistic: 19.08 on 12 and 202 DF,  p-value: < 2.2e-16
```

Con base a la salida previa, se observa que el coeficiente de determinación ajustado (R^2_{adj}) es de 0.5035, es decir, el 50% de la variabilidad de los datos es explicada por el modelo que se ha ajustado. Asimismo, se puede observar que los coeficientes de las variables ficticias estacionales para los meses de abril y mayo presentan una alta significancia, lo que determina un patrón que se repite anualmente (como se observó en la gráfica ACF) en estos meses en particular. De igual manera, se observa significancia para los meses de junio y noviembre, así como para el coeficiente del intercepto, por lo que se determina estacionalidad en la serie de tiempo y se concluye que será necesario aplicar una diferenciación cada 12 meses para remover este problema de estacionalidad.

Diferenciación para eliminar estacionalidad

Enseguida, se diferenciará cada 12 datos la serie de tiempo para atenuar el patrón estacional que se observó en la prueba anterior.

```r
# Datos ddtr_inpc diferenciación cada 12 meses
ddtr_inpc<-diff(dtr_inpc, 12)
# Matriz de gráficos
layout(mat = matrix(c(1, 1, 2, 3), nrow = 2, ncol = 2, byrow = T))
# Márgenes del primer gráfico
par(mar = c(3, 4.5, 2, 1))
# Serie temporal
plot(ddtr_inpc, main = "INPC",
     ylab = "Indicador", xlab = "Año", type = "o", pch = 21, bg = "#D67D3E")
abline(h = mean(ddtr_inpc), col = "red", lty = 2)
# Márgenes del segundo y tercer gráfico
par(mar = c(5, 4.5, 3.5, 1))
# Gráfico de autocorrelación
acf(ddtr_inpc, main = "Autocorrelación", lag = 54)
# Gráfico de autocorrelación parcial
pacf(ddtr_inpc, main = "Autocorrelación parcial", lag = 54)
```



En esta gráfica se puede apreciar cómo se atenuó la estacionalidad, ya que en la serie temporal ya no se percibe de forma clara ese patrón. Asimismo, en el gráfico ACF se eliminó ese patrón que se observaba cada ciclo. No obstante, en el gráfico PACF se puede apreciar que las autocorrelaciones más altas se presentan cuando se cumple un ciclo (12 meses). Este comportamiento debe ser considerado en el ajuste del modelo SARIMA.

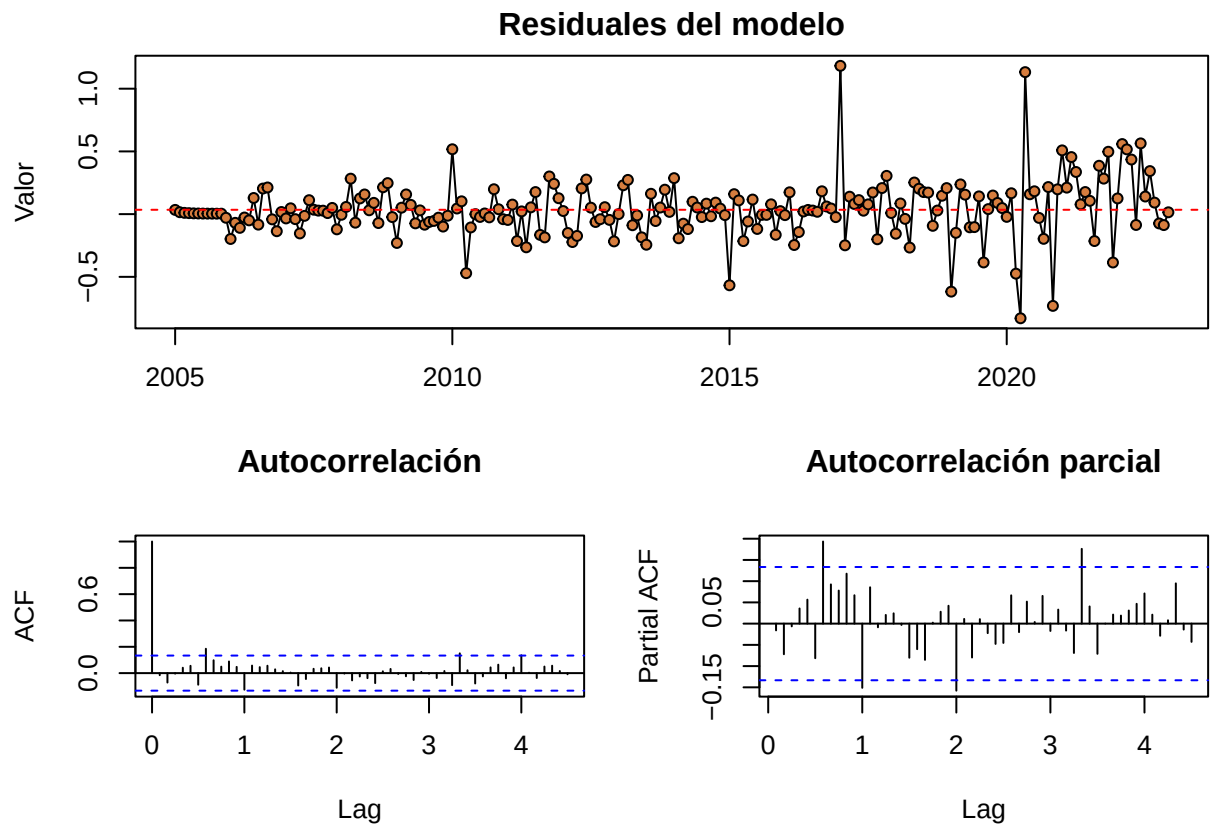
5.3 Ajuste del modelo

Para ajustar el modelo $SARIMA\,(p,d,q)(P,D,Q)_m$, se analiza el comportamiento de las gráficas ACF y PACF, las cuales permiten determinar el número de parámetros del modelo. Posteriormente, se realizarán pruebas de normalidad y ruido blanco a los residuales del modelo; con este procedimiento se ajustará un modelo válido para generar pronósticos.

El modelo inicial que se propone comienza siendo un modelo $SARIMA(0,1,0)(0,1,0)_{m=12}$, debido a la diferenciación por tendencia $d = 1$ y a la diferenciación por estacionalidad $D = 1$. En el análisis de las gráficas, en los componentes AR y MA, se observa que, en la ACF, en la parte no estacional, se distingue un corte dado por la primera barra significativa (después de la barra del coeficiente de autocorrelación situado en 0 para el eje horizontal); asimismo, en la gráfica PACF se aprecia un corte indicado por la primera barra significativa; con lo anterior, se determina un modelo ARMA con parámetros $p = 1$ y $q = 1$, es decir, $ARMA(1,1)$. Se hace notar que las primeras barras tanto de las gráficas ACF y PACF presentan coeficientes posteriores que pueden parecer significativos pero que se identifican muy cercanos al ciclo estacional, y que se espera se atenúen con el ajuste en los componentes SAR y SMA del modelo SARIMA.

Respecto a los componentes SAR y SMA, se observa en el ACF una barra significativa en la parte estacional indicando un corte, mientras que, en la gráfica PACF se aprecia una cola con los coeficientes de los períodos 1, 2 y 3. Teniendo corte y cola se determina el componente SMA con un parámetro $Q = 1$ dado por una barra significativa en la gráfica ACF. Con lo anterior, se sugiere el modelo $SARIMA(1,1,1)(0,1,1)_{m=12}$. Enseguida se realiza esta implementación.

```
#Ajuste del modelo SARIMA.
 modelo_inpc<-arima(tr_inpc, order = c(1, 1, 1),
              seasonal = list(order = c(0, 1, 1), period = 12))
# Análisis de los residuales
residuales_inpc<-modelo_inpc$residuals
# Matriz de gráficos
layout(mat = matrix(c(1, 1, 2, 3), nrow = 2, ncol = 2, byrow = T))
# Márgenes del primer gráfico
par(mar = c(3, 4.5, 2, 1))
# Serie temporal
plot(residuales_inpc, main = "Residuales del modelo", ylab = "Valor",
     xlab = "Año", type = "o", pch = 21, bg = "#D67D3E")
abline(h = mean(residuales_inpc), col = "red", lty = 2)
# Márgenes del segundo y tercer gráfico
par(mar = c(5, 4.5, 3.5, 1))
# Gráfico de autocorrelación
acf(residuales_inpc, main = "Autocorrelación", lag = 54)
# Gráfico de autocorrelación parcial
pacf(residuales_inpc, main = "Autocorrelación parcial", lag = 54)
```

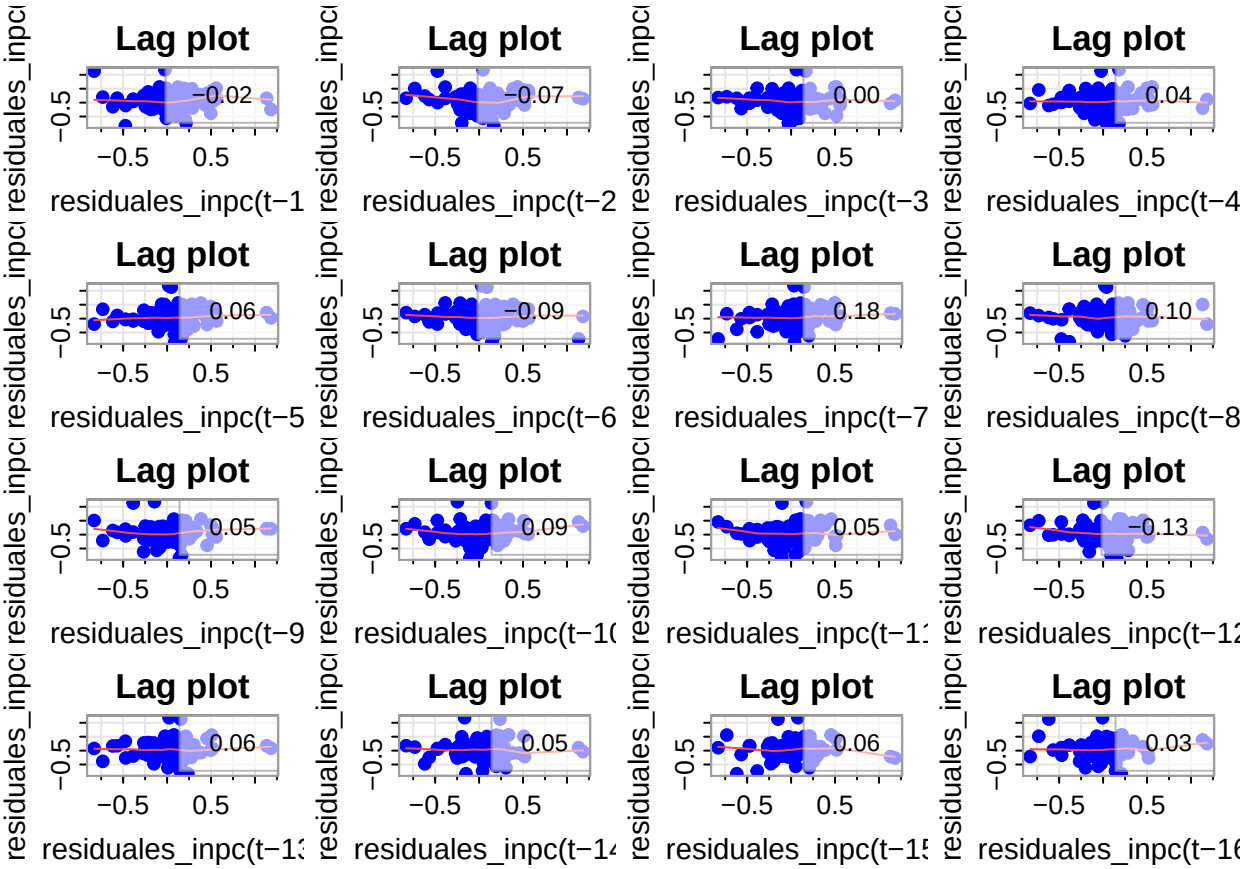


En la gráfica ACF se aprecia que los coeficientes de correlación de los residuales son suficientemente pequeños como para suponer que se comportan como ruido blanco. Por su parte, en la gráfica PACF se aprecian unas barras que también salen de los límites de significancia, pero se hace resaltar que la escala es muy pequeña y que podrían no representar un problema para que el modelo sea considerado como válido. Esta presunción de la cual aún no se tiene certeza se podrá determinar con el análisis de ruido blanco a los residuales del modelo por medio de la prueba *Ljung-Box*.

5.4 Análisis de los residuales del modelo SARIMA ajustado

Enseguida, se presenta la gráfica de rezagos para los residuales del modelo.

```
# Gráfica de rezagos
lag1.plot(residuales_inpc, max.lag = 16, main = "Lag plot", pch = 19, col = "blue")
```



En la salida anterior se puede apreciar que los gráficos de rezagos se asemejan a una nube de puntos aleatoria, por lo que el modelo propuesto da indicios de un buen ajuste. No obstante, se realizará la prueba *Ljung–Box* para determinar si los residuales siguen un comportamiento de ruido blanco. Las hipótesis de la prueba son las siguientes:

H_o : Los datos analizados son ruido blanco
H_a : Los datos analizados no son ruido blanco

Al igual que en la prueba anterior, la regla de decisión es rechazar la hipótesis nula (H_o) si el valor p es menor que un nivel de significancia de 0.05 ($\alpha = 0.05$). La salida de la prueba se muestra a continuación.

```
# Prueba de ruido blanco (Ljung-Box)
# Lag= 16 es un valor para el número de rezagos que sugirió el Dr. Francisco Corona
Box.test(residuales_inpc, type = "Ljung-Box", lag = 16)
```

```
#
#   Box-Ljung test
#
# data:  residuales_inpc
# X-squared = 22.555, df = 16, p-value = 0.1262
```

De acuerdo con la salida anterior, el valor p fue de 0.1262, por lo que no se rechaza la hipótesis nula propuesta, dado que $0.1262 > 0.05$, es decir, hay suficiente evidencia para determinar que los residuales se comportan como ruido blanco, por ende, el modelo propuesto se puede considerar como válido para generar pronósticos de la serie INPC.

5.5 Parámetros del modelo

A continuación, se verifica la significancia de los parámetros del modelo. Se tiene presente que la significancia se determina mediante la división de los coeficientes estimados entre el error estándar. Si el t-ratio resulta mayor que 2 en valor absoluto, se considera que el parámetro es significativo y necesario para continuar dentro del modelo.

```
# Significancia de los parámetros
modelo_inpc$coef/sqrt(diag(modelo_inpc$var.coef))
```

```
#         ar1         ma1        sma1
#   2.9562633  -0.3535565 -15.3376305
```

Según el resultado anterior, se tiene que tanto el coeficiente del componente *ar1* como el componente *sma1* son mayores a 2, por lo que resultan significativos en el modelo. No obstante, el parámetro *ma1* con el coeficiente absoluto igual a 0.3536 no resulta significativo.

Precisión del modelo ajustado

A través del resumen del modelo se identificarán las principales métricas de precisión del modelo generado.

```
# Parámetros del modelo
summary(modelo_inpc)
```

```
#
# Call:
# arima(x = tr_inpc, order = c(1, 1, 1), seasonal = list(order = c(0, 1, 1), period = 12))
#
# Coefficients:
#          ar1      ma1     sma1
#       0.5131  -0.0735  -0.8169
# s.e.  0.1736   0.2080   0.0533
#
# sigma^2 estimated as 0.05524:  log likelihood = -0.81,  aic = 9.62
#
# Training set error measures:
#                       ME     RMSE       MAE        MPE     MAPE      MASE
# Training set 0.03471903 0.228272 0.1500608 0.03368369 0.166831 0.3909924
#                      ACF1
# Training set -0.01587054
```

En la salida *summary* se observan diversas métricas del error de precisión del modelo. En general, estas métricas deberán ser lo más pequeñas posible y podrán ser un indicador para la comparación del modelo contra otros, o cuando se está iterando en el ajuste de los parámetros de éste.

5.6 Función auto.arima

Con el fin de realizar una comparación entre el modelo propuesto y el modelo automático generado con la función auto.arima, se realizó la ejecución de dicha función a los datos originales. El resultado se muestra a continuación.

```r
# Modelo ajustado automáticamente
auto.arima(tr_inpc)
```

```
# Series: tr_inpc
# ARIMA(1,1,0)(0,1,2)[12]
#
# Coefficients:
#          ar1     sma1    sma2
#       0.4750  -0.9554  0.1762
# s.e.  0.0646   0.1042  0.1092
#
# sigma^2 = 0.05539:  log likelihood = 0.55
# AIC=6.9   AICc=7.1   BIC=20.15
```

La función auto.arima sugiere un modelo $SARIMA(1,1,0)(0,1,2)_{[12]}$, el cual difiere en los parámetros q y Q respecto al que se propone en el análisis $SARIMA(1,1,1)(0,1,1)_{[12]}$.

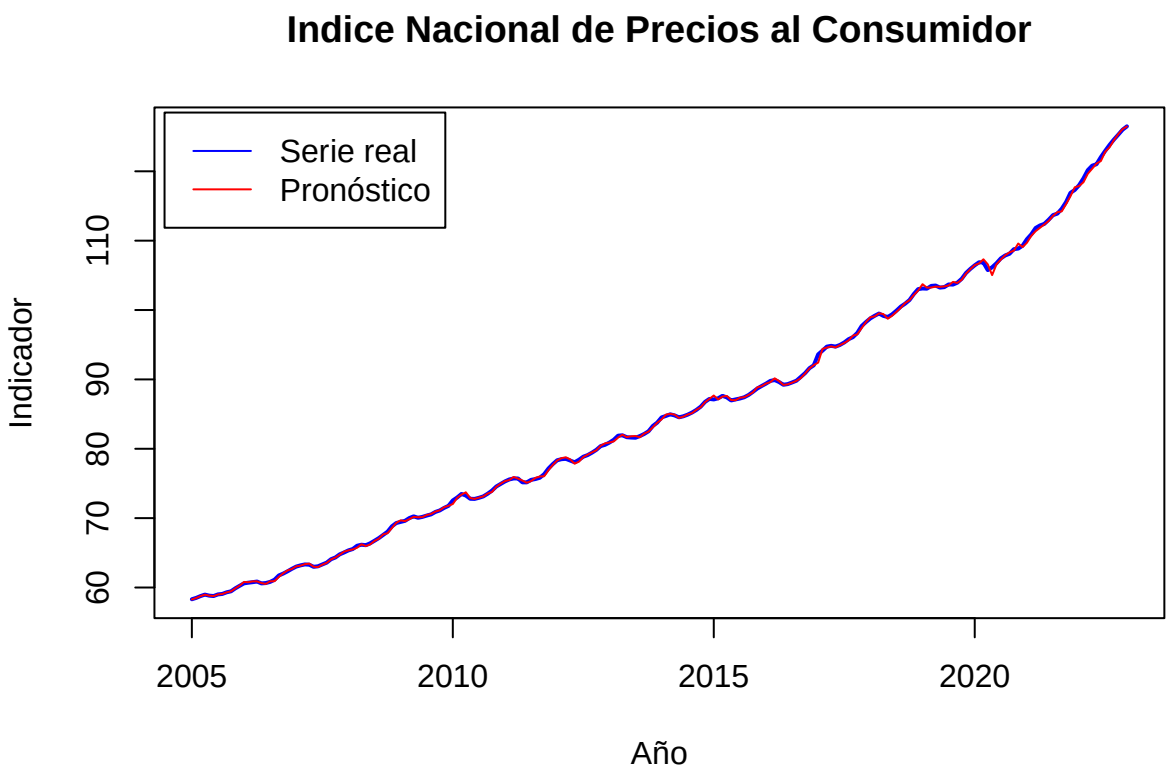
Comparativa a través del AIC

Una métrica de comparación entre los dos modelos puede ser el criterio el AIC (*Akaike Information Criteria*) ya que cuanto más pequeño resulte mejor será el modelo para la serie temporal. Para el modelo propuesto se tiene un valor AIC de 9.62, mientras que, para el modelo ajustado mediante *auto.arima* el valor es 6.9, el cual es menor al del modelo propuesto, por lo que en términos de AIC el modelo arrojado por *auto.arima* es mejor. Esto puede deberse a que el modelo generado por la función auto.arima emplea dos parámetros más que el modelo ajustado a través del análisis de las gráficas de autocorrelación.

5.7 Pronósticos para el INPC

Como parte final del análisis, se presenta el gráfico de la serie real contra la serie pronosticada con la intención de visualizar el ajuste. El resultado se muestra a continuación.

```r
# Comparativo de la serie pronosticada vs real
plot(tr_inpc, main = "Indice Nacional de Precios al Consumidor",
     ylab = "Indicador", xlab = "Año", col = "blue", lwd = 2)
# Serie pronosticada
lines(fitted(modelo_inpc), col = "red")
# Leyenda del gráfico
legend("topleft", inset = 0.01, legend = c("Serie real", "Pronóstico"),
       lty = c(1, 1), col = c("blue", "red"))
```



Con base en la gráfica previa, se observa que el modelo propuesto realiza un buen ajuste, ya que la línea azul (serie real) y la línea roja (serie pronosticada) son muy similares entre sí.

Finalmente, se evalúa el funcionamiento del modelo en términos de predicción con los 12 datos pertenecientes a 2022. Con la función que se ejecutará a continuación, se espera una tabla de salida con 3 campos:

- OBSERVADO: Son los datos observados en la serie de tiempo que se extrajeron con el horizonte H.
- PRONOSTICO: Es la estimación puntual para los datos observados, que se obtiene con la función *forecast*.
- ERROR: Es la diferencia entre lo OBSERVADO con el PRONOSTICO.

A continuación, se ejecuta el código.

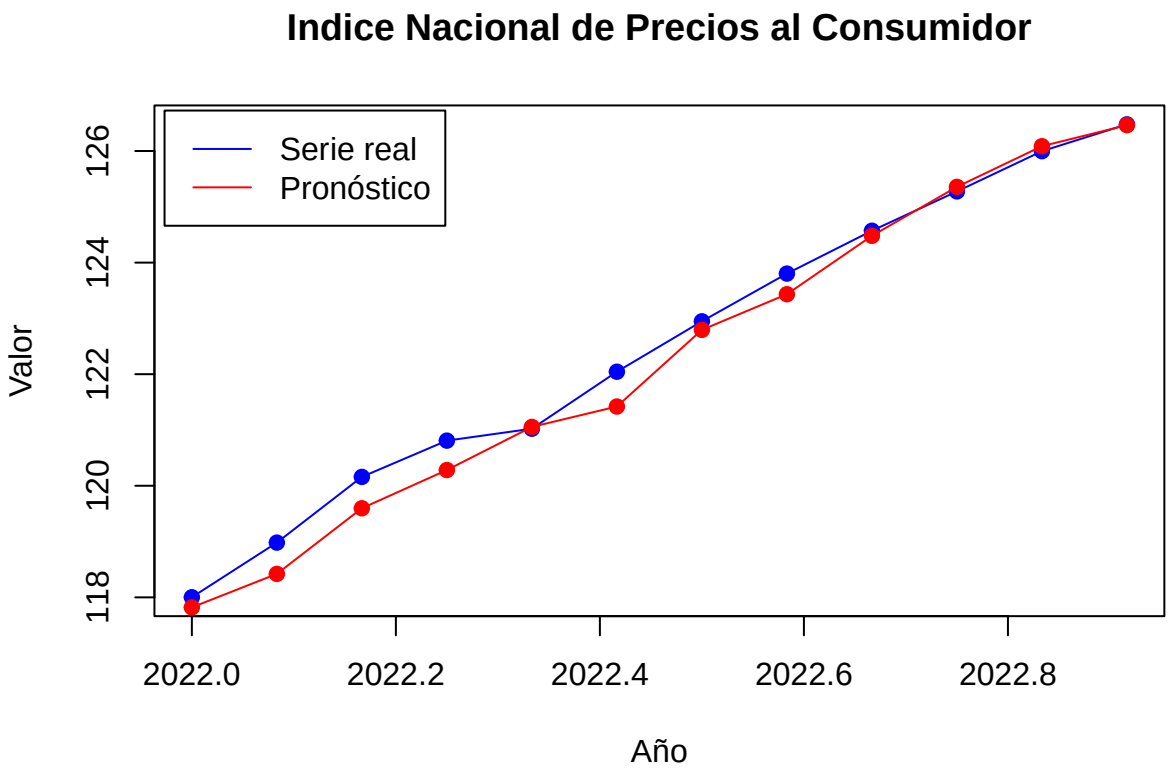
```
# Tamaño de la serie
n<-length(tr_inpc)
# Número de pronósticos
H<-12
# Se extraen los datos que serán los observados
observado<-tr_inpc[(n - H + 1):n]
# Tabla resumen de la salida de R
resumen<-data.frame(OBSERVADO = observado, PRONOSTICO = rep(0, H), ERROR = rep(0, H))

# Se crea el ciclo for para los pronósticos
for(h in 1:H){

# Se extraen los datos de interés de la serie
```

```r
serie<-ts(tr_inpc[1:(n - H - 1 + h)], start = c(2005, 1), frequency = 12)
# Ajuste del modelo SARIMA
modelo<-arima(serie, order = c(1, 1, 1),
              seasonal = list(order = c(0, 1, 1), period = 12))
# Se realiza el pronóstico y se extrae la estimación puntual
pronostico<-forecast(modelo, h = 1)$mean
# Se guarda el resultado del pronóstico en la tabla resumen
resumen[h, "PRONOSTICO"]<-pronostico

} # Del ciclo for

# Se genera el error
resumen[, "ERROR"]<-resumen$OBSERVADO - resumen$PRONOSTICO
# Se imprime la tabla resumen
resumen
```

```
#    OBSERVADO PRONOSTICO       ERROR
# 1    118.002   117.8202  0.18179846
# 2    118.981   118.4205  0.56046256
# 3    120.159   119.5966  0.56236328
# 4    120.809   120.2819  0.52714358
# 5    121.022   121.0550 -0.03297825
# 6    122.044   121.4204  0.62356045
# 7    122.948   122.7944  0.15359014
# 8    123.803   123.4333  0.36965111
# 9    124.571   124.4799  0.09114542
# 10   125.276   125.3578 -0.08180398
# 11   125.997   126.0863 -0.08933852
# 12   126.478   126.4631  0.01487134
```

Para apreciar de mejor manera los resultados obtenidos, a continuación, se presenta la siguiente gráfica temporal.

```r
# Convirtiendo a serie de tiempo
obs<-ts(resumen$OBSERVADO, start = c(2022, 1), frequency = 12)
pronost<-ts(resumen$PRONOSTICO, start = c(2022, 1), frequency = 12)
# Gráfico con los datos observados
plot(obs, xlab = "Año",  ylab = "Valor", type = "o", pch = 19, col = "blue",
     main = "Indice Nacional de Precios al Consumidor")
# Datos pronosticados
lines(pronost, col = "red", type = "o", pch = 19)
# Leyenda del gráfico
legend("topleft", inset = 0.01, legend = c("Serie real", "Pronóstico"),
       lty = c(1, 1), col = c("blue", "red"))
```



La salida anterior muestra la representación gráfica de los datos observados (línea azul) vs los datos pronosticados (línea roja) para un horizonte H de longitud 12. Finalmente, se puede observar que la serie pronosticada muestra pequeñas diferencias con respecto a la serie real.

Para fines comparativos de los modelos ajustados, se calculó el error de predicción a través del error cuadrático medio (MSE, por sus siglas en inglés), cuya expresión es la siguiente:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{f}(X_i))^2 = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 = \frac{1}{n}RSS$$

```
# Se realizan los cálculos del MSE
MSE_INPC<-sum(resumen$ERROR^2)/12
MSE_INPC
```

```
# [1] 0.1262209
```

De acuerdo con la salida previa, el valor del *MSE* para el modelo SARIMA ajustado para realizar pronósticos del INPC, fue de 0.1262209

6. Modelo VAR/VEC

6.1 Selección de variables

En este apartado, se trabaja con un ajuste de los modelos Vectoriales Autorregresivos (VAR)/ Vectores de Corrección de Error (VEC). Para poder realizar el ajuste con estos modelos primero se selecciona un subconjunto de series (de un total de 28), la finalidad es realizar un pronóstico a la variable de interés (INPC).

Con respecto a la selección de variables, se implementaron 3 métodos. Con respecto al criterio para la selección del mejor modelo, se utilizó el número de variables y el valor del error cuadrático medio que reportaron.

```
#####################
# Modelo VAR/VEC
#####################

# Se limpia el espacio de trabajo
rm(list = ls())
invisible(gc(reset = TRUE))

# Se cargan las librerías
library(astsa)       # Gráfico de retrasos
library(tseries)     # Prueba de ADF
library(forecast)    # Pronósticos
library(seasonal)    # Estacionalidad
library(vars)        # Modelos VAR/VEC
library(dlm)         # Análisis de modelos dinámicos lineales
library(kableExtra)  # Edición de tablas
library(corrplot)    # Graficar las correlaciones
library(glmnet)      # Para modelo LASSO o RIDGE
library(leaps)       # Contiene la función regsubsets()

# Establecer dirección de trabajo
ruta<-("C:/Proyecto de series/")
# Se cargan las funciones
source(paste(ruta, "functions.r", sep = ""))

# Se leen los datos
datos<-read.csv(file = paste0(ruta, 'IOAE2.csv'))
# Se elimina la primera columna
datos_ioae<-datos[, -1]
```

Antes de comenzar con el ajuste del modelo VAR/VEC, se implementan métodos de selección de variables para determinar las que más aporten para realizar predicciones del INPC como si se tratase de un modelo de regresión. Con el propósito de no generar un documento demasiado extenso, se mostrará el código utilizado para la selección de variables, sin ejecutar alguna salida.

a) Método de selección de variables con un modelo lineal con regularización (LASSO)

El modelo lineal con regularización LASSO permite que algunos coeficientes sean exactamente igual a cero, por lo que puede ser visto como un método de selección de variables. Este modelo es relativamente de reciente creación (1996), por Robert Tibshirani. Dado que las técnicas tradicionales basadas en criterios de información quedan rebasadas al incrementarse exponencialmente la cantidad de covariables. Minimizar la suma de cuadrados residuales sujeto a que la suma del valor absoluto de los coeficientes sea menor que una constante, permite alcanzar dos objetivos:

i) Dejar fuera todas las variables irrelevantes y retener aquellas que sí lo son.
ii) Estimar los coeficientes con la misma velocidad y distribución con la que se hubieran estimado de conocer desde un principio cuáles eran las variables relevantes.

El primer paso es construir un conjunto de entrenamiento y otro de validación, para lo cual se dividen los datos en un conjunto de entrenamiento (75%) y un conjunto de validación (25%). Esto se realiza de manera aleatoria.

```
# Tamaño de los datos
n<-nrow(datos)
# Se fija una semilla para la replicabilidad
set.seed(12345)
# Selección de la muestra
trainIndex<-sample(1:n, size = round(0.75*n), replace = FALSE)
datos_train<-datos[trainIndex,]

# Se comprueba la dimensión del conjunto de entrenamiento
dim_train<-dim (datos_train)
# Se construye el conjunto de entrenamiento/validación
datos_test = datos[-trainIndex,]
# Se comprueba la dimensión del conjunto de prueba
dim_test<-dim(datos_test)#54 observaciones, 29 variables.
# Se elimina la primera columna
datos_train<-datos_train[,-1]
# Se revisa el tipo de dato del conjunto de datos de entrenamiento
str(datos_train)
# Se elimina la primera columna
datos_test<-datos_test[,-1]
# Se revisa el tipo de dato del conjunto de datos de prueba
str(datos_test)
```

Además de revisar algunas características de las variables, se generaron los datos de prueba y entrenamiento. Se procede al ajuste del modelo de regresión.

```
# Para X se usa la función model.matrix() para generar la matriz de diseño
# Se le quita la primera columna porque el intercepto no
# se contempla en la contracción de los parámetros
model_matrix<-model.matrix(INPC ~ ., datos_train)
x_train<-model.matrix(INPC ~ ., datos_train)[,-1]
y_train<-datos_train$INPC
```

Se emplea la biblioteca *glmnet*. Nótese que $\alpha = 1$ para regresión LASSO. Nota: Si no se le pone la cuadrícula para los valores de lambda, la función automáticamente crea unos valores de lambda apropiados.

```
# Modelo de regresión LASSO
lasso.mod<-glmnet(x_train, y_train, alpha = 1)
lasso.mod
```

Ahora, se obtiene el mejor valor para lambda haciendo validación cruzada. Para seleccionar el valor de lamba óptimo se usa la función *cv.glmnet()*. Para obtener el mejor valor de lambda, se elige aquella que minimice el *MSE* y nuevamente se construye el modelo de regresión LASSO usando la mejor lambda.

```
# Determinar la mejor lambda
cv.out_lasso<-cv.glmnet(x_train, y_train, alpha = 1)
# Lambda óptima queda con una lambda de 0.02241.
mejor_lambda_lasso<-cv.out_lasso$lambda.min
# El modelo es:
lasso.mod.final<-glmnet(x_train, y_train, alpha = 1, lambda = mejor_lambda_lasso)
lasso.mod.final
```

Finalmente, se muestran los coeficientes usando la lambda elegida por validación cruzada.

```
# Determinar los coeficientes
out_lasso<-glmnet(x_train, y_train, alpha = 1)
# Predicciones
predict(out_lasso, type = "coefficients", s = mejor_lambda_lasso)
```

La salida muestra que en el modelo LASSO algunos coeficientes tienen el valor de cero, éste sugiere considerar las siguientes 20 variables: "IAI", "TDU", "M", "CONF_MAN", "TIIE_28", "SP_500", "IPI_EUA", "ANTAD", "PROD_VEH", "OCUP_HOT", "REMESAS", "M4", "X", "PEDIDOS_MANU", "IMEF", "PRECIO", "TOTAL_NONFARM", "TOTAL_MANUF", "TOTAL_SERV" y "MANUF_USA".

Considerar 20 variables no es lo óptimo, ya que, según lo comentado en clase, los modelos VAR/VEC requieren un número menor de series de tiempo para que sea factible la ejecución de las pruebas, por ejemplo *Johansen*.

Con base en lo obtenido, se decidió realizar otros métodos de selección de variables y ver qué resultados arrojan, para comparar el error de predicción y elegir el de menor error.

b) Método de selección de variables del mejor subconjunto

Este método considera la métrica asociada a modelos con mejor poder predictivo y se realiza con el conjunto de datos de entrenamiento. Para este ejemplo, se forzó el modelo para considerar 5 variables.

```
# Método selección del mejor subconjunto
regfit_completo<-regsubsets(INPC ~ ., data = datos_train, nvmax = 5)
# Resumen del resultado
summary(regfit_completo)
```

El resultado que se obtiene en *summary* mediante este método, es que para el modelo con 1 variable se elige a la variable predictora "M4"; para el modelo con 2 variables se eligen las variables "M4" y "REMESAS"; para el modelo con 3 variables, las seleccionadas son "IAI", "M" y "M4" y así sucesivamente, hasta considerar al modelo que contenga las 5 variables predictoras. Lo siguiente es considerar la métrica con mejor poder predictivo y ver cuáles son las variables seleccionadas. Para la métrica con mejor poder predictivo se utilizó la estadística de *Mallow* (Cp).

```
# Se puede guardar la aplicación de la función summary() para obtener otros indicadores
reg_summary<-summary(regfit_completo)
# Usamos estadístico Cp-Mallow (modelos con mejor poder predictivo)
plot(reg_summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
# Determinar el mínimo Cp
minimo_cp<-which.min(reg_summary$cp)
# Se coloca el mínimo porque la regla dice que se selecciona al menor
points (minimo_cp, reg_summary$cp[minimo_cp] , col = "red", cex = 2, pch = 20)
# Variables seleccionadas
var_sel<-c("INPC", names(coef(regfit_completo, 5))[-1])
```

```r
modelo1<-lm(INPC ~ ., data = datos_train[, var_sel])
# Resumen del modelo
summary(modelo1)
```

En el summary del modelo, las 5 variables son significativas; además, el R^2_{adj} es de 0.997, es decir, es un buen ajuste al modelo. Las variables seleccionadas son: "IAI", "IPI_EUA", "M4", "X" y "TOTAL_SERV".

c) Método de selección hacia adelante

Nuevamente, se considera la métrica asociada a modelos con mejor poder predictivo. Se realiza con el conjunto de datos de entrenamiento. Para este ejemplo, se forzó el modelo para considerar 5 variables.

```r
# Método selección hacia adelante.
regfit_completo<-regsubsets(INPC ~ ., data=datos_train, nvmax=5, method = "forward")
# Resumen del modelo
summary(regfit_completo)
```

Para este caso, lo que arroja *summary* es que en el modelo con 1 variable se elige a la variable predictora "M4"; para el modelo con 2 variables se eligen las variables "M4" y "REMESAS"; para el modelo con 3 variables se seleccionan "REMESAS", "M4" y "TOTAL_MANUF" y así sucesivamente, hasta considerar al modelo que contenga las 5 variables predictoras.

```r
# Se puede guardar la aplicación de la función summary() para obtener otros indicadores
reg_summary<-summary(regfit_completo)
# Usamos estadístico Cp-Mallow (modelos con mejor poder predictivo)
plot(reg_summary$cp, xlab = "Number of Variables", ylab = "Cp", type = "l")
# Determinar el valor mínimo del Cp
minimo_cp<-which.min(reg_summary$cp)
# Se coloca el mínimo porque la regla nos dice que se selecciona al menor
points(minimo_cp, reg_summary$cp[minimo_cp] , col = "red", cex = 2, pch = 20)

# Variables seleccionadas
var_sel<-c("INPC", names(coef(regfit_completo, 5))[-1])
modelo2<-lm(INPC ~ ., data = datos_train[, var_sel])
# Resumen del modelo
summary(modelo2)
```

Para este caso, las 5 variables son significativas; además, el R^2_{adj} es de 0.9964, es decir, es un buen ajuste al modelo. Las variables seleccionadas son: "M", "TIIE_28", "REMESAS", "M4" y "TOTAL_MANUF". Enseguida, se calcula el error de predicción, recordando que el error de predicción se mide con el error cuadrático medio. A continuación, se trabaja con el conjunto de datos de prueba.

```r
# Datos de prueba y de entrenamiento
model_matrix_pr<-model.matrix(INPC ~ ., datos_test)
x_test<-model.matrix(INPC ~., datos_test)[,-1]
y_test<-datos_test$INPC
```

Ahora, se obtienen los errores de predicción de los métodos de selección de variables aplicados.

Error de predicción del modelo de regresión LASSO

```r
# Pronóstico del modelo LASSO
lasso_pred<-predict(lasso.mod.final, s = mejor_lambda_lasso, newx = x_test)
# Cálculo del MSE (arroja 0.9704037)
MSE_lasso<-mean((lasso_pred - y_test)^2)
```

Error de predicción del modelo de mejor subconjunto

```
# Pronóstico del modelo
normal_pred = predict(modelo1, newx = x_test)
# Cálculo del MSE (arroja 530.9281)
MSE_mej_sub<-(mean((normal_pred - y_test)^2))
```

Error de predicción del modelo selección hacia adelante

```
# Pronóstico del modelo
normal_pred = predict(modelo2, newx = x_test)
# Cálculo del MSE (arroja 537.3517)
MSE_ade<-mean((normal_pred - y_test)^2)
```

Con base en lo anterior se obtiene que el MSE es: 1. Modelo de regresión lasso: 0.9704037 2. Modelo de mejor subconjunto: 530.9281 3. Modelo de selección hacia adelante: 537.3517

A pesar de que el modelo LASSO tiene un error de predicción menor, éste considera a 20 variables, lo que implica que quizá el ajuste del modelo VAR/VEC no sea viable, por lo que, de alguna manera se invalida este método de selección; por otro lado, se realizó el método del mejor subconjunto, con la condición de que se mostrara el mejor modelo con 5 variables; al realizar la regresión, todas las variables fueron significativas, el R^2_{adj} fue bastante bueno, con un valor de 0.997 y el error de predicción fue de 530.9281. Finalmente, con el método de selección hacia adelante, las 5 variables seleccionadas fueron significativas, el R^2 fue de 0.9969 y el error de predicción fue de 537.3517.

Por todo lo anterior, se decidió considerar el modelo de selección del mejor subconjunto, debido a que, al condicionar 5 variables, el modelo de regresión se ajusta bien, a diferencia del modelo *LASSO* que se compone de 20 variables; además, con respecto al modelo de selección hacia adelante, el modelo del mejor subconjunto tiene un error de predicción menor.

6.2 Preparación de las series

Con base en lo anterior, se seleccionan las variables INPC, IAI, IPI_EUA, M4, X y TOTAL_SERV para trabajar en el ajuste del modelo VAR/VEC.

```
# Se convierten a series de tiempo
datos1<-ts(datos_ioae, start = c(2005, 1), frequency = 12)
# Se seleccionan las variables de interés
ioae_int<-datos1[, c("INPC", "IAI", "IPI_EUA", "M4", "X","TOTAL_SERV")]
# Estadística básica y valores nulos
summary(ioae_int)
```

```
#       INPC             IAI            IPI_EUA             M4
#  Min.   : 58.31   Min.   : 72.60   Min.   : 84.73   Min.   : 2772
#  1st Qu.: 70.34   1st Qu.: 97.22   1st Qu.: 96.76   1st Qu.: 4712
#  Median : 84.14   Median : 99.67   Median : 99.18   Median : 8631
#  Mean   : 85.45   Mean   : 99.14   Mean   : 98.42   Mean   : 8605
#  3rd Qu.: 99.60   3rd Qu.:102.78   3rd Qu.:101.74   3rd Qu.:11926
#  Max.   :126.48   Max.   :106.10   Max.   :104.83   Max.   :16448
#        X           TOTAL_SERV
#  Min.   :16376   Min.   : 89084
#  1st Qu.:23902   1st Qu.: 92173
#  Median :31272   Median : 96420
```

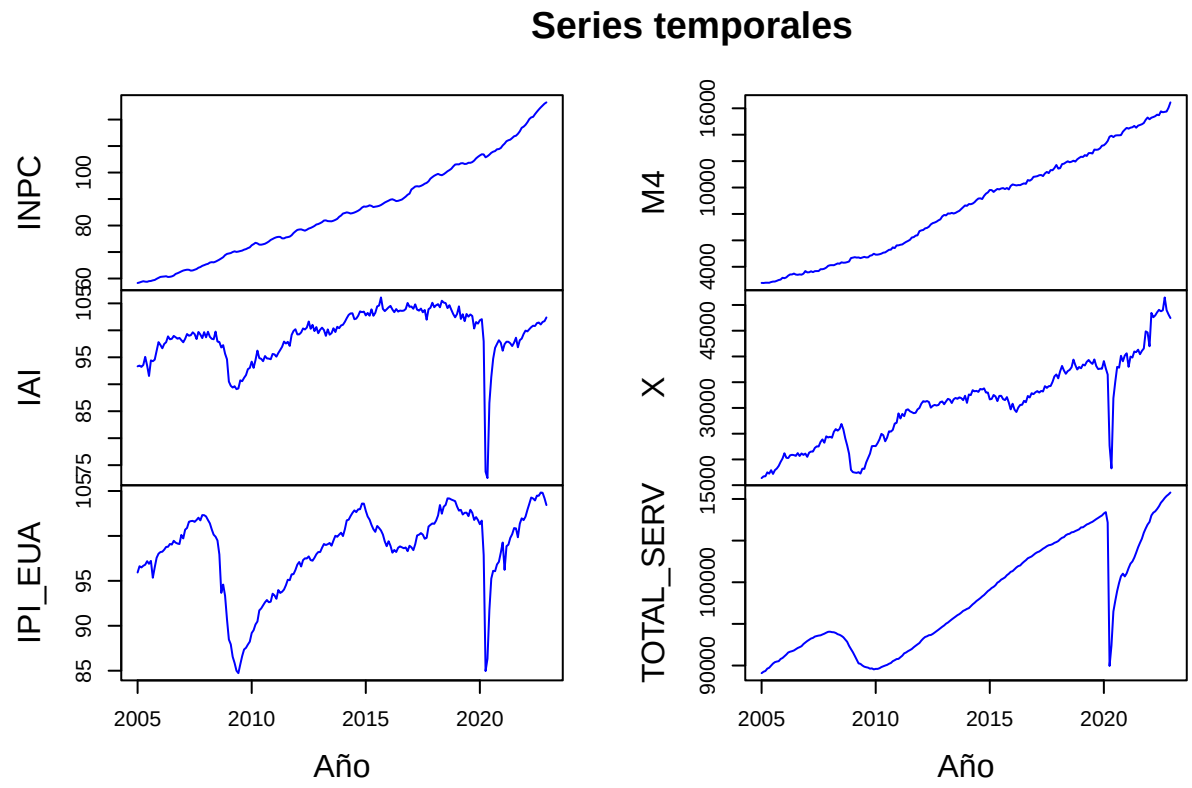
```
#  Mean   :30608   Mean   : 97856
#  3rd Qu.:36533   3rd Qu.:103732
#  Max.   :51460   Max.   :110773
```

```
# Número de series
N<-ncol(ioae_int)
```

La salida anterior muestra que los valores promedio de las series de tiempo en el periodo bajo estudio son los siguientes: Índice Nacional de Precios al Consumidor (INPC), con un valor de 85.45; Índice de producción industrial (IAI), de 99.14; Índice de producción industrial de los Estados Unidos (IPI_EUA), de 98.42; Agregado monetario (M4), de 8,605; Exportaciones totales (X), de 30,608 y, finalmente, para TOTAL_SERV, es de 97,856. Además, no se cuenta con valores faltantes en ninguna de las variables de interés.

Se realiza el análisis exploratorio de los datos. A continuación, se muestran las gráficas de las series de tiempo.

```
# Análisis exploratorio
plot(ioae_int, xlab = "Año", main = "Series temporales", col = "blue")
```



La salida anterior muestra el gráfico temporal de las series de tiempo. Las gráficas del INPC y M4 muestran una tendencia creciente; además, son muy parecidas entre ellas. De manera particular, el IAI muestra un valor mínimo en el 2009, luego se observa un comportamiento ligeramente al alza, casi constante; además, se muestra un valor atípico cerca del año 2020, muy probablemente por motivo de la pandemia por COVID-19.

Con respecto a las series X y TOTAL SERV, éstas muestran un comportamiento similar entre ellas. En 2010, se muestra un mínimo y luego una tendencia al alza; después muestran un valor atípico cerca del año 2020 y en el periodo final se nota una tendencia creciente. Finalmente, la serie IPI_EUA reporta un mínimo en 2010, después muestra una tendencia al alza y cerca del año 2020 muestra un valor atípico; como ya se

mencionó, probablemente es por un tema de pandemia. En el último periodo registrado esta variable muestra un comportamiento al alza.

Una vez analizadas las series de tiempo, es importante comentar que algunas series ya están desestacionalizadas y por lo regular son las que se descargaron directamente de la página de INEGI; sin embargo, otras provienen de otras dependencias, por ejemplo, del Banco de México y al parecer no están desestacionalizadas, por lo tanto, la siguiente parte del análisis consiste en analizar formalmente los problemas de estacionalidad. El método visto en clase se basa en incluir variables ficticias estacionales y comprobar si tienen valores p significativos al calcular la regresión. Si los meses individuales tienen coeficientes significativos, la serie de tiempo tiene problemas de estacionalidad. El código implementado se muestra enseguida. Cabe destacar que, para todas las pruebas se considera un $\alpha = 0.05$.

```
# Modelo para determinar estacionalidad
for(i in 1:N){
estacionalidad<-lm(ioae_int[, i] ~ c(1:nrow(ioae_int)) + seasonaldummy(ioae_int[, i]))
# Resumen del modelo
print(colnames(ioae_int)[i])
print(summary(estacionalidad)) }
```

```
# [1] "INPC"
#
# Call:
# lm(formula = ioae_int[, i] ~ c(1:nrow(ioae_int)) + seasonaldummy(ioae_int[,
#     i]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -4.3693 -1.4974 -0.4229  0.8955  9.5080
#
# Coefficients:
#                                 Estimate Std. Error t value Pr(>|t|)
# (Intercept)                    54.525198   0.714241  76.340   <2e-16 ***
# c(1:nrow(ioae_int))             0.289096   0.002928  98.738   <2e-16 ***
# seasonaldummy(ioae_int[, i])Jan -0.135415   0.893588  -0.152    0.880
# seasonaldummy(ioae_int[, i])Feb -0.099660   0.893487  -0.112    0.911
# seasonaldummy(ioae_int[, i])Mar -0.015649   0.893396  -0.018    0.986
# seasonaldummy(ioae_int[, i])Apr -0.342502   0.893314  -0.383    0.702
# seasonaldummy(ioae_int[, i])May -0.819052   0.893242  -0.917    0.360
# seasonaldummy(ioae_int[, i])Jun -0.883875   0.893180  -0.990    0.324
# seasonaldummy(ioae_int[, i])Jul -0.823016   0.893127  -0.921    0.358
# seasonaldummy(ioae_int[, i])Aug -0.820070   0.893084  -0.918    0.360
# seasonaldummy(ioae_int[, i])Sep -0.700815   0.893050  -0.785    0.434
# seasonaldummy(ioae_int[, i])Oct -0.512960   0.893026  -0.574    0.566
# seasonaldummy(ioae_int[, i])Nov -0.146800   0.893012  -0.164    0.870
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 2.679 on 203 degrees of freedom
# Multiple R-squared:  0.9797,  Adjusted R-squared:  0.9785
# F-statistic:   815 on 12 and 203 DF,  p-value: < 2.2e-16
#
# [1] "IAI"
#
# Call:
```

```
# lm(formula = ioae_int[, i] ~ c(1:nrow(ioae_int)) + seasonaldummy(ioae_int[,
#     i]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -27.487  -2.390   1.008   3.100   6.188
#
# Coefficients:
#                                 Estimate Std. Error t value Pr(>|t|)
# (Intercept)                    96.938759   1.196309  81.032  < 2e-16 ***
# c(1:nrow(ioae_int))             0.026553   0.004904   5.415 1.73e-07 ***
# seasonaldummy(ioae_int[, i])Jan -0.535242   1.496702  -0.358    0.721
# seasonaldummy(ioae_int[, i])Feb -0.384985   1.496533  -0.257    0.797
# seasonaldummy(ioae_int[, i])Mar -0.569794   1.496380  -0.381    0.704
# seasonaldummy(ioae_int[, i])Apr -1.764849   1.496244  -1.180    0.240
# seasonaldummy(ioae_int[, i])May -1.762472   1.496123  -1.178    0.240
# seasonaldummy(ioae_int[, i])Jun -0.959857   1.496019  -0.642    0.522
# seasonaldummy(ioae_int[, i])Jul -0.813757   1.495930  -0.544    0.587
# seasonaldummy(ioae_int[, i])Aug -0.415051   1.495858  -0.277    0.782
# seasonaldummy(ioae_int[, i])Sep -0.455767   1.495802  -0.305    0.761
# seasonaldummy(ioae_int[, i])Oct -0.339736   1.495762  -0.227    0.821
# seasonaldummy(ioae_int[, i])Nov -0.138461   1.495737  -0.093    0.926
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 4.487 on 203 degrees of freedom
# Multiple R-squared:  0.1401,  Adjusted R-squared:  0.08926
# F-statistic: 2.756 on 12 and 203 DF,  p-value: 0.001723
#
# [1] "IPI_EUA"
#
# Call:
# lm(formula = ioae_int[, i] ~ c(1:nrow(ioae_int)) + seasonaldummy(ioae_int[,
#     i]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -15.117  -1.404   1.050   2.939   6.139
#
# Coefficients:
#                                 Estimate Std. Error t value Pr(>|t|)
# (Intercept)                    95.469471   1.129953  84.490  < 2e-16 ***
# c(1:nrow(ioae_int))             0.029727   0.004632   6.418 9.58e-10 ***
# seasonaldummy(ioae_int[, i])Jan -0.180158   1.413684  -0.127    0.899
# seasonaldummy(ioae_int[, i])Feb -0.276057   1.413525  -0.195    0.845
# seasonaldummy(ioae_int[, i])Mar -0.264984   1.413381  -0.187    0.851
# seasonaldummy(ioae_int[, i])Apr -0.863349   1.413252  -0.611    0.542
# seasonaldummy(ioae_int[, i])May -0.811753   1.413138  -0.574    0.566
# seasonaldummy(ioae_int[, i])Jun -0.400696   1.413039  -0.284    0.777
# seasonaldummy(ioae_int[, i])Jul -0.081056   1.412956  -0.057    0.954
# seasonaldummy(ioae_int[, i])Aug  0.040051   1.412887   0.028    0.977
# seasonaldummy(ioae_int[, i])Sep -0.306548   1.412834  -0.217    0.828
# seasonaldummy(ioae_int[, i])Oct -0.076564   1.412796  -0.054    0.957
# seasonaldummy(ioae_int[, i])Nov -0.017262   1.412774  -0.012    0.990
```

```
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 4.238 on 203 degrees of freedom
# Multiple R-squared:  0.1735,  Adjusted R-squared:  0.1247
# F-statistic: 3.552 on 12 and 203 DF,  p-value: 8.29e-05
#
# [1] "M4"
#
# Call:
# lm(formula = ioae_int[, i] ~ c(1:nrow(ioae_int)) + seasonaldummy(ioae_int[,
#     i]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -786.48 -205.96  -53.45  289.64 1012.34
#
# Coefficients:
#                                 Estimate Std. Error t value Pr(>|t|)
# (Intercept)                    1745.0927   110.8907  15.737   <2e-16 ***
# c(1:nrow(ioae_int))              64.2031     0.4546 141.237   <2e-16 ***
# seasonaldummy(ioae_int[, i])Jan  -42.4615   138.7354  -0.306    0.760
# seasonaldummy(ioae_int[, i])Feb  -81.0559   138.7198  -0.584    0.560
# seasonaldummy(ioae_int[, i])Mar  -80.3755   138.7056  -0.579    0.563
# seasonaldummy(ioae_int[, i])Apr  -95.6049   138.6930  -0.689    0.491
# seasonaldummy(ioae_int[, i])May  -97.3305   138.6818  -0.702    0.484
# seasonaldummy(ioae_int[, i])Jun -129.2963   138.6721  -0.932    0.352
# seasonaldummy(ioae_int[, i])Jul -124.0250   138.6639  -0.894    0.372
# seasonaldummy(ioae_int[, i])Aug -147.8663   138.6572  -1.066    0.288
# seasonaldummy(ioae_int[, i])Sep -167.5864   138.6520  -1.209    0.228
# seasonaldummy(ioae_int[, i])Oct -170.2629   138.6482  -1.228    0.221
# seasonaldummy(ioae_int[, i])Nov -132.6925   138.6460  -0.957    0.340
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 415.9 on 203 degrees of freedom
# Multiple R-squared:   0.99,   Adjusted R-squared:  0.9894
# F-statistic:  1667 on 12 and 203 DF,  p-value: < 2.2e-16
#
# [1] "X"
#
# Call:
# lm(formula = ioae_int[, i] ~ c(1:nrow(ioae_int)) + seasonaldummy(ioae_int[,
#     i]))
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -20516.4  -1455.0    293.3   1737.8   8060.4
#
# Coefficients:
#                                 Estimate Std. Error t value Pr(>|t|)
# (Intercept)                    17973.204    870.300  20.652   <2e-16 ***
# c(1:nrow(ioae_int))              118.490      3.568  33.212   <2e-16 ***
# seasonaldummy(ioae_int[, i])Jan  -471.267   1088.833  -0.433    0.666
```

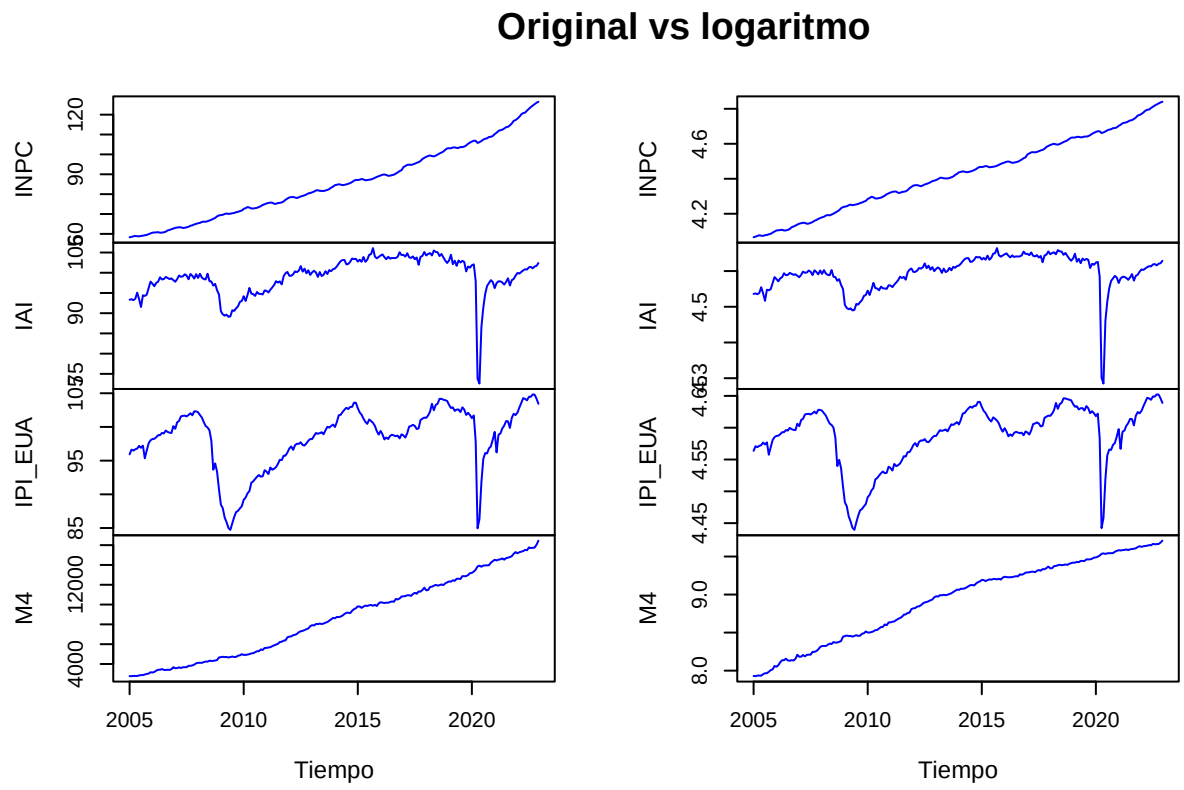
```
# seasonaldummy(ioae_int[, i])Feb  -257.625   1088.710  -0.237    0.813
# seasonaldummy(ioae_int[, i])Mar  -155.938   1088.599  -0.143    0.886
# seasonaldummy(ioae_int[, i])Apr  -796.821   1088.500  -0.732    0.465
# seasonaldummy(ioae_int[, i])May -1113.260   1088.412  -1.023    0.308
# seasonaldummy(ioae_int[, i])Jun  -288.325   1088.336  -0.265    0.791
# seasonaldummy(ioae_int[, i])Jul  -152.399   1088.272  -0.140    0.889
# seasonaldummy(ioae_int[, i])Aug   114.330   1088.219   0.105    0.916
# seasonaldummy(ioae_int[, i])Sep   188.171   1088.178   0.173    0.863
# seasonaldummy(ioae_int[, i])Oct    79.485   1088.149   0.073    0.942
# seasonaldummy(ioae_int[, i])Nov   197.105   1088.131   0.181    0.856
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 3264 on 203 degrees of freedom
# Multiple R-squared:  0.8458,  Adjusted R-squared:  0.8367
# F-statistic: 92.79 on 12 and 203 DF,  p-value: < 2.2e-16
#
# [1] "TOTAL_SERV"
#
# Call:
# lm(formula = ioae_int[, i] ~ c(1:nrow(ioae_int)) + seasonaldummy(ioae_int[,
#     i]))
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -14102.5  -2288.6    869.3   2463.9   3983.0
#
# Coefficients:
#                                Estimate Std. Error t value Pr(>|t|)
# (Intercept)                   88236.822    814.131 108.382   <2e-16 ***
# c(1:nrow(ioae_int))              91.041      3.337  27.279   <2e-16 ***
# seasonaldummy(ioae_int[, i])Jan   -67.937   1018.560  -0.067    0.947
# seasonaldummy(ioae_int[, i])Feb    14.244   1018.445   0.014    0.989
# seasonaldummy(ioae_int[, i])Mar    -8.242   1018.341  -0.008    0.994
# seasonaldummy(ioae_int[, i])Apr  -942.894   1018.248  -0.926    0.356
# seasonaldummy(ioae_int[, i])May  -761.657   1018.166  -0.748    0.455
# seasonaldummy(ioae_int[, i])Jun  -486.587   1018.095  -0.478    0.633
# seasonaldummy(ioae_int[, i])Jul  -359.073   1018.035  -0.353    0.725
# seasonaldummy(ioae_int[, i])Aug  -247.614   1017.986  -0.243    0.808
# seasonaldummy(ioae_int[, i])Sep  -164.044   1017.947  -0.161    0.872
# seasonaldummy(ioae_int[, i])Oct   -74.251   1017.920  -0.073    0.942
# seasonaldummy(ioae_int[, i])Nov    -9.903   1017.904  -0.010    0.992
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 3054 on 203 degrees of freedom
# Multiple R-squared:  0.7869,  Adjusted R-squared:  0.7743
# F-statistic: 62.47 on 12 and 203 DF,  p-value: < 2.2e-16
```

Como se muestra en la salida previa, los coeficientes de las variables ficticias estacionales resultaron no ser significativas, excepto en el intercepto y en el relacionado con la tendencia, por lo que no hay estacionalidad en las series de tiempo.

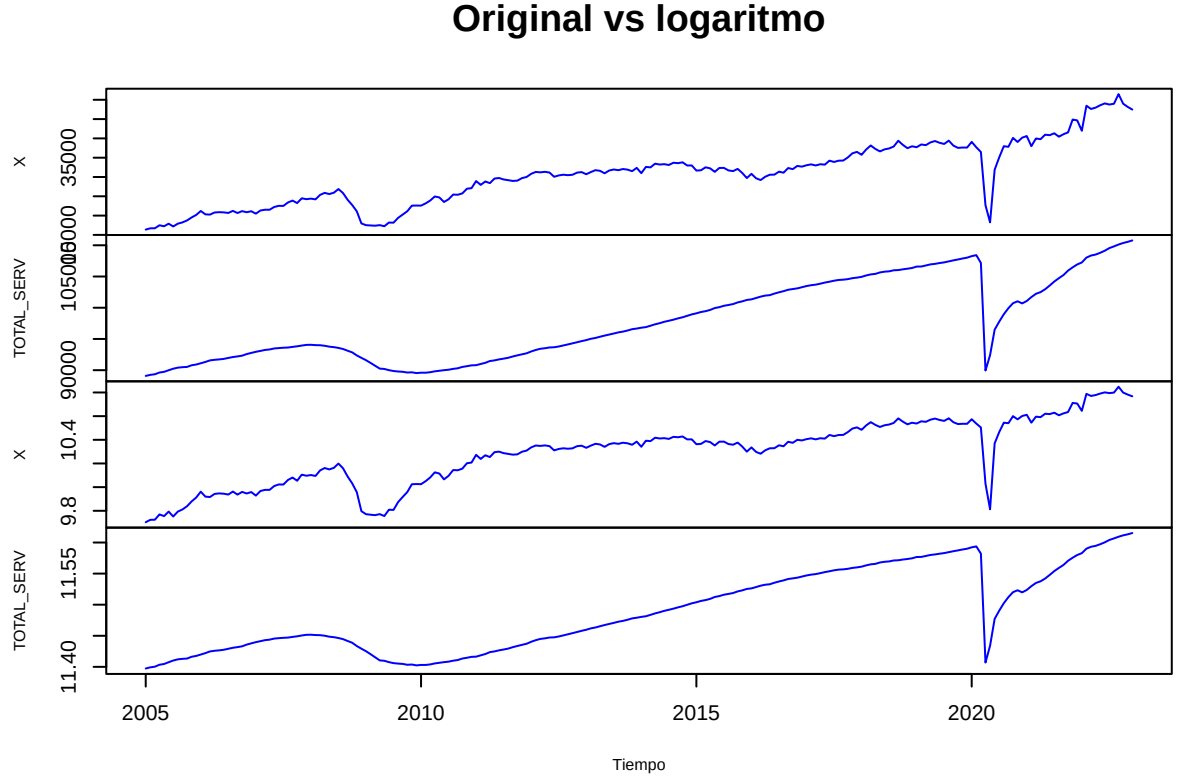
Posteriormente, se aplicaron logaritmos a las series con la intención de observar si la varianza tiene una mejora relevante. No obstante, al comparar la serie contra su logaritmo no se observaron cambios relevantes,

por lo que se optó por trabajar con las series sin ningún tipo de transformación. A continuación, se presentan las gráficas.

```
# Gráfico temporal para las 4 primeras series
series<-cbind(ioae_int[, 1:4], log(ioae_int[, 1:4]))
colnames(series)<-rep(colnames(ioae_int)[1:4], 2)
# Se realiza el gráfico temporal
plot(series, cex.lab = 0.75, main = "Original vs logaritmo",
     xlab = "Tiempo", col= "blue")
```




```
# Gráfico temporal para las 2 series restantes
series<-cbind(ioae_int[, 5:6], log(ioae_int[, 5:6]))
colnames(series)<-rep(colnames(ioae_int)[5:6], 2)
# Se realiza el gráfico temporal
plot(series, cex.lab = 0.5, main = "Original vs logaritmo", xlab = "Tiempo",
     col = "blue")
```



De acuerdo con la gráfica previa, la aplicación de logaritmo no modifica en nada la varianza de la serie, por lo que se optó por trabajar con los datos originales. La siguiente parte del análisis consiste en aplicar la prueba aumentada de *Dickey Fuller* para evaluar la posibilidad de que exista una raíz unitaria debido a la tendencia de los datos. Las hipótesis de la prueba son las siguientes:

H_o : La serie de tiempo es no estacionaria
H_a : La serie de tiempo es estacionaria

La regla de decisión es rechazar la hipótesis nula (H_o) si el valor p es menor que un nivel de significancia de 0.05 ($\alpha = 0.05$). La salida de los valores p de la prueba, tanto para la serie original como en primeras diferencias se muestra a continuación.

```
# Matriz para pruebas ADF
adf_tests<-matrix(NA, N, 2)
# Nombres de columnas y renglones
colnames(adf_tests)<-c("Original", "Diferenciada")
rownames(adf_tests)<-colnames(ioae_int)

# Ciclo for para las pruebas
for(i in 1:N){
adf_tests[i, "Original"]<-adf(ioae_int[, i], "const")$p.value
adf_tests[i, "Diferenciada"]<-adf(diff(ioae_int[, i]), "const")$p.value }

# Se imprime el resultado de las pruebas ADF
kbl(adf_tests, longtable = T, booktabs = T, caption = "Resultado de la prueba ADF",
    col.names = c("Sin diferenciar", "Diferenciada"), escape = TRUE, digits = 3)
```

Cuadro 1: Resultado de la prueba ADF

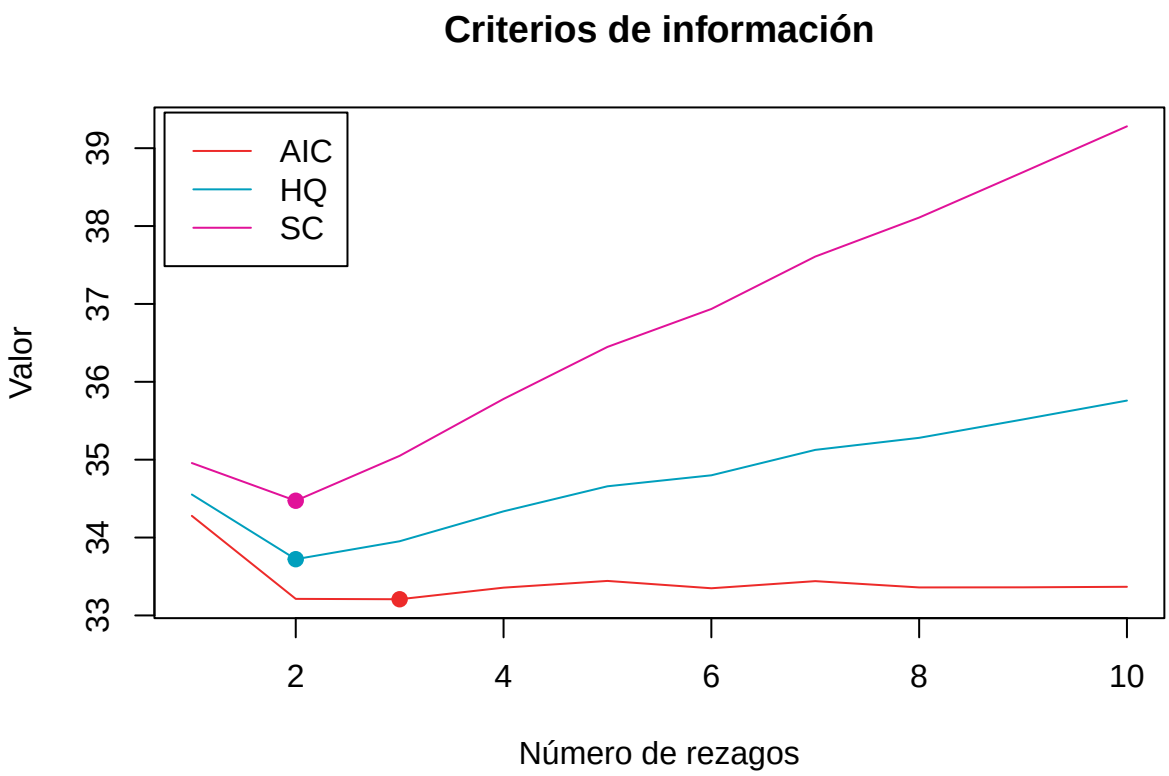
	Sin diferenciar	Diferenciada
INPC	0.990	0.01
IAI	0.087	0.01
IPI_EUA	0.249	0.01
M4	0.990	0.01
X	0.813	0.01
TOTAL_SERV	0.868	0.01

La tabla anterior contiene el resultado de los valores p de la prueba *ADF* tanto para la serie original como la serie diferenciada. En ella se puede observar que las series no son estacionarias en los datos originales, pero sí lo son en las primeras diferencias; es decir, las 6 series tienen una raíz unitaria, por lo que el ajuste de un modelo VEC sería una buena opción, ya que un elemento importante de estos modelos es que las variables deben tener orden de integración 1.

Siguiendo con el análisis, para la prueba de cointegración es indispensable especificar el número de rezagos a incluir. El orden del modelo VEC correspondiente es siempre uno menos que el VAR; el comando *vec* hace este ajuste automáticamente, por lo que siempre se referirá al orden del VAR subyacente. La salida siguiente utiliza la función *VARorder* para determinar el orden de rezago del modelo VAR con base a los criterios de información de *Akaike* (AIC), *Schwartz* (SC) y *Hannan - Quinn* (HQ).

6.3 Determinación del número de rezagos

```
# Determinar el número de rezagos
criterios<-VARselect(ioae_int, type = "const")

# Gráfico de los criterios
matplot(1:10, t(criterios$criteria[c(1, 2, 3), ]), type = "l",
        ylab = "Valor", col = c("#ED2B2A", "#009FBD", "#E11299"), lty = 1,
        main = "Criterios de información", xlab = "Número de rezagos")
# Leyenda del gráfico
legend("topleft", inset = 0.01, legend = c("AIC", "HQ", "SC"), lty = 1,
       col = c("#ED2B2A", "#009FBD", "#E11299"))
# Número de rezagos para AIC
points(criterios$selection[1], criterios$criteria[1, criterios$selection[1]],
       pch = 19, col = "#ED2B2A")
# Número de rezagos para HQ
points(criterios$selection[2], criterios$criteria[2, criterios$selection[2]],
       pch = 19, col = "#009FBD")
# Número de rezagos para SC
points(criterios$selection[3], criterios$criteria[3, criterios$selection[3]],
       pch = 19, col = "#E11299")
```



Los criterios se basan en la teoría de la información e indican la información relativa perdida cuando los datos se ajustan usando diferentes especificaciones. La longitud de rezago que produce el valor mínimo del estadístico de la información es la especificación elegida. De acuerdo con la salida anterior, los criterios de información de *Hannan - Quinn* (HQ) y *Schwartz* (SC) determinaron 2 rezagos, mientras que, Akaike (AIC) sugiere que sean 3 rezagos, por lo que se optó por seleccionar 2. Siguiendo con el análisis, ahora se realiza la prueba de cointegración de Johansen.

La prueba de *Soren Johansen* permite evidenciar si dos o más series de tiempo se mueven en la misma tendencia a lo largo del tiempo, teniendo a su vez estabilidad en las diferencias entre ellas. Así pues, este conjunto de series de tiempo no estacionarias de orden I(1), estarán cointegradas si existe una combinación lineal de esas series que sea estacionaria. La prueba de hipótesis según la metodología de *Johansen* se presenta a continuación.

6.4 Prueba de Johansen al 5%

```
# Prueba de Johansen
# K es el número de rezagos seleccionados
# Los parámetros ecdet y spec son sugerencias del Dr. Francisco, así lo vimos en clase
johansen<-ca.jo(ioae_int, ecdet = "const", spec = "transitory", K = 2)
# Se imprime el resultado de la prueba
summary(johansen)
```

```
#
# ######################
# # Johansen-Procedure #
```

```
# ######################
#
# Test type: maximal eigenvalue statistic (lambda max) , without linear trend and constant in cointegra
#
# Eigenvalues (lambda):
# [1] 3.721294e-01 2.916076e-01 1.657074e-01 5.882546e-02 2.796598e-02
# [6] 1.220616e-02 7.738406e-17
#
# Values of teststatistic and critical values of test:
#
#           test 10pct  5pct  1pct
# r <= 5 |  2.63  7.52  9.24 12.97
# r <= 4 |  6.07 13.75 15.67 20.20
# r <= 3 | 12.97 19.77 22.00 26.81
# r <= 2 | 38.77 25.56 28.14 33.24
# r <= 1 | 73.78 31.66 34.40 39.79
# r = 0  | 99.60 37.45 40.30 46.82
#
# Eigenvectors, normalised to first column:
# (These are the cointegration relations)
#
#                     INPC.l1        IAI.l1    IPI_EUA.l1         M4.l1
# INPC.l1        1.000000e+00  1.000000e+00  1.000000e+00  1.000000e+00
# IAI.l1         1.630215e+00  1.082636e+00  6.455498e-01  2.680712e-01
# IPI_EUA.l1    -2.651362e-02  2.777170e-01  4.329065e-01  4.419268e-02
# M4.l1         -4.278387e-03 -2.473316e-03 -2.039452e-03 -4.046543e-03
# X.l1           2.955382e-04 -1.049753e-03 -9.027924e-04 -1.133137e-03
# TOTAL_SERV.l1 -7.677521e-04 -7.915793e-04 -8.828291e-04  1.070142e-03
# constant      -1.390650e+02 -9.656478e+01 -6.024050e+01 -1.508544e+02
#                       X.l1 TOTAL_SERV.l1      constant
# INPC.l1        1.0000000000  1.000000e+00  1.000000e+00
# IAI.l1         1.5335591262 -1.461770e-01 -2.270309e-01
# IPI_EUA.l1    -3.4939650854  2.007909e-01  5.535766e-01
# M4.l1         -0.0048038641 -5.363085e-03 -3.655296e-03
# X.l1           0.0005896600  1.394659e-04 -1.774941e-04
# TOTAL_SERV.l1  0.0005947342 -9.625686e-05 -1.765391e-04
# constant      71.7528492949 -3.891924e+01 -6.354386e+01
#
# Weights W:
# (This is the loading matrix)
#
#                     INPC.l1       IAI.l1   IPI_EUA.l1         M4.l1
# INPC.d       -2.604368e-04  -0.02367927  -0.02270151  -0.002689801
# IAI.d        -1.106021e-01  -0.04464164   0.17689966  -0.095592384
# IPI_EUA.d    -3.637077e-02  -0.02447587   0.26894246  -0.059353670
# M4.d          4.088194e+00  -7.29533993   6.46365844   0.109725546
# X.d          -9.455484e+01 -51.61670319 378.27228238 -17.554999605
# TOTAL_SERV.d -1.806821e+01 -20.67038794 216.47895262 -49.330207438
#                       X.l1 TOTAL_SERV.l1      constant
# INPC.d       -4.935830e-04  -0.005539843  1.734017e-14
# IAI.d        -1.464352e-02  -0.002936750  3.070168e-14
# IPI_EUA.d     7.249236e-04  -0.010039181 -3.129537e-14
# M4.d          4.056754e-01   1.712254290  9.687123e-13
# X.d          -4.588617e+00  -8.889085248 -5.327928e-11
```

```
# TOTAL_SERV.d -1.171253e+01  -8.634473138 -1.509202e-11
```

La primera sección muestra los valores propios generados por la prueba. En este caso se tiene un valor propio de 0.3721294, el segundo valor propio es de 0.2916076, el tercero de 0.1657074 y así sucesivamente.

La siguiente sección muestra la estadística de prueba para las seis hipótesis nulas (H_o) de $r \leq 5$, $r \leq 4$, $r \leq 3$, $r \leq 2$, $r \leq 1$ y $r = 0$. Para cada una de estas pruebas se tiene no solo la estadística en sí (dada debajo de la columna) sino también los valores críticos en ciertos niveles de confianza: 10%, 5% y 1%, respectivamente.

La primera hipótesis, $H_o : r = 0$ vs $H_a : r > 0$ es la prueba para detectar la presencia de cointegración. Dado que la estadística de la prueba supera significativamente el nivel del 5% ($99.60 > 40.30$), se tiene suficiente evidencia para rechazar la hipótesis nula de la no cointegración. La segunda prueba para $H_o : r \leq 1$ vs $H_a : r > 1$ proporciona suficiente evidencia para rechazar H_o, ya que el estadístico de prueba supera el nivel del 5% ($73.78 > 34.40$). En el caso de la tercera prueba, para $H_o : r \leq 2$ vs $H_a : r > 2$, nuevamente, proporciona evidencia para rechazar H_o, ya que el estadístico de prueba supera el nivel del 5% ($38.77 > 28.14$). Finalmente, la cuarta prueba para $H_o : r \leq 3$ vs $H_a : r > 3$ no proporciona suficiente evidencia para rechazar H_o, ya que el estadístico de prueba no supera el nivel del 5% ($12.97 < 22.00$). Entonces se concluye que hay tres relaciones de cointegración.

Por lo tanto, la mejor estimación del rango de la matriz es $r = 3$, lo que indica que son necesarias hasta 3 combinaciones lineales de las series temporales para formar una serie estacionaria. Dado que la matriz de cointegración está normalizada en la primera columna, la relación es la siguiente:

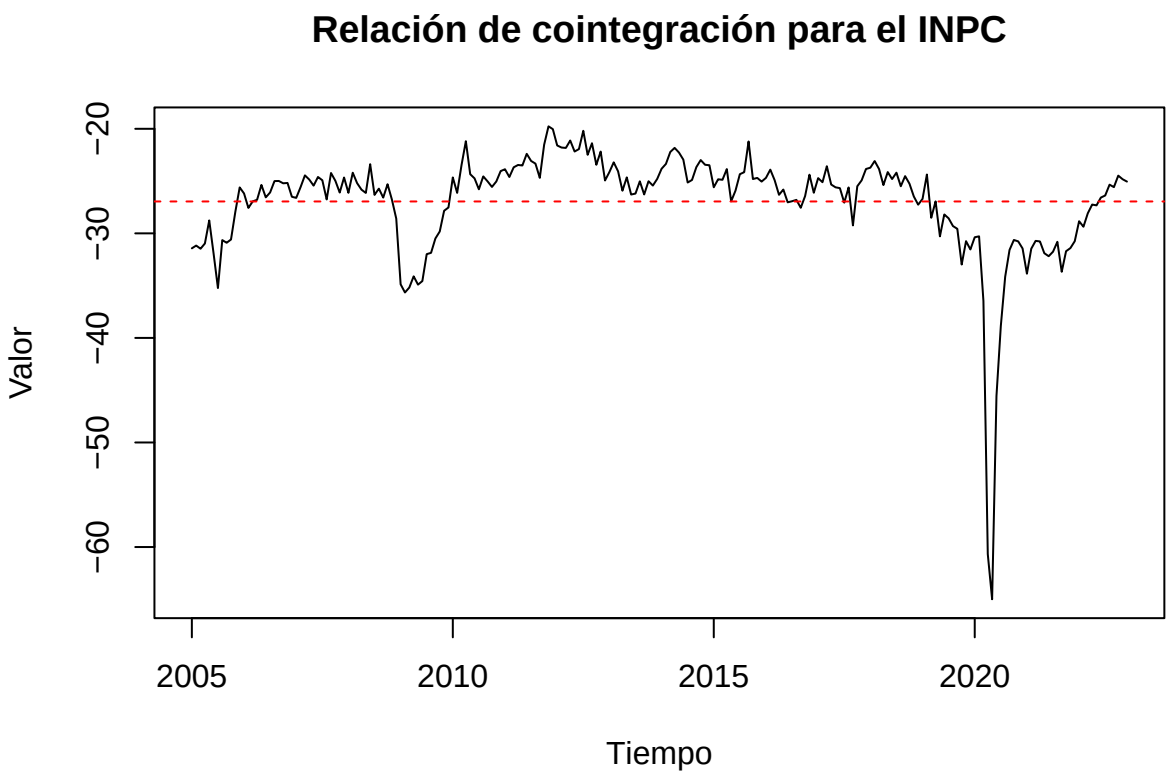
$$INPC + 1.630IAI - 0.027IPI\_EUA - 0.004M4 - 0.001TOTAL\_SERV - 139.065 = e_t \Longrightarrow$$

$$INPC = 139.065 - 1.630IAI + 0.027IPI\_EUA + 0.004M4 + 0.001TOTAL\_SERV + e_t$$

Con base a la anterior, hay una relación negativa entre el Índice Nacional de Precios al Consumidor (INPC) y el Índice de producción industrial (IAI), ya que, si el IAI aumenta el INPC disminuirá. Con respecto al Índice de producción industrial de los Estados Unidos (IPI_EUA), el INPC tiene una relación positiva, de tal forma que si IPI_EUA aumenta el INPC tenderá a aumentar. Por otro lado, el agregado monetario (M4) tiene una relación positiva con el INPC, de tal manera que, si M4 aumenta, el INPC aumentará. Con relación a las Exportaciones totales (X), el coeficiente es cero a 3 decimales, por lo que se podría descartar. En cuanto a la serie ($TOTAL\_SERV$) también tiene una relación positiva con el INPC, entonces, si TOTAL_SERV aumenta, también lo hará ($TOTAL\_SERV$).

La representación gráfica de la relación anterior es la siguiente:

```
# Determinación de la serie
et_1<-ioae_int[, "INPC"] + 1.63*ioae_int[, "IAI"] - 0.027*ioae_int[, "IPI_EUA"] -
  0.004*ioae_int[, "M4"] - 0.001*ioae_int[, "TOTAL_SERV"] - 139.065
# Gráfico temporal
plot(et_1, xlab = "Tiempo", ylab = "Valor",
     main = "Relación de cointegración para el INPC")
# Promedio del error
abline(h = mean(et_1), col = "red", lty = 2)
```



La gráfica muestra que, el equilibrio parece algo estable, ya que en teoría los residuales de la ecuación de cointegración deberían fluctuar alrededor de una media, que en este caso el promedio de los residuales es de -26.94887 (línea punteada). Asimismo, se observa que los residuales cuentan con un punto mínimo en 2020, muy posiblemente por el efecto de la pandemia por COVID 19. Ahora, es necesario revisar si los residuales son estacionarios, para ello, se aplicará la prueba aumentada de Dickey-Fuller (ADF) implementada por el Dr. Francisco de Jesús Corona.

```
# Prueba ADF
adf(et_1, "const")$p.value
```

```
# [1] 0.02967196
```

De acuerdo con la salida previa, ya que el valor p es de 0.02967196, se concluye que, los residuales de la ecuación de cointegración son estacionarios, por lo que el equilibrio podría ser estable.

En conclusión, se está en el caso $\mathbf{0 < Rango(\Pi) < K}$, cuando el rango es mayor a cero, pero menor al número de variables o series (K), lo que significa que hay relaciones de cointegración. En este caso, las variables que componen el vector Y_t están cointegradas, existiendo m vectores de cointegración linealmente independientes, que vienen dados por las combinaciones lineales $w = \beta^T Y_t$.

6.5 Estimación del modelo VAR/VEC

Para la estimación del **modelo VEC** se emplea la función *cajorls*. El resultado del modelo se presenta enseguida.

```r
# Ajuste del modelo VEC
modelo_vec_inpc<-cajorls(johansen, r = 3)
# Resumen del modelo
summary(modelo_vec_inpc$rlm)
```

```
# Response INPC.d :
#
# Call:
# lm(formula = INPC.d ~ ect1 + ect2 + ect3 + INPC.dl1 + IAI.dl1 +
#     IPI_EUA.dl1 + M4.dl1 + X.dl1 + TOTAL_SERV.dl1 - 1, data = data.mat)
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -1.23346 -0.14919  0.02938  0.14825  1.16546
#
# Coefficients:
#                  Estimate Std. Error t value Pr(>|t|)
# ect1           -4.664e-02  2.027e-02  -2.301   0.0224 *
# ect2           -4.072e-02  1.438e-02  -2.831   0.0051 **
# ect3           -1.640e-02  8.563e-03  -1.915   0.0569 .
# INPC.dl1        5.280e-01  6.476e-02   8.154 3.47e-14 ***
# IAI.dl1         3.403e-02  1.993e-02   1.707   0.0893 .
# IPI_EUA.dl1     5.473e-02  3.006e-02   1.821   0.0701 .
# M4.dl1         -3.120e-05  2.126e-04  -0.147   0.8835
# X.dl1          -5.020e-05  2.062e-05  -2.434   0.0158 *
# TOTAL_SERV.dl1 -7.619e-05  3.380e-05  -2.254   0.0253 *
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 0.2778 on 205 degrees of freedom
# Multiple R-squared:  0.6566,  Adjusted R-squared:  0.6415
# F-statistic: 43.55 on 9 and 205 DF,  p-value: < 2.2e-16
#
#
# Response IAI.d :
#
# Call:
# lm(formula = IAI.d ~ ect1 + ect2 + ect3 + INPC.dl1 + IAI.dl1 +
#     IPI_EUA.dl1 + M4.dl1 + X.dl1 + TOTAL_SERV.dl1 - 1, data = data.mat)
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -21.6341  -0.6266   0.1171   0.7995   5.4825
#
# Coefficients:
#                  Estimate Std. Error t value Pr(>|t|)
# ect1            0.0216560  0.1410808   0.154    0.878
# ect2           -0.1144382  0.1000782  -1.143    0.254
# ect3            0.0671157  0.0595884   1.126    0.261
# INPC.dl1        0.2021889  0.4506607   0.449    0.654
# IAI.dl1        -0.1683125  0.1387085  -1.213    0.226
# IPI_EUA.dl1     0.8859544  0.2091752   4.235 3.44e-05 ***
# M4.dl1         -0.0018011  0.0014794  -1.217    0.225
```

```
# X.dl1          -0.0001512  0.0001435  -1.054    0.293
# TOTAL_SERV.dl1  0.0001785  0.0002352   0.759    0.449
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 1.933 on 205 degrees of freedom
# Multiple R-squared:  0.2399,  Adjusted R-squared:  0.2065
# F-statistic: 7.188 on 9 and 205 DF,  p-value: 4.717e-09
#
#
# Response IPI_EUA.d :
#
# Call:
# lm(formula = IPI_EUA.d ~ ect1 + ect2 + ect3 + INPC.dl1 + IAI.dl1 +
#     IPI_EUA.dl1 + M4.dl1 + X.dl1 + TOTAL_SERV.dl1 - 1, data = data.mat)
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -11.2935  -0.3447   0.0312   0.4335   2.9695
#
# Coefficients:
#                  Estimate Std. Error t value Pr(>|t|)
# ect1            2.081e-01  8.535e-02   2.438  0.01561 *
# ect2            8.783e-02  6.054e-02   1.451  0.14841
# ect3            1.106e-01  3.605e-02   3.068  0.00245 **
# INPC.dl1       -1.109e-01  2.726e-01  -0.407  0.68458
# IAI.dl1         5.554e-02  8.391e-02   0.662  0.50876
# IPI_EUA.dl1     2.895e-01  1.265e-01   2.288  0.02317 *
# M4.dl1         -9.630e-04  8.950e-04  -1.076  0.28321
# X.dl1           3.285e-05  8.682e-05   0.378  0.70555
# TOTAL_SERV.dl1 -1.883e-04  1.423e-04  -1.323  0.18723
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 1.17 on 205 degrees of freedom
# Multiple R-squared:  0.1603,  Adjusted R-squared:  0.1235
# F-statistic: 4.349 on 9 and 205 DF,  p-value: 3.338e-05
#
#
# Response M4.d :
#
# Call:
# lm(formula = M4.d ~ ect1 + ect2 + ect3 + INPC.dl1 + IAI.dl1 +
#     IPI_EUA.dl1 + M4.dl1 + X.dl1 + TOTAL_SERV.dl1 - 1, data = data.mat)
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -337.63  -58.40    0.41   51.81  296.55
#
# Coefficients:
#                  Estimate Std. Error t value Pr(>|t|)
# ect1             3.256513   6.789712   0.480  0.63201
# ect2             2.939052   4.816406   0.610  0.54239
# ect3             0.663727   2.867776   0.231  0.81720
```

```
# INPC.dl1        37.927383  21.688681   1.749  0.08184 .
# IAI.dl1          8.926932   6.675541   1.337  0.18262
# IPI_EUA.dl1    -12.935879  10.066851  -1.285  0.20024
# M4.dl1          -0.196957   0.071200  -2.766  0.00619 **
# X.dl1           -0.001432   0.006907  -0.207  0.83601
# TOTAL_SERV.dl1  -0.013922   0.011321  -1.230  0.22022
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 93.05 on 205 degrees of freedom
# Multiple R-squared:  0.3898,  Adjusted R-squared:  0.363
# F-statistic: 14.55 on 9 and 205 DF,  p-value: < 2.2e-16
#
#
# Response X.d :
#
# Call:
# lm(formula = X.d ~ ect1 + ect2 + ect3 + INPC.dl1 + IAI.dl1 +
#     IPI_EUA.dl1 + M4.dl1 + X.dl1 + TOTAL_SERV.dl1 - 1, data = data.mat)
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -10928.5   -450.6    104.8    653.2   4275.3
#
# Coefficients:
#                 Estimate Std. Error t value Pr(>|t|)
# ect1           232.10074   93.40812   2.485 0.013763 *
# ect2            34.16684   66.26075   0.516 0.606660
# ect3           151.92868   39.45286   3.851 0.000157 ***
# INPC.dl1       161.47144  298.37772   0.541 0.588982
# IAI.dl1         95.37481   91.83743   1.039 0.300252
# IPI_EUA.dl1    547.02439  138.49270   3.950 0.000108 ***
# M4.dl1          -2.09501    0.97952  -2.139 0.033633 *
# X.dl1           -0.40247    0.09502  -4.236 3.44e-05 ***
# TOTAL_SERV.dl1   0.36166    0.15575   2.322 0.021213 *
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 1280 on 205 degrees of freedom
# Multiple R-squared:  0.4535,  Adjusted R-squared:  0.4295
# F-statistic:  18.9 on 9 and 205 DF,  p-value: < 2.2e-16
#
#
# Response TOTAL_SERV.d :
#
# Call:
# lm(formula = TOTAL_SERV.d ~ ect1 + ect2 + ect3 + INPC.dl1 + IAI.dl1 +
#     IPI_EUA.dl1 + M4.dl1 + X.dl1 + TOTAL_SERV.dl1 - 1, data = data.mat)
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -15243.3   -193.1     45.9    314.4   2020.7
#
# Coefficients:
```

```
#                 Estimate Std. Error t value Pr(>|t|)
# ect1           177.74035   85.43975   2.080 0.038741 *
# ect2            87.91438   60.60824   1.451 0.148438
# ect3            88.45368   36.08726   2.451 0.015079 *
# INPC.dl1       182.16472  272.92399   0.667 0.505232
# IAI.dl1         32.89847   84.00305   0.392 0.695735
# IPI_EUA.dl1    488.61900  126.67830   3.857 0.000154 ***
# M4.dl1          -1.15199    0.89596  -1.286 0.199976
# X.dl1           -0.09204    0.08692  -1.059 0.290864
# TOTAL_SERV.dl1  -0.33821    0.14247  -2.374 0.018520 *
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 1171 on 205 degrees of freedom
# Multiple R-squared:  0.1645,  Adjusted R-squared:  0.1278
# F-statistic: 4.484 on 9 and 205 DF,  p-value: 2.182e-05
```

Considerando un nivel de significancia del 0.05 y que, por el objetivo de este trabajo, el INPC es la variable de interés, entonces, el análisis se enfocará en esa parte.

Con base a la salida previa y considerando el INPC como la variable respuesta, se observa que los ect1, ect2 y ect3 (errores correction term) son negativos y significativos a un nivel de 0.01, 0.001 y 0.1 de manera respectiva, es decir, hay convergencia al estado de equilibrio. Por otro lado, el INPC es significativo y positivo, lo que implica que el propio INPC está relacionado con el dato anterior de manera positiva, ya que, si aumenta el dato anterior, seguramente el actual también lo hará; además, en el modelo se observa que X y $TOTAL\_SERV$ son significativos, pero con signo negativo, lo que implica que se relacionan en el corto plazo con el INPC, por lo que, un aumento en el dato anterior para estas variables ocasionará que el INPC actual disminuya.

Por otra parte, para la conversión de un modelo VEC a un **modelo VAR** en niveles se emplea la función *vec2var*. El resultado se presenta a continuación.

```r
# Ajuste del modelo VAR
modelo_var_inpc<-vec2var(johansen, r = 3)
# Resumen del modelo
modelo_var_inpc
```

```
#
# Coefficient matrix of lagged endogenous variables:
#
# A1:
#                 INPC.l1        IAI.l1   IPI_EUA.l1         M4.l1          X.l1
# INPC         1.48140219  -0.006688284   0.03833121  7.478163e-05 -0.0000049230
# IAI          0.22384482   0.717249307   0.95307017 -1.578292e-03 -0.0002967544
# IPI_EUA      0.09719083   0.143368795   1.40008582 -1.295320e-03 -0.0001950045
# M4          41.18389572  11.865983764 -12.27215212  7.904133e-01  0.0015995548
# X          393.57217951 129.541649654 698.95307349 -2.334276e+00  0.2822675118
# TOTAL_SERV 359.90506954 120.812850466 577.07267712 -1.465060e+00 -0.2711177991
#            TOTAL_SERV.l1
# INPC       -3.720921e-05
# IAI         1.426253e-04
# IPI_EUA    -3.784442e-04
# M4         -1.699221e-02
# X           1.411657e-01
```

```
# TOTAL_SERV  5.009057e-01
#
#
# A2:
#                 INPC.12       IAI.12    IPI_EUA.12        M4.12          X.12
# INPC         -0.5280434  -0.03402726   -0.05472807 3.119757e-05  5.019816e-05
# IAI          -0.2021889   0.16831252   -0.88595443 1.801125e-03  1.512263e-04
# IPI_EUA       0.1109050  -0.05554365   -0.28949193 9.629691e-04 -3.285007e-05
# M4          -37.9273828  -8.92693176   12.93587911 1.969572e-01  1.431624e-03
# X          -161.4714379 -95.37480793 -547.02439359 2.095014e+00  4.024714e-01
# TOTAL_SERV -182.1647197 -32.89847074 -488.61899885 1.151989e+00  9.204120e-02
#            TOTAL_SERV.12
# INPC        7.619474e-05
# IAI        -1.785451e-04
# IPI_EUA     1.883123e-04
# M4          1.392203e-02
# X          -3.616622e-01
# TOTAL_SERV  3.382145e-01
#
#
# Coefficient matrix of deterministic regressor(s).
#
#                constant
# INPC           3.690352
# IAI            9.035158
# IPI_EUA       -8.779822
# M4          -253.425726
# X          -4653.690741
# TOTAL_SERV -8532.113385
```

Sean $Y_{1t} = INPC$, $Y_{2t} = IAI$, $Y_{3t} = IPI\_EUA$, $Y_{4t} = M4$, $Y_{5t} = X$ y $Y_{6t} = TOTAL\_SERV$, la salida previa se puede representar de forma matricial de la siguiente manera.

$$\begin{bmatrix} Y_{1t} \\ Y_{2t} \\ Y_{3t} \\ Y_{4t} \\ Y_{5t} \\ Y_{6t} \end{bmatrix} = \begin{bmatrix} 3.7 \\ 9 \\ -8.8 \\ -253.4 \\ -4653.7 \\ -8532.1 \end{bmatrix} + \begin{bmatrix} 1.5 & 0 & 0 & 0 & 0 & 0 \\ 0.2 & 0.7 & 1 & 0 & 0 & 0 \\ 0.1 & 0.1 & 1.4 & 0 & 0 & 0 \\ 41.2 & 11.9 & -12.3 & 0.8 & 0 & 0 \\ 393.6 & 129.5 & 699 & -2.3 & 0.3 & 0.1 \\ 359.9 & 120.8 & 577.1 & -1.5 & -0.3 & 0.5 \end{bmatrix} \begin{bmatrix} Y_{1t-1} \\ Y_{2t-1} \\ Y_{3t-1} \\ Y_{4t-1} \\ Y_{5t-1} \\ Y_{6t-1} \end{bmatrix} +$$

$$\begin{bmatrix} -0.5 & 0 & -0.1 & 0 & 0 & 0 \\ -0.2 & 0.2 & -0.9 & 0 & 0 & 0 \\ 0.1 & -0.1 & -0.3 & 0 & 0 & 0 \\ -37.9 & -8.9 & 12.9 & 0.2 & 0 & 0 \\ -161.5 & -95.4 & -547 & 2.1 & 0.4 & -0.4 \\ -182.2 & -32.9 & -488.6 & 1.2 & 0.1 & 0.3 \end{bmatrix} \begin{bmatrix} Y_{1t-2} \\ Y_{2t-2} \\ Y_{3t-2} \\ Y_{4t-2} \\ Y_{5t-2} \\ Y_{6t-2} \end{bmatrix} + \begin{bmatrix} \epsilon_{1t} \\ \epsilon_{2t} \\ \epsilon_{3t} \\ \epsilon_{4t} \\ \epsilon_{5t} \\ \epsilon_{6t} \end{bmatrix}$$

6.6 Pruebas a los residuos

Continuando con el análisis, siguen las pruebas a los residuales. La regla de decisión para todas las pruebas es rechazar la hipótesis nula (H_o) si el valor p es menor que un nivel de significancia de 0.05 ($\alpha = 0.05$). Uno de los supuestos que deben cumplir los residuos es la no autocorrelación. En caso de existir errores autocorrelacionados se generarían intervalos de confianza inválidos para interpretar los t-estadísticos

asociados al modelo de regresión. En este caso, se puede utilizar la prueba *Portmanteau* para verificar si los residuales del modelo están libres de correlación serial. La hipótesis nula de esta prueba denota la no presencia de errores autocorrelacionados, mientras que, la hipótesis alternativa denota la presencia de errores autocorrelacionados. El código implementado en R para la generación de la prueba se muestra a continuación.

```
# Prueba para autocorrelación
serial.test(modelo_var_inpc)
```

```
#
#   Portmanteau Test (asymptotic)
#
# data:  Residuals of VAR object modelo_var_inpc
# Chi-squared = 709.36, df = 510, p-value = 1.099e-08
```

Como se muestra en la salida del modelo, el valor p es de 1.099e-08, por lo que se rechaza H_o, ya que (1.099e-08 < 0.05), es decir, existe suficiente evidencia para determinar la presencia de errores autocorrelacionados.

Por otra parte, el equivalente a errores heteroscedásticos en el caso de corte transversal está relacionado al efecto ARCH en series de tiempo, que son varianzas cambiantes y dependientes a través del tiempo. La hipótesis nula de esta prueba indica la no presencia del efecto ARCH. El resultado de la prueba es el siguiente:

```
# Prueba de ARCH
arch.test(modelo_var_inpc)
```

```
#
#   ARCH (multivariate)
#
# data:  Residuals of VAR object modelo_var_inpc
# Chi-squared = 2777.1, df = 2205, p-value = 7.772e-16
```

Como se muestra en la salida anterior, dado que el valor p es de 7.772e-16, se rechaza H_o (7.772e-16 < 0.05), es decir, existe suficiente evidencia para determinar la presencia de efecto ARCH, por lo que los errores presentan varianzas cambiantes y dependientes a través del tiempo.

Otro requisito deseable es la normalidad de la distribución de los residuos. Para revisarlo se emplea la prueba de *Jarque-Bera* a través del comando *normality.test*. La hipótesis nula de esta prueba indica que los errores se distribuyen de manera normal. El resultado es el siguiente:

```
# Prueba de Jarque-Bera para normalidad
normality.test(modelo_var_inpc, multivariate.only = TRUE)$jb.mul$JB
```

```
#
#   JB-Test (multivariate)
#
# data:  Residuals of VAR object modelo_var_inpc
# Chi-squared = 30964, df = 12, p-value < 2.2e-16
```

De acuerdo con la salida anterior, el valor p es $<$ 2.2e-16, por lo que se rechaza H_o, es decir, existe suficiente evidencia para determinar la no distribución normal de los errores.

Con base en lo anterior, se observa que ninguno de los supuestos del modelo se cumplió, ya que las pruebas realizadas arrojan problemas de autocorrelación, efecto ARCH y la ausencia de normalidad.

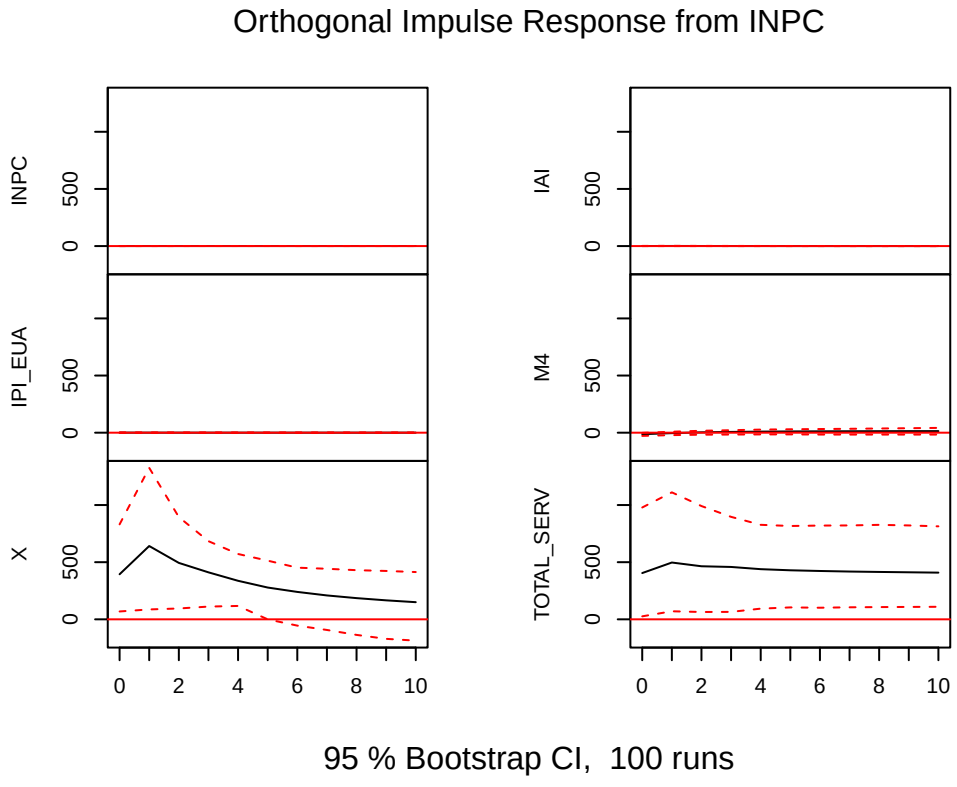
Entonces, aunque hay relaciones de cointegración, hay que tener cuidado con las interpretaciones de los coeficientes, dado que los errores no están siendo bien estimados. Adicionalmente, se aplicó la transformación logarítmica, raíz cuadrada e inversa a las series, pero ninguna de ellas mejoró los resultados de las pruebas; también se incrementó el número de rezagos del modelo hasta 5, pero, la situación fue bastante similar, es decir, ninguno de los supuestos del modelo se cumplió.

Una de las sugerencias del Dr. Francisco de Jesús Corona es analizar los resultados con los errores robustos de White, ya que es muy complicado poder corregir los problemas en los residuos. Este es un tema que requiere un análisis más detallado; sin embargo, al no cumplirse los supuestos del modelo, las interpretaciones realizadas no son válidas.

6.7 Gráfico de impulso-respuesta

Adicional a los análisis presentados con anterioridad, se generaron los gráficos de impulso-respuesta, los cuales se presentan a continuación.

```
# Gráficos de impulso respuesta
impresp<-irf(modelo_var_inpc, impulse = "INPC",
             response = c("INPC","IAI", "IPI_EUA", "M4", "X", "TOTAL_SERV"))
# Se genera el gráfico
plot(impresp)
```



Con base en la gráfica anterior, no se observa ningún impulso (choque) de la variable INPC sobre ella misma; de hecho, se nota que es estable a largo plazo y no se puede visualizar algo relevante. Para las variables IPI_EUA, IAI y M4 también permanecen sin efecto ante un impulso en el momento 0 por parte del INPC. En cambio, un impulso en el momento 0 para las variables X y TOTAL_SERV ocasiona que tengan

un aumento significativo hasta el periodo 1, después comienzan a bajar paulatinamente a largo plazo en el caso de la serie X, mientras que, para TOTAL_SERV se estabiliza a partir del periodo 2 aproximadamente.

En resumen, considerando a la variable del INPC como respuesta, los criterios para ver el número de rezagos sugirieron que el modelo incluyera 2 rezagos. La prueba de cointegración de *Johansen* indicó que existen al menos tres ecuaciones de cointegración. Los *ect1, ect2 y ect3* resultaron significativos y de signo negativo. Al realizar las pruebas a los residuales no se cumplieron los supuestos del modelo (ausencia de errores autocorrelacionados, no efecto ARCH y normalidad); no obstante, se aplicó la transformación logarítmica, raíz e inversa, también se incrementó el número de rezagos hasta 5, pero los resultados no tuvieron ningún cambio significativo. Por tal motivo, las interpretaciones realizadas al modelo no son válidas.

6.8 Pronósticos para el INPC

Dadas las conclusiones anteriores, no tiene demasiado sentido generar pronósticos para la variable INPC con el modelo VAR/VEC ajustado, ya que los supuestos del modelo no se cumplieron. Solo como ejercicio ilustrativo se hará el pronóstico para la variable del INPC con el modelo VAR/VEC ajustado.

Mediante el modelo propuesto se generaron 12 pronósticos que corresponden al año 2022. Para ello se ajustó un modelo VAR (con el comando *vec2var*) para realizar pronósticos un paso hacia adelante con un horizonte 12 de longitud. Es importante mencionar que, como criterio de información para la identificación del número de rezagos fue seleccionado el criterio de Hannan - Quinn (HQ). Asimismo, en cada paso hacia adelante se actualiza el pronóstico. El código implementado se presenta a continuación.

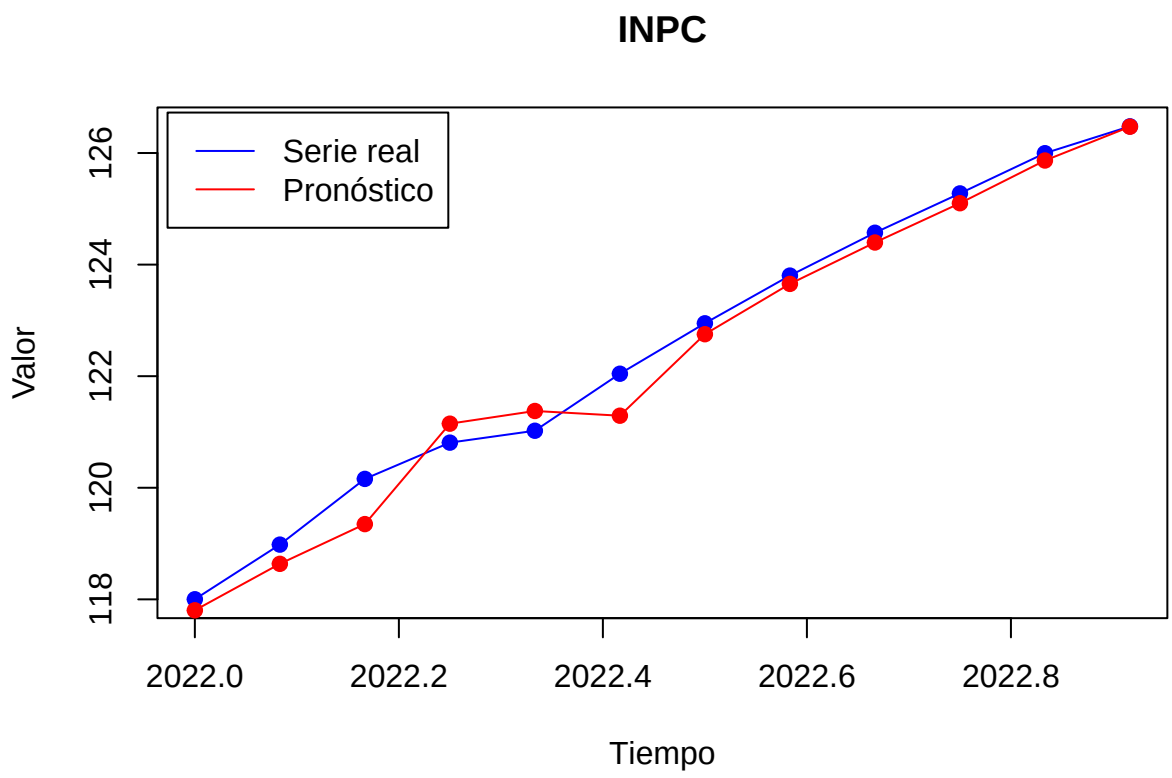
```
# Tamaño de la serie
n<-nrow(ioae_int)
# Número de pronósticos
H<-12
# Se extraen los datos que serán los observados
observado<-ioae_int[(n - H + 1):n, "INPC"]

# Tabla resumen de la salida de R
resumen<-data.frame(OBSERVADO = observado, PRONOSTICO = rep(0, H), ERROR = rep(0, H))
# Se crea el ciclo for para los pronósticos
for(h in 1:H){
# Se extraen los datos de interés de la serie
serie<-ts(ioae_int[1:(n - H - 1 + h), ], start = c(2005, 1), frequency = 12)
# Determinar el número de rezagos
p<-VARselect(serie)$selection["HQ(n)"]
# Prueba de Johansen
johansen<-ca.jo(serie, ecdet = "const", spec = "transitory", K = p)
# Ajuste del modelo VAR
modelo<-vec2var(johansen, r = 3)
# Se realiza el pronóstico
pronostico<-predict(modelo, n.ahead = 1)
# Se guarda el resultado del pronóstico en la tabla resumen
resumen[h, "PRONOSTICO"]<-pronostico$fcst$INPC[, "fcst"]
} # Del ciclo for

# Se calcula el error
resumen[, "ERROR"]<-resumen$OBSERVADO - resumen$PRONOSTICO
resumen
```

```
#    OBSERVADO PRONOSTICO        ERROR
# 1    118.002   117.8042  0.197813979
```

```
# 2    118.981   118.6364  0.344627851
# 3    120.159   119.3489  0.810108082
# 4    120.809   121.1489 -0.339932659
# 5    121.022   121.3761 -0.354079442
# 6    122.044   121.2915  0.752451715
# 7    122.948   122.7522  0.195785892
# 8    123.803   123.6541  0.148866778
# 9    124.571   124.3962  0.174771674
# 10   125.276   125.1004  0.175612799
# 11   125.997   125.8659  0.131078842
# 12   126.478   126.4714  0.006635125
```

Para apreciar de mejor manera los resultados obtenidos, a continuación, se presenta la siguiente gráfica temporal.

```r
# Transformar los pronósticos a serie de tiempo
observado_ts<-ts(resumen$OBSERVADO, start = c(2022, 1), frequency = 12)
pronostico_ts<-ts(resumen$PRONOSTICO, start = c(2022, 1), frequency = 12)
# Gráfico temporal
plot(observado_ts, xlab = "Tiempo", ylab = "Valor", type = "o", pch = 19,
col = "blue", main = "INPC")
# Datos pronosticados
points(pronostico_ts, col = "red", type = "o", pch = 19)
# Leyenda del gráfico
legend("topleft", inset = 0.01, legend = c("Serie real", "Pronóstico"),
       lty = c(1, 1), col = c("blue", "red"))
```



La salida anterior muestra la representación gráfica de los datos observados (línea azul) vs los datos pronosticados (línea roja) para un horizonte H de longitud 12. Como puede apreciarse, a pesar de que los valores pronosticados son bastante similares a los observados, se sugiere un uso cauteloso de estos valores.

Finalmente, para fines comparativos de los modelos ajustados, se calculó el error de predicción a través del error cuadrático medio (MSE por sus siglas en inglés).

```
# Se realizan los cálculos del MSE
MSE_VAR_VEC<-sum(resumen$ERROR^2)/H
MSE_VAR_VEC
```

```
# [1] 0.146699
```

De acuerdo con la salida previa, el valor del *MSE* para el Modelo VAR/VEC ajustado para generar pronósticos del INPC, fue de 0.146699

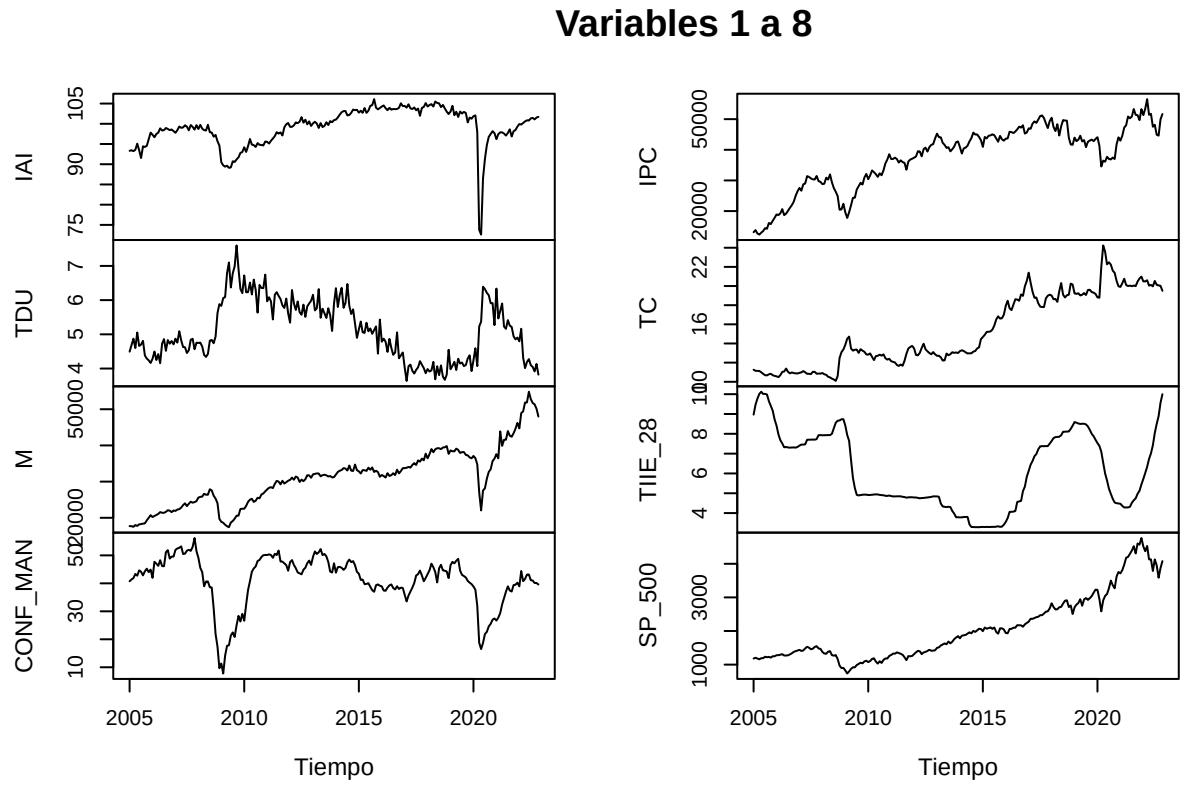
7. Modelo de Factores Dinámicos (MFD)

A continuación, se muestra el desarrollo para el ajuste del Modelo de Factores Dinámicos (MFD), en el que se utilizarán las 28 series con las que cuenta la tabla de datos. El objetivo es aprovechar las series para ajustar un MFD y realizar pronósticos a la variable del INPC.

7.1 Preparación de las series

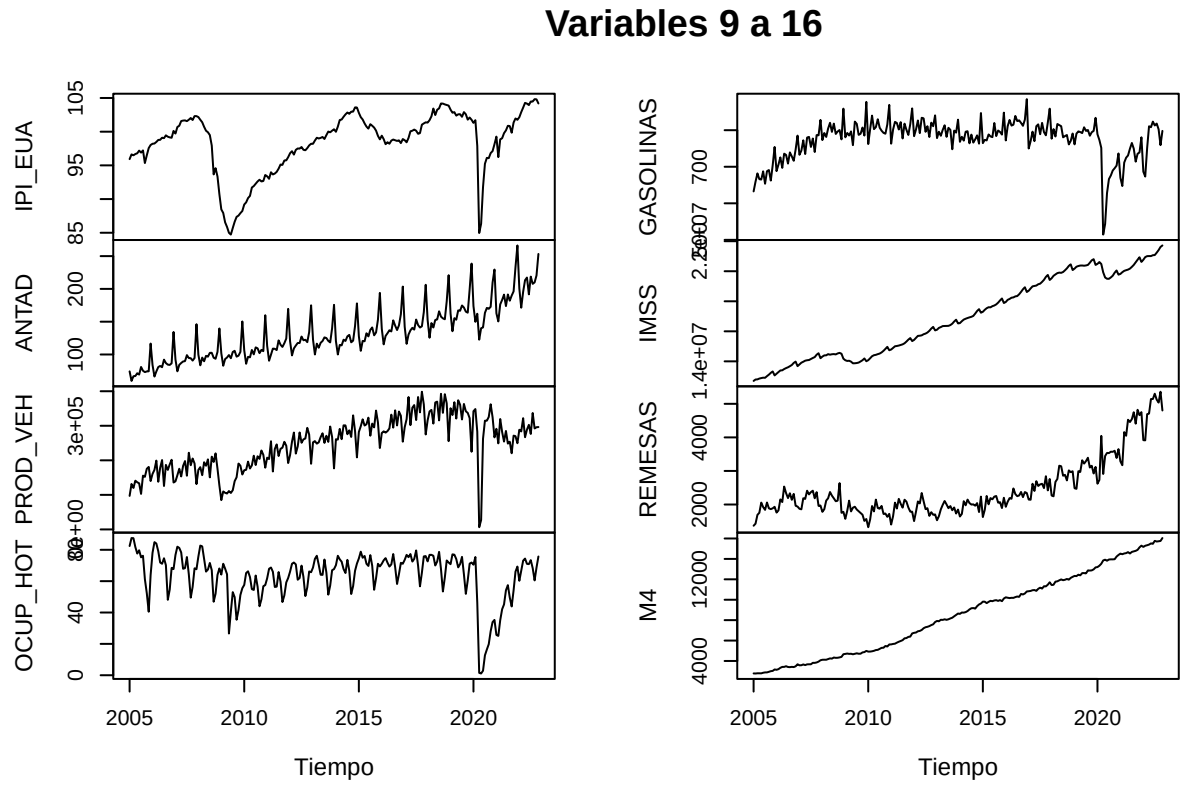
El ajuste del MFD comienza con el análisis visual de las series de tiempo, con el propósito de identificar patrones comunes y analizar posibles indicios de estacionalidad. Es importante mencionar que, la serie del IGAE cuenta con un dato menos que el resto de las variables, por lo que se excluyó el último valor de la tabla de datos.

```
################################
# Modelos de factores dinámicos
################################

# Se limpia el espacio de trabajo
rm(list = ls())
invisible(gc(reset = TRUE))

# Se cargan las librerías
library(astsa)      # Gráfico de retrasos
library(tseries)    # Prueba de ADF
library(forecast)   # Pronósticos
library(seasonal)   # Estacionalidad
library(vars)       # Modelos VAR/VEC
library(dlm)        # Análisis de modelos dinámicos lineales
library(kableExtra) # Edición de tablas

# Establecer dirección de trabajo
ruta<-("C:/Proyecto de series/")
# Se cargan las funciones
source(paste(ruta, "functions.r", sep = ""))

# Se leen los datos
datos<-read.csv(file = paste0(ruta, 'IOAE.csv'))

# Se convierten a series de tiempo
tr<-ts(datos[-nrow(datos), -1], start = c(2005, 1), frequency = 12)
# Número de series
N<-ncol(tr)
# Gráfico temporal de las series originales
plot(tr[, 1:8], main = "Variables 1 a 8",
     xlab = "Tiempo", cex.lab = 0.75)
```



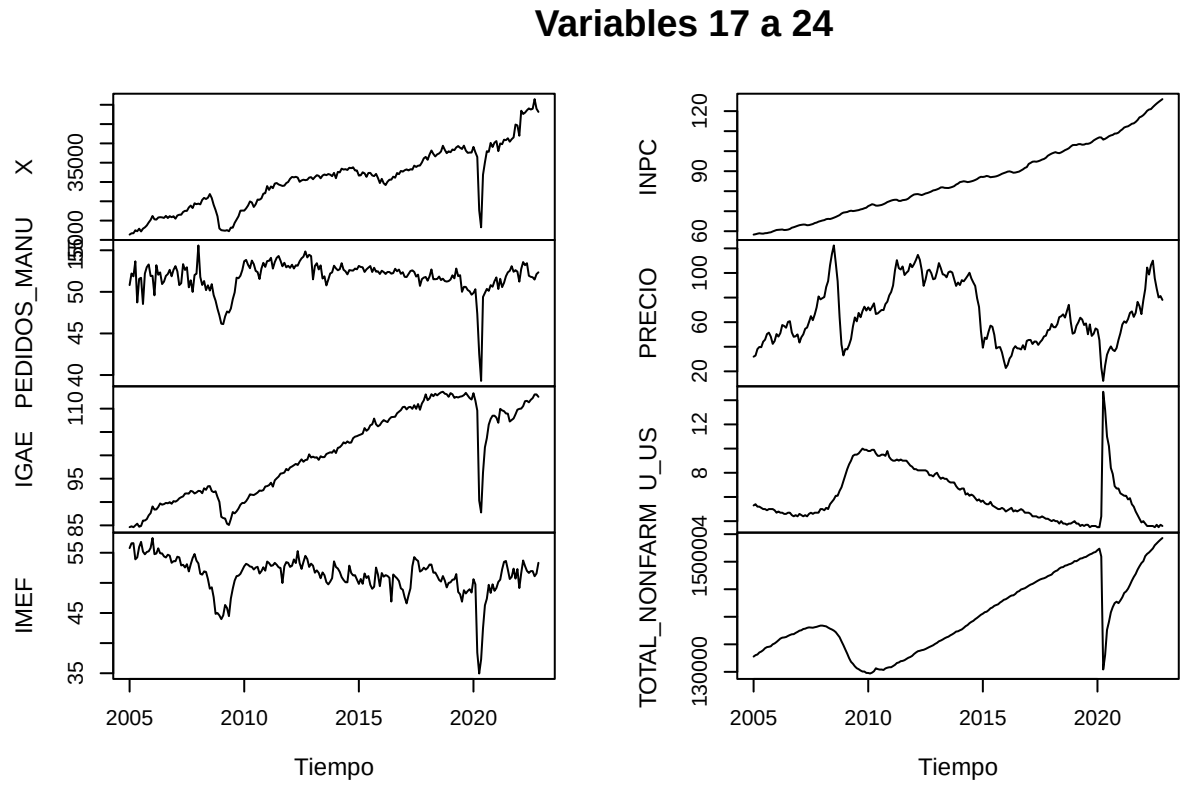
De acuerdo con la salida previa, para la serie de IAI existe un valor atípico cerca del año 2020, podría ser debido a la pandemia por COVID-19. Las series IPC, TC, M y SP_500 cuentan con una tendencia al alza muy pronunciada, en cambio, las series TDU, CONF_MAN y TIIE_28 no tienen una tendencia al alza como las anteriores. Por otra parte, a simple vista ninguna de las series cuenta con un patrón que dé indicios de estacionalidad. De este grupo, las series IPC, TC, M y SP_500 son las que cuentan con un comportamiento bastante similar.

```
# Gráfico temporal de las series originales
plot(tr[, 9:16], main = "Variables 9 a 16",
     xlab = "Tiempo", cex.lab = 0.75)
```



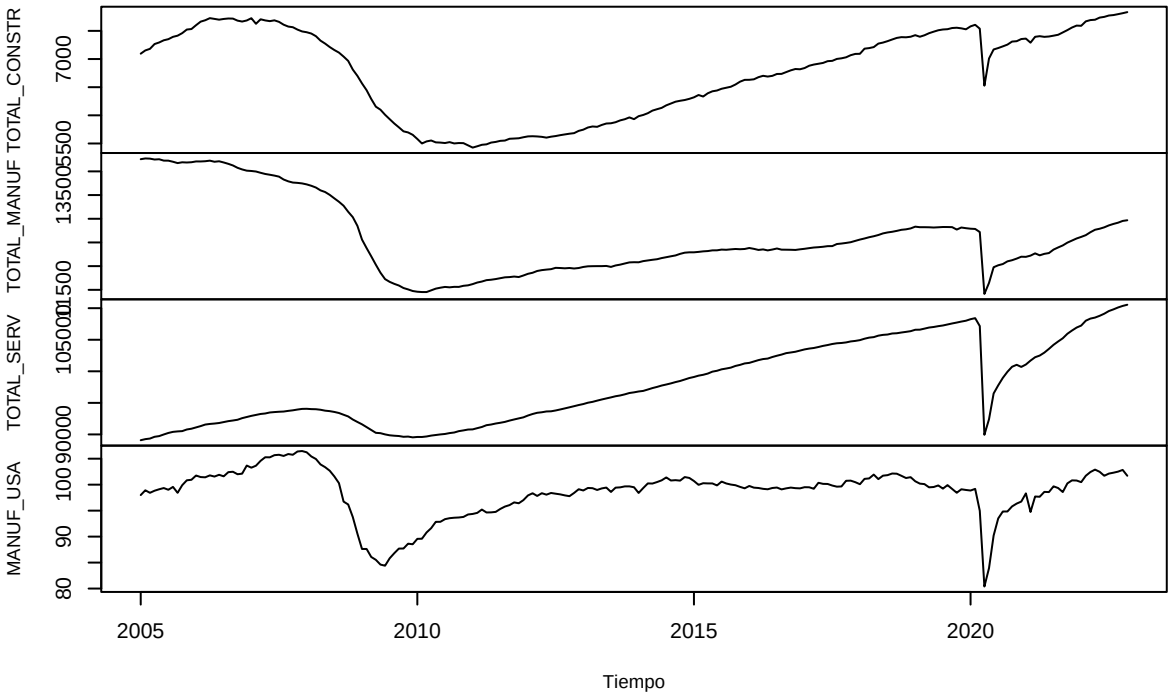
Con base a la anterior, para las series IPI_EUA, PROD_VEH, OCUP_HOT y GASOLINAS existe un dato atípico cerca de 2020, posiblemente esté relacionada con la pandemia por COVID-19. Las series M4 e IMMS tienen una tendencia muy pronunciada; esta última parece tener un patrón, lo que daría indicios de estacionalidad. Por otra parte, las series de ANTAD y REMESAS también cuentan con una tendencia al alza, además, se detectan indicios de estacionalidad, ya que cuentan con un patrón que se repite. En resumen, las series con posibles problemas de estacionalidad son: ANTAD, PROD_VEH, OCUP_HOT, GASOLINAS, IMSS y REMESAS. De este grupo, las parejas de series IMMS y M4, ANTAD y REMESAS, además de PROD_VEH y GASOLINAS tienen un comportamiento similar entre ellas.

```
# Gráfico temporal de las series originales
plot(tr[, 17:24], main = "Variables 17 a 24",
     xlab = "Tiempo", cex.lab = 0.75)
```



Derivada de la salida previa, en las series X, PEDIDOS_MANU, IGAE, IMEF, TOTAL_NONFARM y U_US existe un dato atípico cerca de 2020, posiblemente esté relacionado con la pandemia por COVID-19. Por otra parte, la serie del INPC tiene una tendencia al alza. Otro aspecto relevante es que las series TOTAL_NONFARM y U_US tienen el mismo comportamiento, pero a la inversa, como si se tratara de un espejo. A simple vista, este grupo de series parece no tener problemas de estacionalidad, ya que no se distingue un patrón. Finalmente, las parejas de series X e IGAE, además de PEDIDOS_MANU e IMEF, tienen un comportamiento similar entre ellas.

```
# Gráfico temporal de las series originales
plot(tr[, 25:28], main = "Variables 25 a 28",
     xlab = "Tiempo", cex.lab = 0.6)
```



De acuerdo con la gráfica anterior, este grupo de series también presenta un dato atípico cerca de 2020, posiblemente esté relacionado con la pandemia por COVID-19. Otro aspecto importante es que, a simple vista no se distinguen indicios de estacionalidad, ya que no se observa un patrón que se repita en las series. De este grupo, las series TOTAL_CONSTR y TOTAL_MANUF tienen un comportamiento bastante similar.

Antes de aplicar algún tipo de transformación a las series, ya que en la tabla de datos existen series desestacionalizadas que se descargaron directamente de la página de INEGI, además de otras que provienen principalmente de la página del Banco de México, cuyas series parecen no estar desestacionalizadas, la siguiente parte del análisis consiste en analizar formalmente los problemas de estacionalidad. El método visto en clase consiste en incluir variables ficticias estacionales y comprobar si tienen valores p significativos al calcular la regresión. Si los meses individuales tienen coeficientes significativos, la serie de tiempo tiene problemas de estacionalidad. El código implementado se muestra enseguida.

Nota importante: Para no hacer tan extenso el documento, solo se imprimirán los modelos de regresión en los que se detectaron problemas de estacionalidad.

```
# Modelo para determinar estacionalidad
for(i in 1:N){
estacionalidad<-lm(tr[, i] ~ c(1:nrow(tr)) + seasonaldummy(tr[, i]))
# Resumen del modelo
print(colnames(tr)[i])
print(summary(estacionalidad)) }
```

```
# [1] "ANTAD"
#
# Call:
# lm(formula = tr[, var_des[i]] ~ c(1:nrow(tr)) + seasonaldummy(tr[,
```

```
#     var_des[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -41.938  -6.588  -0.757   4.495  47.345
#
# Coefficients:
#                                    Estimate Std. Error t value Pr(>|t|)
# (Intercept)                       121.39884    2.92747   41.47   <2e-16 ***
# c(1:nrow(tr))                       0.56683    0.01193   47.53   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Jan -52.20427    3.66638  -14.24   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Feb -74.14823    3.66620  -20.23   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Mar -60.54551    3.66607  -16.52   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Apr -61.21105    3.66597  -16.70   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])May -54.70203    3.66591  -14.92   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Jun -59.08460    3.66589  -16.12   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Jul -47.13001    3.66591  -12.86   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Aug -51.19342    3.66597  -13.96   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Sep -55.62578    3.66607  -15.17   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Oct -54.27694    3.66620  -14.80   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Nov -37.33533    3.66638  -10.18   <2e-16 ***
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 10.84 on 202 degrees of freedom
# Multiple R-squared:  0.9335,  Adjusted R-squared:  0.9295
# F-statistic: 236.3 on 12 and 202 DF,  p-value: < 2.2e-16
#
# [1] "PROD_VEH"
#
# Call:
# lm(formula = tr[, var_des[i]] ~ c(1:nrow(tr)) + seasonaldummy(tr[,
#     var_des[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -280771  -20100    6243   30526   87232
#
# Coefficients:
#                                    Estimate Std. Error t value Pr(>|t|)
# (Intercept)                       103167.25   13319.35   7.746 4.52e-13 ***
# c(1:nrow(tr))                        896.32      54.26  16.520  < 2e-16 ***
# seasonaldummy(tr[, var_des[i]])Jan  38254.16   16681.20   2.293 0.022863 *
# seasonaldummy(tr[, var_des[i]])Feb  45369.40   16680.41   2.720 0.007099 **
# seasonaldummy(tr[, var_des[i]])Mar  61039.02   16679.79   3.659 0.000323 ***
# seasonaldummy(tr[, var_des[i]])Apr  17203.69   16679.35   1.031 0.303569
# seasonaldummy(tr[, var_des[i]])May  38756.54   16679.08   2.324 0.021138 *
# seasonaldummy(tr[, var_des[i]])Jun  64248.55   16679.00   3.852 0.000157 ***
# seasonaldummy(tr[, var_des[i]])Jul  31319.62   16679.08   1.878 0.061853 .
# seasonaldummy(tr[, var_des[i]])Aug  68979.46   16679.35   4.136 5.19e-05 ***
# seasonaldummy(tr[, var_des[i]])Sep  49217.19   16679.79   2.951 0.003545 **
# seasonaldummy(tr[, var_des[i]])Oct  81208.54   16680.41   4.868 2.26e-06 ***
# seasonaldummy(tr[, var_des[i]])Nov  62026.21   16681.20   3.718 0.000260 ***
# ---
```

```
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 49320 on 202 degrees of freedom
# Multiple R-squared:  0.6133,  Adjusted R-squared:  0.5903
# F-statistic:  26.7 on 12 and 202 DF,  p-value: < 2.2e-16
#
# [1] "OCUP_HOT"
#
# Call:
# lm(formula = tr[, var_des[i]] ~ c(1:nrow(tr)) + seasonaldummy(tr[,
#     var_des[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -61.998  -4.616   3.338   8.095  18.683
#
# Coefficients:
#                                    Estimate Std. Error t value Pr(>|t|)
# (Intercept)                        70.82693    3.67693  19.263  < 2e-16 ***
# c(1:nrow(tr))                      -0.05024    0.01498  -3.354 0.000950 ***
# seasonaldummy(tr[, var_des[i]])Jan   2.15541    4.60500   0.468 0.640247
# seasonaldummy(tr[, var_des[i]])Feb   7.32899    4.60478   1.592 0.113037
# seasonaldummy(tr[, var_des[i]])Mar   6.26701    4.60461   1.361 0.175020
# seasonaldummy(tr[, var_des[i]])Apr   1.91615    4.60449   0.416 0.677744
# seasonaldummy(tr[, var_des[i]])May  -5.28472    4.60441  -1.148 0.252429
# seasonaldummy(tr[, var_des[i]])Jun  -3.44781    4.60439  -0.749 0.454843
# seasonaldummy(tr[, var_des[i]])Jul   2.85854    4.60441   0.621 0.535413
# seasonaldummy(tr[, var_des[i]])Aug  -3.43954    4.60449  -0.747 0.455933
# seasonaldummy(tr[, var_des[i]])Sep -16.11430    4.60461  -3.500 0.000573 ***
# seasonaldummy(tr[, var_des[i]])Oct -10.09795    4.60478  -2.193 0.029454 *
# seasonaldummy(tr[, var_des[i]])Nov  -1.43215    4.60500  -0.311 0.756123
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 13.61 on 202 degrees of freedom
# Multiple R-squared:  0.2312,  Adjusted R-squared:  0.1856
# F-statistic: 5.063 on 12 and 202 DF,  p-value: 2.322e-07
#
# [1] "GASOLINAS"
#
# Call:
# lm(formula = tr[, var_des[i]] ~ c(1:nrow(tr)) + seasonaldummy(tr[,
#     var_des[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -251.85  -18.57   16.10   33.05   64.60
#
# Coefficients:
#                                    Estimate Std. Error t value Pr(>|t|)
# (Intercept)                       824.68297   13.90976  59.288  < 2e-16 ***
# c(1:nrow(tr))                       0.03461    0.05666   0.611 0.542042
# seasonaldummy(tr[, var_des[i]])Jan -82.02072   17.42064  -4.708 4.64e-06 ***
# seasonaldummy(tr[, var_des[i]])Feb -66.07184   17.41981  -3.793 0.000197 ***
```

```
# seasonaldummy(tr[, var_des[i]])Mar -51.52403   17.41916  -2.958 0.003467 **
# seasonaldummy(tr[, var_des[i]])Apr -64.83407   17.41870  -3.722 0.000256 ***
# seasonaldummy(tr[, var_des[i]])May -55.47963   17.41842  -3.185 0.001676 **
# seasonaldummy(tr[, var_des[i]])Jun -44.72709   17.41833  -2.568 0.010956 *
# seasonaldummy(tr[, var_des[i]])Jul -51.64917   17.41842  -2.965 0.003389 **
# seasonaldummy(tr[, var_des[i]])Aug -48.75879   17.41870  -2.799 0.005619 **
# seasonaldummy(tr[, var_des[i]])Sep -59.85653   17.41916  -3.436 0.000716 ***
# seasonaldummy(tr[, var_des[i]])Oct -56.59955   17.41981  -3.249 0.001356 **
# seasonaldummy(tr[, var_des[i]])Nov -48.30696   17.42064  -2.773 0.006075 **
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 51.5 on 202 degrees of freedom
# Multiple R-squared:  0.1199,  Adjusted R-squared:  0.06758
# F-statistic: 2.292 on 12 and 202 DF,  p-value: 0.009361
#
# [1] "IMSS"
#
# Call:
# lm(formula = tr[, var_des[i]] ~ c(1:nrow(tr)) + seasonaldummy(tr[,
#     var_des[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -896531 -372020   -1740  423699  824249
#
# Coefficients:
#                                   Estimate Std. Error t value Pr(>|t|)
# (Intercept)                       1.225e+07  1.317e+05  93.048   <2e-16 ***
# c(1:nrow(tr))                     4.224e+04  5.365e+02  78.735   <2e-16 ***
# seasonaldummy(tr[, var_des[i]])Jan 4.759e+04  1.649e+05   0.289   0.7732
# seasonaldummy(tr[, var_des[i]])Feb 1.223e+05  1.649e+05   0.742   0.4591
# seasonaldummy(tr[, var_des[i]])Mar 1.472e+05  1.649e+05   0.892   0.3732
# seasonaldummy(tr[, var_des[i]])Apr 1.138e+05  1.649e+05   0.690   0.4908
# seasonaldummy(tr[, var_des[i]])May 6.599e+04  1.649e+05   0.400   0.6895
# seasonaldummy(tr[, var_des[i]])Jun 5.681e+04  1.649e+05   0.344   0.7308
# seasonaldummy(tr[, var_des[i]])Jul 4.817e+04  1.649e+05   0.292   0.7705
# seasonaldummy(tr[, var_des[i]])Aug 8.526e+04  1.649e+05   0.517   0.6057
# seasonaldummy(tr[, var_des[i]])Sep 1.603e+05  1.649e+05   0.972   0.3322
# seasonaldummy(tr[, var_des[i]])Oct 2.667e+05  1.649e+05   1.617   0.1074
# seasonaldummy(tr[, var_des[i]])Nov 3.317e+05  1.649e+05   2.011   0.0457 *
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 487600 on 202 degrees of freedom
# Multiple R-squared:  0.9686,  Adjusted R-squared:  0.9668
# F-statistic: 519.5 on 12 and 202 DF,  p-value: < 2.2e-16
#
# [1] "REMESAS"
#
# Call:
# lm(formula = tr[, var_des[i]] ~ c(1:nrow(tr)) + seasonaldummy(tr[,
#     var_des[i]]))
#
```

```
# Residuals:
#    Min     1Q Median     3Q    Max
# -754.8 -427.5 -176.8  428.7 1632.1
#
# Coefficients:
#                                    Estimate Std. Error t value Pr(>|t|)
# (Intercept)                        1208.4272   154.7380   7.810 3.06e-13 ***
# c(1:nrow(tr))                        10.7636     0.6303  17.076  < 2e-16 ***
# seasonaldummy(tr[, var_des[i]])Jan -314.5281   193.7944  -1.623   0.1061
# seasonaldummy(tr[, var_des[i]])Feb -231.3574   193.7852  -1.194   0.2339
# seasonaldummy(tr[, var_des[i]])Mar  213.8870   193.7780   1.104   0.2710
# seasonaldummy(tr[, var_des[i]])Apr   74.2358   193.7729   0.383   0.7020
# seasonaldummy(tr[, var_des[i]])May  369.9193   193.7698   1.909   0.0577 .
# seasonaldummy(tr[, var_des[i]])Jun  259.0376   193.7688   1.337   0.1828
# seasonaldummy(tr[, var_des[i]])Jul  206.1486   193.7698   1.064   0.2887
# seasonaldummy(tr[, var_des[i]])Aug  247.3364   193.7729   1.276   0.2033
# seasonaldummy(tr[, var_des[i]])Sep  101.6713   193.7780   0.525   0.6004
# seasonaldummy(tr[, var_des[i]])Oct  215.8630   193.7852   1.114   0.2666
# seasonaldummy(tr[, var_des[i]])Nov  -29.4050   193.7944  -0.152   0.8795
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 572.9 on 202 degrees of freedom
# Multiple R-squared:  0.6141,  Adjusted R-squared:  0.5912
# F-statistic: 26.79 on 12 and 202 DF,  p-value: < 2.2e-16
```

De acuerdo con la salida anterior, las series ANTAD, PROD_VEH, OCUP_HOT, GASOLINAS, IMSS y REMESAS tienen valores p significativos en las variables ficticias estacionales a un nivel de 0.1 o menos, por lo que estas series tienen problemas de estacionalidad. Para desestacionalizar las series se utiliza la librería *seasonal*, en particular la función *seas*. Después de desestacionalizar las series, se generaron nuevamente los modelos de regresión. El resultado se presenta a continuación.

```
# Desestacionalizar las series que resultaron significativas
# Crear una copia
tr_d<-tr
# Series a desestacionalizar
var_d<-c("ANTAD", "PROD_VEH", "OCUP_HOT", "GASOLINAS", "IMSS", "REMESAS")
# Desestacionalizar las series
for(i in 1:length(var_d)){
tr_d[, var_d[i]]<-seas(tr[, var_d[i]])$series$s11 }

# Revisar la estacionalidad
for(i in 1:length(var_d)){
estacionalidad<-lm(tr_d[, var_d[i]] ~ c(1:nrow(tr_d)) +
                     seasonaldummy(tr_d[, var_d[i]]))
# Resumen del modelo
print(var_d[i])
print(summary(estacionalidad)) }
```

```
# [1] "ANTAD"
#
# Call:
# lm(formula = tr_d[, var_d[i]] ~ c(1:nrow(tr_d)) + seasonaldummy(tr_d[,
```

```
#     var_d[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -41.812  -5.609  -0.237   3.863  29.627
#
# Coefficients:
#                                  Estimate Std. Error t value Pr(>|t|)
# (Intercept)                      70.13420    2.55721  27.426   <2e-16 ***
# c(1:nrow(tr_d))                   0.57140    0.01042  54.853   <2e-16 ***
# seasonaldummy(tr_d[, var_d[i]])Jan -0.89165    3.20266  -0.278    0.781
# seasonaldummy(tr_d[, var_d[i]])Feb  0.35713    3.20251   0.112    0.911
# seasonaldummy(tr_d[, var_d[i]])Mar  0.55446    3.20239   0.173    0.863
# seasonaldummy(tr_d[, var_d[i]])Apr -1.72989    3.20230  -0.540    0.590
# seasonaldummy(tr_d[, var_d[i]])May -1.64776    3.20225  -0.515    0.607
# seasonaldummy(tr_d[, var_d[i]])Jun -0.28742    3.20224  -0.090    0.929
# seasonaldummy(tr_d[, var_d[i]])Jul -0.14135    3.20225  -0.044    0.965
# seasonaldummy(tr_d[, var_d[i]])Aug  0.91386    3.20230   0.285    0.776
# seasonaldummy(tr_d[, var_d[i]])Sep  1.13973    3.20239   0.356    0.722
# seasonaldummy(tr_d[, var_d[i]])Oct  1.37103    3.20251   0.428    0.669
# seasonaldummy(tr_d[, var_d[i]])Nov  1.19848    3.20266   0.374    0.709
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 9.468 on 202 degrees of freedom
# Multiple R-squared:  0.9374,  Adjusted R-squared:  0.9337
# F-statistic:   252 on 12 and 202 DF,  p-value: < 2.2e-16
#
# [1] "PROD_VEH"
#
# Call:
# lm(formula = tr_d[, var_d[i]] ~ c(1:nrow(tr_d)) + seasonaldummy(tr_d[,
#     var_d[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -275449  -16993    6549   30056   72811
#
# Coefficients:
#                                   Estimate Std. Error t value Pr(>|t|)
# (Intercept)                      159748.65   12915.99  12.368   <2e-16 ***
# c(1:nrow(tr_d))                     893.44      52.61  16.981   <2e-16 ***
# seasonaldummy(tr_d[, var_d[i]])Jan  -6133.10   16176.03  -0.379    0.705
# seasonaldummy(tr_d[, var_d[i]])Feb  -6117.59   16175.26  -0.378    0.706
# seasonaldummy(tr_d[, var_d[i]])Mar -10814.67   16174.66  -0.669    0.505
# seasonaldummy(tr_d[, var_d[i]])Apr -23112.87   16174.23  -1.429    0.155
# seasonaldummy(tr_d[, var_d[i]])May -23146.62   16173.98  -1.431    0.154
# seasonaldummy(tr_d[, var_d[i]])Jun -11435.23   16173.89  -0.707    0.480
# seasonaldummy(tr_d[, var_d[i]])Jul  -6953.80   16173.98  -0.430    0.668
# seasonaldummy(tr_d[, var_d[i]])Aug  -7036.83   16174.23  -0.435    0.664
# seasonaldummy(tr_d[, var_d[i]])Sep  -7136.27   16174.66  -0.441    0.660
# seasonaldummy(tr_d[, var_d[i]])Oct  -7252.14   16175.26  -0.448    0.654
# seasonaldummy(tr_d[, var_d[i]])Nov  -7384.43   16176.03  -0.457    0.649
# ---
```

```
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 47820 on 202 degrees of freedom
# Multiple R-squared:  0.5925,  Adjusted R-squared:  0.5683
# F-statistic: 24.48 on 12 and 202 DF,  p-value: < 2.2e-16
#
# [1] "OCUP_HOT"
#
# Call:
# lm(formula = tr_d[, var_d[i]] ~ c(1:nrow(tr_d)) + seasonaldummy(tr_d[,
#     var_d[i]]))
#
# Residuals:
#    Min     1Q Median     3Q    Max
# -60.58  -4.00   3.07   8.43  14.10
#
# Coefficients:
#                                    Estimate Std. Error t value Pr(>|t|)
# (Intercept)                        70.30630    3.59877  19.536  < 2e-16 ***
# c(1:nrow(tr_d))                    -0.05005    0.01466  -3.414 0.000773 ***
# seasonaldummy(tr_d[, var_d[i]])Jan  0.05920    4.50711   0.013 0.989533
# seasonaldummy(tr_d[, var_d[i]])Feb  0.45632    4.50689   0.101 0.919452
# seasonaldummy(tr_d[, var_d[i]])Mar -0.98461    4.50673  -0.218 0.827279
# seasonaldummy(tr_d[, var_d[i]])Apr -3.14634    4.50661  -0.698 0.485879
# seasonaldummy(tr_d[, var_d[i]])May -4.18765    4.50653  -0.929 0.353874
# seasonaldummy(tr_d[, var_d[i]])Jun -2.82381    4.50651  -0.627 0.531625
# seasonaldummy(tr_d[, var_d[i]])Jul -2.36624    4.50653  -0.525 0.600112
# seasonaldummy(tr_d[, var_d[i]])Aug -0.79921    4.50661  -0.177 0.859418
# seasonaldummy(tr_d[, var_d[i]])Sep  0.36480    4.50673   0.081 0.935566
# seasonaldummy(tr_d[, var_d[i]])Oct -0.06180    4.50689  -0.014 0.989073
# seasonaldummy(tr_d[, var_d[i]])Nov -0.95143    4.50711  -0.211 0.833025
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 13.32 on 202 degrees of freedom
# Multiple R-squared:  0.06623, Adjusted R-squared:  0.01076
# F-statistic: 1.194 on 12 and 202 DF,  p-value: 0.2893
#
# [1] "GASOLINAS"
#
# Call:
# lm(formula = tr_d[, var_d[i]] ~ c(1:nrow(tr_d)) + seasonaldummy(tr_d[,
#     var_d[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -254.53  -11.75   17.13   30.67   67.12
#
# Coefficients:
#                                     Estimate Std. Error t value Pr(>|t|)
# (Intercept)                        778.85391   13.64056  57.098   <2e-16 ***
# c(1:nrow(tr_d))                      0.03292    0.05557   0.593    0.554
# seasonaldummy(tr_d[, var_d[i]])Jan  -7.87878   17.08349  -0.461    0.645
# seasonaldummy(tr_d[, var_d[i]])Feb -13.36800   17.08267  -0.783    0.435
```

```
# seasonaldummy(tr_d[, var_d[i]])Mar  -5.93378   17.08204  -0.347    0.729
# seasonaldummy(tr_d[, var_d[i]])Apr -12.94176   17.08159  -0.758    0.450
# seasonaldummy(tr_d[, var_d[i]])May -16.96650   17.08132  -0.993    0.322
# seasonaldummy(tr_d[, var_d[i]])Jun  -8.40540   17.08123  -0.492    0.623
# seasonaldummy(tr_d[, var_d[i]])Jul  -7.45188   17.08132  -0.436    0.663
# seasonaldummy(tr_d[, var_d[i]])Aug  -3.21494   17.08159  -0.188    0.851
# seasonaldummy(tr_d[, var_d[i]])Sep   0.06851   17.08204   0.004    0.997
# seasonaldummy(tr_d[, var_d[i]])Oct  -1.32898   17.08267  -0.078    0.938
# seasonaldummy(tr_d[, var_d[i]])Nov  -0.39232   17.08349  -0.023    0.982
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 50.51 on 202 degrees of freedom
# Multiple R-squared:  0.01451, Adjusted R-squared:  -0.04404
# F-statistic: 0.2478 on 12 and 202 DF,  p-value: 0.9954
#
# [1] "IMSS"
#
# Call:
# lm(formula = tr_d[, var_d[i]] ~ c(1:nrow(tr_d)) + seasonaldummy(tr_d[,
#     var_d[i]]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -859395 -362526  -14361  427230  809314
#
# Coefficients:
#                                    Estimate Std. Error t value Pr(>|t|)
# (Intercept)                       12376968.4   131376.6  94.210   <2e-16 ***
# c(1:nrow(tr_d))                      42235.6      535.2  78.920   <2e-16 ***
# seasonaldummy(tr_d[, var_d[i]])Jan   23824.9   164536.6   0.145    0.885
# seasonaldummy(tr_d[, var_d[i]])Feb   26529.8   164528.8   0.161    0.872
# seasonaldummy(tr_d[, var_d[i]])Mar   19870.9   164522.7   0.121    0.904
# seasonaldummy(tr_d[, var_d[i]])Apr   -7210.2   164518.3  -0.044    0.965
# seasonaldummy(tr_d[, var_d[i]])May  -23688.3   164515.7  -0.144    0.886
# seasonaldummy(tr_d[, var_d[i]])Jun  -20949.6   164514.8  -0.127    0.899
# seasonaldummy(tr_d[, var_d[i]])Jul  -17882.3   164515.7  -0.109    0.914
# seasonaldummy(tr_d[, var_d[i]])Aug  -14418.6   164518.3  -0.088    0.930
# seasonaldummy(tr_d[, var_d[i]])Sep  -10761.4   164522.7  -0.065    0.948
# seasonaldummy(tr_d[, var_d[i]])Oct   -6999.9   164528.8  -0.043    0.966
# seasonaldummy(tr_d[, var_d[i]])Nov   -3474.1   164536.6  -0.021    0.983
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 486400 on 202 degrees of freedom
# Multiple R-squared:  0.9686,  Adjusted R-squared:  0.9668
# F-statistic: 520.1 on 12 and 202 DF,  p-value: < 2.2e-16
#
# [1] "REMESAS"
#
# Call:
# lm(formula = tr_d[, var_d[i]] ~ c(1:nrow(tr_d)) + seasonaldummy(tr_d[,
#     var_d[i]]))
#
```

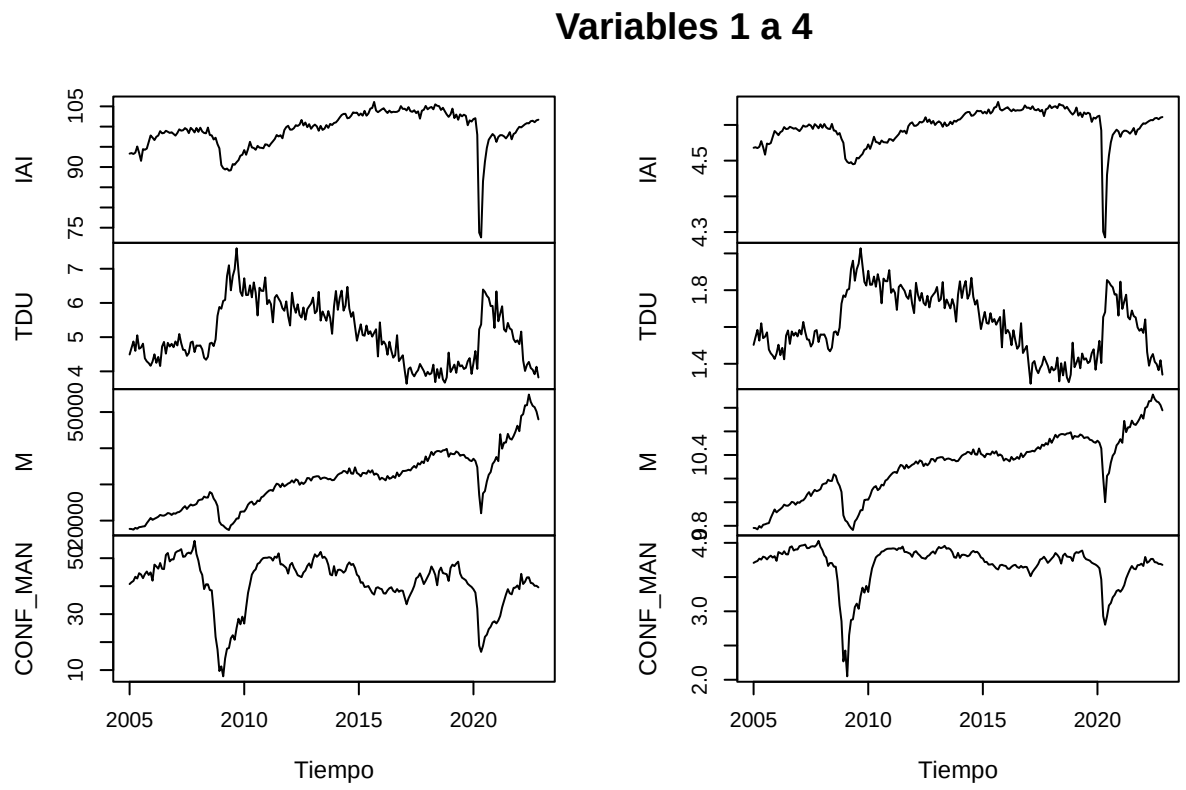
```
# Residuals:
#    Min     1Q Median     3Q    Max
# -710.9 -442.4 -199.7  419.4 1504.4
#
# Coefficients:
#                                    Estimate Std. Error t value Pr(>|t|)
# (Intercept)                      1249.6389   151.4179   8.253 1.98e-14 ***
# c(1:nrow(tr_d))                    10.7510     0.6168  17.430  < 2e-16 ***
# seasonaldummy(tr_d[, var_d[i]])Jan  27.2675   189.6363   0.144    0.886
# seasonaldummy(tr_d[, var_d[i]])Feb  20.6985   189.6273   0.109    0.913
# seasonaldummy(tr_d[, var_d[i]])Mar  68.0603   189.6202   0.359    0.720
# seasonaldummy(tr_d[, var_d[i]])Apr  36.6716   189.6152   0.193    0.847
# seasonaldummy(tr_d[, var_d[i]])May  38.1899   189.6122   0.201    0.841
# seasonaldummy(tr_d[, var_d[i]])Jun  47.4489   189.6112   0.250    0.803
# seasonaldummy(tr_d[, var_d[i]])Jul  58.5313   189.6122   0.309    0.758
# seasonaldummy(tr_d[, var_d[i]])Aug  58.2385   189.6152   0.307    0.759
# seasonaldummy(tr_d[, var_d[i]])Sep  66.8201   189.6202   0.352    0.725
# seasonaldummy(tr_d[, var_d[i]])Oct 113.2371   189.6273   0.597    0.551
# seasonaldummy(tr_d[, var_d[i]])Nov  80.6918   189.6363   0.426    0.671
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 560.6 on 202 degrees of freedom
# Multiple R-squared:  0.6023,  Adjusted R-squared:  0.5787
# F-statistic:  25.5 on 12 and 202 DF,  p-value: < 2.2e-16
```

Con base a la salida previa, las series ya no presentan valores p significativos en las variables ficticias estacionales, por lo que los problemas de estacionalidad fueron resueltos.

Posteriormente, se aplicaron logaritmos a las series con la intención de observar si la varianza tiene una mejora relevante. No obstante, al comparar la serie contra su logaritmo no se observaron cambios relevantes, por lo que se optó por trabajar con las series sin ningún tipo de transformación. Para no hacer tan extenso el documento, se decidió imprimir únicamente el primer gráfico para las 4 primeras series. Sin embargo, el código utilizado para todo el conjunto de series se muestra enseguida.

```
# Gráfico temporal para las primeras 4 series
series<-cbind(tr_d[, 1:4], log(tr_d[, 1:4]))
colnames(series)<-rep(colnames(tr_d)[1:4], 2)
plot(series, cex.lab = 0.75, main = "Variables 1 a 4", xlab = "Tiempo")
# Gráfico temporal para las segundas 4 series
series<-cbind(tr_d[, 5:8], log(tr_d[, 5:8]))
colnames(series)<-rep(colnames(tr_d)[5:8], 2)
plot(series, cex.lab = 0.75, main = "Variables 5 a 8", xlab = "Tiempo")
# Gráfico temporal para las terceras 4 series
series<-cbind(tr_d[, 9:12], log(tr_d[, 9:12]))
colnames(series)<-rep(colnames(tr_d)[9:12], 2)
plot(series, cex.lab = 0.75, main = "Variables 9 a 12", xlab = "Tiempo")
# Gráfico temporal para las cuartas 4 series
series<-cbind(tr_d[, 13:16], log(tr_d[, 13:16]))
colnames(series)<-rep(colnames(tr_d)[13:16], 2)
plot(series, cex.lab = 0.75, main = "Variables 13 a 16", xlab = "Tiempo")
# Gráfico temporal para las quintas 4 series
series<-cbind(tr_d[, 17:20], log(tr_d[, 17:20]))
colnames(series)<-rep(colnames(tr_d)[17:20], 2)
```

```r
plot(series, cex.lab = 0.75, main = "Variables 17 a 20", xlab = "Tiempo")
# Gráfico temporal para las sextas 4 series
series<-cbind(tr_d[, 21:24], log(tr_d[, 21:24]))
colnames(series)<-rep(colnames(tr_d)[21:24], 2)
plot(series, cex.lab = 0.75, main = "Variables 21 a 24", xlab = "Tiempo")
# Gráfico temporal para las últimas 4 series
series<-cbind(tr_d[, 25:28], log(tr_d[, 25:28]))
colnames(series)<-rep(colnames(tr_d)[25:28], 2)
plot(series, cex.lab = 0.6, main = "Variables 25 a 28", xlab = "Tiempo")
```



De acuerdo con la gráfica anterior, del lado izquierdo se presentan las series sin ningún tipo de transformación, mientras que, del lado derecho se tienen las series transformadas con logaritmo. Como se puede apreciar, no existen cambios relevantes en cuanto a la varianza, ya que la forma de ambas series es prácticamente la misma. Esta misma situación se presentó en el resto de las series, por lo que se optó por trabajar con las series sin ningún tipo de transformación.

Después, se realizó la prueba aumentada de *Dickey Fuller* con la intención de determinar el orden de integración de las 28 series del estudio. Las hipótesis de la prueba son las siguientes:

H_o : La serie de tiempo es no estacionaria
H_a : La serie de tiempo es estacionaria

La regla de decisión es rechazar la hipótesis nula (H_o) si el valor p es menor que un nivel de significancia de 0.05 ($\alpha = 0.05$). La salida de los valores p de la prueba, tanto para la serie original como diferenciada se muestra a continuación.

```r
# Matriz para pruebas ADF
adf_tests<-matrix(NA, N, 2)
# Nombres de columnas y renglones
colnames(adf_tests)<-c("Original", "Diferenciada")
rownames(adf_tests)<-colnames(tr_d)

# Ciclo for para las pruebas
for(i in 1:N){
adf_tests[i, "Original"]<-adf(tr_d[, i], "const")$p.value
adf_tests[i, "Diferenciada"]<-adf(diff(tr_d[, i]), "const")$p.value }

# Se imprime el resultado de las pruebas ADF
kbl(adf_tests, longtable = T, booktabs = T,
    caption = "Resultado de la prueba ADF",
    col.names = c("Sin diferenciar", "Diferenciada"),
    escape = TRUE, digits = 3)
```

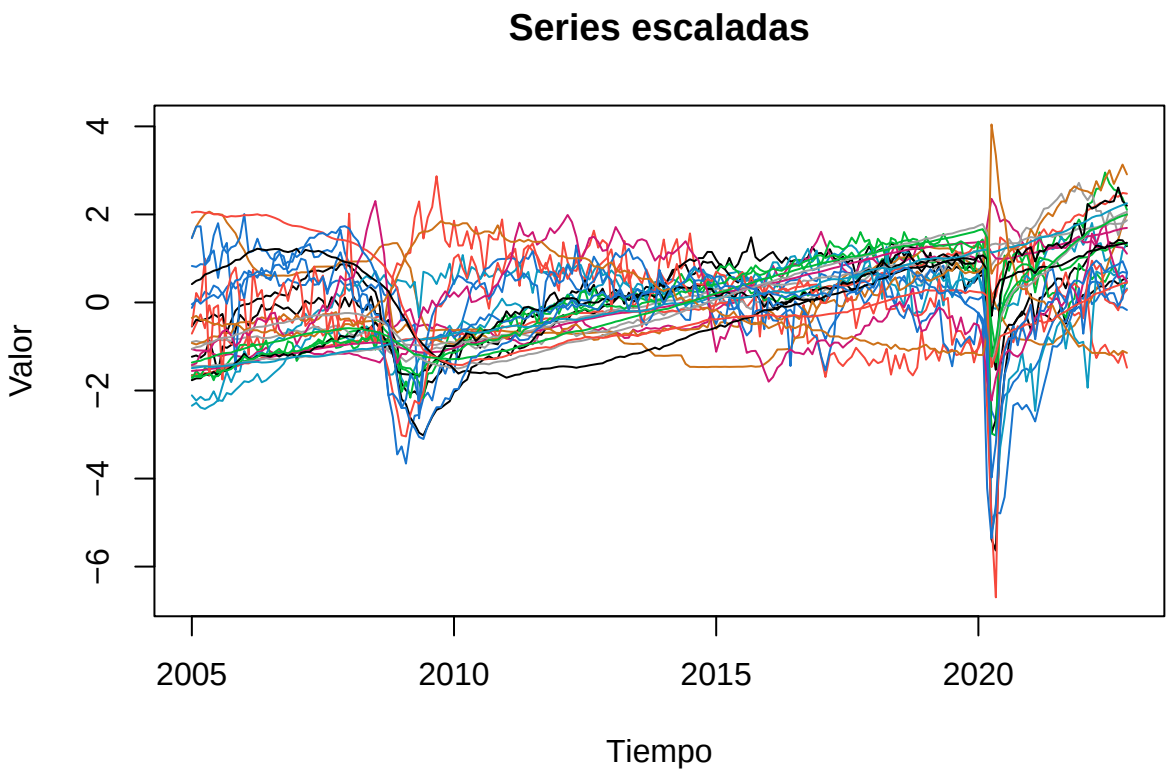
Cuadro 2: Resultado de la prueba ADF

	Sin diferenciar	Diferenciada
IAI	0.086	0.01
TDU	0.376	0.01
M	0.627	0.01
CONF_MAN	0.064	0.01
IPC	0.266	0.01
TC	0.703	0.01
TIIE_28	0.239	0.02
SP_500	0.962	0.01
IPI_EUA	0.266	0.01
ANTAD	0.990	0.01
PROD_VEH	0.270	0.01
OCUP_HOT	0.082	0.01
GASOLINAS	0.025	0.01
IMSS	0.939	0.01
REMESAS	0.990	0.01
M4	0.990	0.01
X	0.858	0.01
PEDIDOS_MANU	0.032	0.01
IGAE	0.539	0.01
IMEF	0.026	0.01
INPC	0.990	0.01
PRECIO	0.123	0.01
U_US	0.299	0.01
TOTAL_NONFARM	0.848	0.01
TOTAL_CONSTR	0.693	0.01
TOTAL_MANUF	0.262	0.01
TOTAL_SERV	0.859	0.01
MANUF_USA	0.134	0.01

La tabla anterior contiene el resultado de los valores p de la prueba *ADF* tanto para la serie original como la serie diferenciada. En ella se puede observar que, la gran mayoría de las series no son estacionarias en los datos sin diferenciar, excepto por GASOLINAS, PEDIDOS_MANU e IMEF, las cuales sí son estacionarias en las series sin diferenciar. El resto de las series son estacionarias en primeras diferencias, por lo que su orden de integración es 1.

Otro punto relevante antes de continuar con el análisis es el de escalar las series, ya que al igual que en un análisis de factores, los cambios de escala podrían afectar drásticamente los resultados.

```
# Escalar las series para evitar problemas de diferentes escalas
tr_sc<-scale(tr_d, center = TRUE, scale = TRUE)
# Series escaladas
ts.plot(tr_sc, col = 1:8, main = "Series escaladas",
        xlab = "Tiempo", ylab = "Valor")
```



De acuerdo con la gráfica previa, existen series que tienen un comportamiento similar, por lo que uno de los factores podría capturar la forma de las series que presentan un dato atípico cerca del año 2020. Otro factor podría capturar esa curva inicial que distingue a las variables TOTAL_CONSTR, TOTAL_MANUF y MANUF_USA. Otro posible factor podría capturar el comportamiento que tuvieron algunas series con el dato atípico cerca del año 2010. Otro posible factor podría estar dominado por alguna serie que no tenga un comportamiento similar al resto. En resumen, un estimado visual y razonable del número de factores que tendría el modelo de factores dinámicos *(MFD)* sería de entre 3 y 4 factores.

Luego, se calculó la dependencia efectiva muestral, que es la autocorrelación multivariada entre un conjunto de series de tiempo. Cuando este valor es cercano a 0 significa que no hay nada de dependencia entre las variables del estudio, mientras que, cuando es cercano a 1 significa que hay total dependencia entre las variables de estudio y que serie viable ajustar un *MFD*. El resultado se muestra a continuación.

```r
# Dependencia efectiva muestral
dem<-1 - det(cor(tr_sc))^(1/(N-1))
# Se imprime el resultado
dem
```

```
# [1] 0.9624063
```

De acuerdo con la salida anterior, el valor de la dependencia efectiva muestral fue de 0.9624063, por lo que el ajuste de un modelo de factores dinámicos es bastante viable. La siguiente parte del análisis consiste en calcular el número de factores que se incluirán en el modelo.

7.2 Determinación del número de factores

Para la selección del número de factores en el *MFD* se utilizan los criterios de información de *Bai and Ng (2002)*, *Onatski (2010)* y *Ahn and Horenstein (2013)*. El resultado para *Bai and Ng* se presenta a continuación.

```r
### Determinar el número de factores comunes
# Fijar kmax
kmax<-N*0.25

# Función de Bai and Ng
# Se debe ejecutar ya que las funciones no la traen
bai_ng <- function(Y, kmax = kmax){
  ic <- matrix(0, kmax + 1, 3)
  colnames(ic) <- c("ICP1k","ICP2k", "ICP3k")
  for(j in 0 : kmax)
    ic[j+1, ] <- pcfest(Y, demean = 0, r = j)$ICPk
  rhat <- apply(ic, 2, function(x) which(x == min(x)) - 1)
  return(rhat)
}

# Bai and Ng (2002)
r_hat_bai_ng<-bai_ng(tr_sc, kmax = kmax)
r_hat_bai_ng
```

```
# ICP1k ICP2k ICP3k
#     7     7     7
```

Cabe destacar que, para establecer el número de factores máximos (*kmax*) del modelo, una sugerencia es colocar 1/4 del total de las variables de estudio, que en este caso el *kmax* sería de 7. De acuerdo con la salida anterior, el criterio de información de *Bai and Ng* estima que el modelo contenga 7 factores; cuando este criterio toma el valor máximo fijado, posiblemente se tenga problemas con los errores autocorrelacionados, ya que este criterio es muy sensible a ese problema.

```r
# Onatski (2010)
# demean = 0 significa sin escalamiento
r_hat_ed<-onatski2010(tr_sc, demean = 0)
r_hat_ed
```

```
#       ed  betahat
# 3.000000 1.527311
```

El siguiente criterio de información fue el de *Onatski*, que de acuerdo con la salida previa sugiere que en el modelo de factores dinámicos se incluyan un total de 3 factores.

```
# Ahn y Horenstein (2013)
r_hat_ratio<-ratio.test(tr_sc, kmax = kmax, demean = 0)[c("ker", "kgr")]
r_hat_ratio
```

```
# ker kgr
#   3   3
```

Por último, los criterios de información de *Ahn y Horenstein* sugieren que el modelo de factores dinámicos incluya un total de 3 factores. Por lo anterior y derivado del análisis exploratorio, se optó por incluir 3 factores en el modelo, ya que la inclusión de 7 factores se descarta por la situación comentada con anterioridad.

7.3 Estimación de factores

Para la estimación de factores se optó por implementar los cinco métodos, con la intención de comparar los resultados obtenidos y seleccionar el más apropiado. Los métodos son:

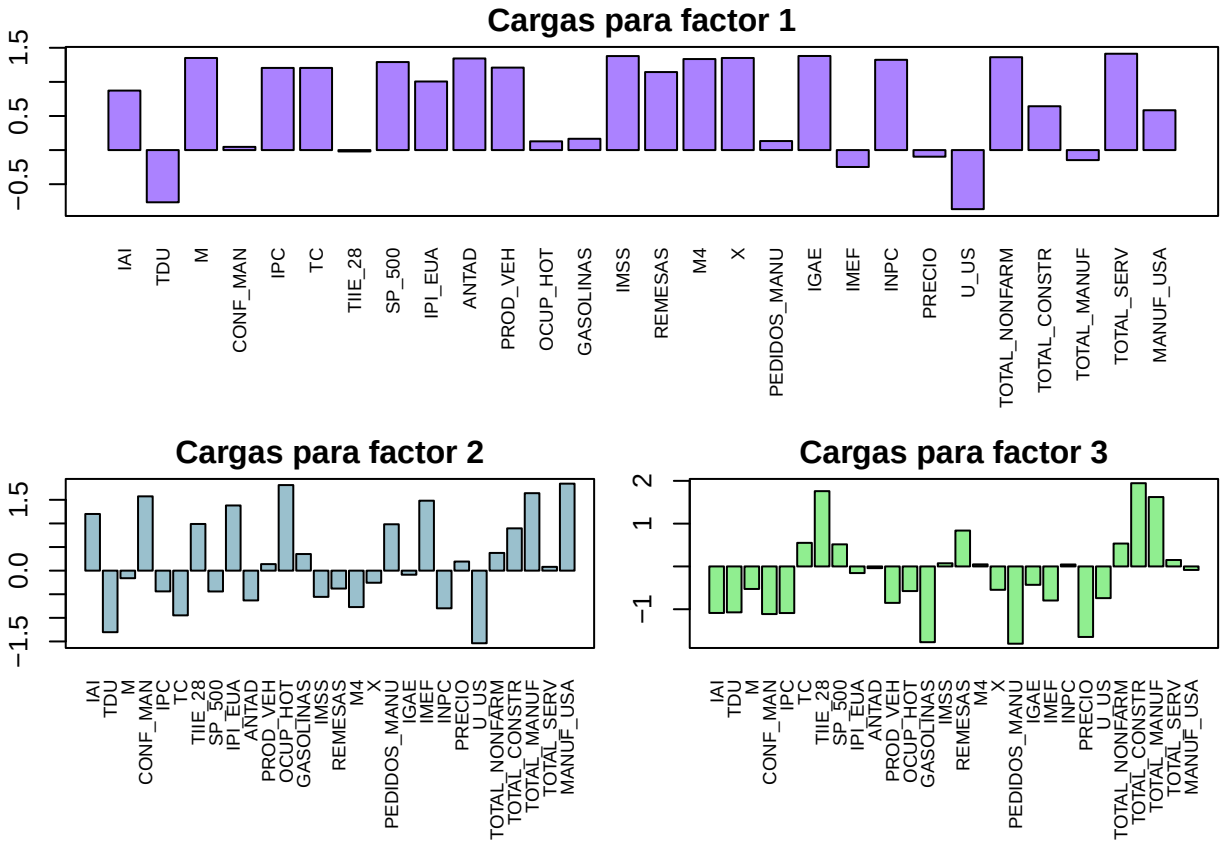
a) Estimación en niveles.
b) Estimación en diferencias.
c) Componentes principales generalizados (GPCE).
d) Barigozzi.
e) Factor dinámico (suavizado de Kalman en 2 pasos).

Método a) Estimación en niveles

El código implementado en R se presenta a continuación.

```
### Estimación de factores
# Número de factores
r<-3
## Método a) Estimación en niveles
# Ajuste del modelo
pc_lev<-pcfest(tr_sc, r = r)
fhat<-pc_lev$Fhat
# Hacerlo serie de tiempo
fhat_a<-ts(scale(fhat), start = c(2005, 1), frequency = 12)
# Extraer las cargas
Phat_a<-pc_lev$lambda
# Colocar los nombres de columna a los renglones de las cargas
rownames(Phat_a)<-colnames(tr_sc)

# Gráfico de las cargas estimadas
par(mar = c(6.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
barplot(Phat_a[, 1], las = 3, cex.names = 0.7, col = "mediumpurple1",
        main = "Cargas para factor 1", ylim = c(min(Phat_a[, 1]) - 0.1,
                                                max(Phat_a[, 1]) + 0.1))
box()
barplot(Phat_a[, 2], las = 3, cex.names = 0.7, col = "lightblue3",
        main = "Cargas para factor 2", ylim = c(min(Phat_a[, 2]) - 0.1,
```

```
                                                max(Phat_a[, 2]) + 0.1))
box()
barplot(Phat_a[, 3], las = 3, cex.names = 0.7, col = "lightgreen",
        main = "Cargas para factor 3", ylim = c(min(Phat_a[, 3]) - 0.1,
                                                max(Phat_a[, 3]) + 0.1))
box()
```



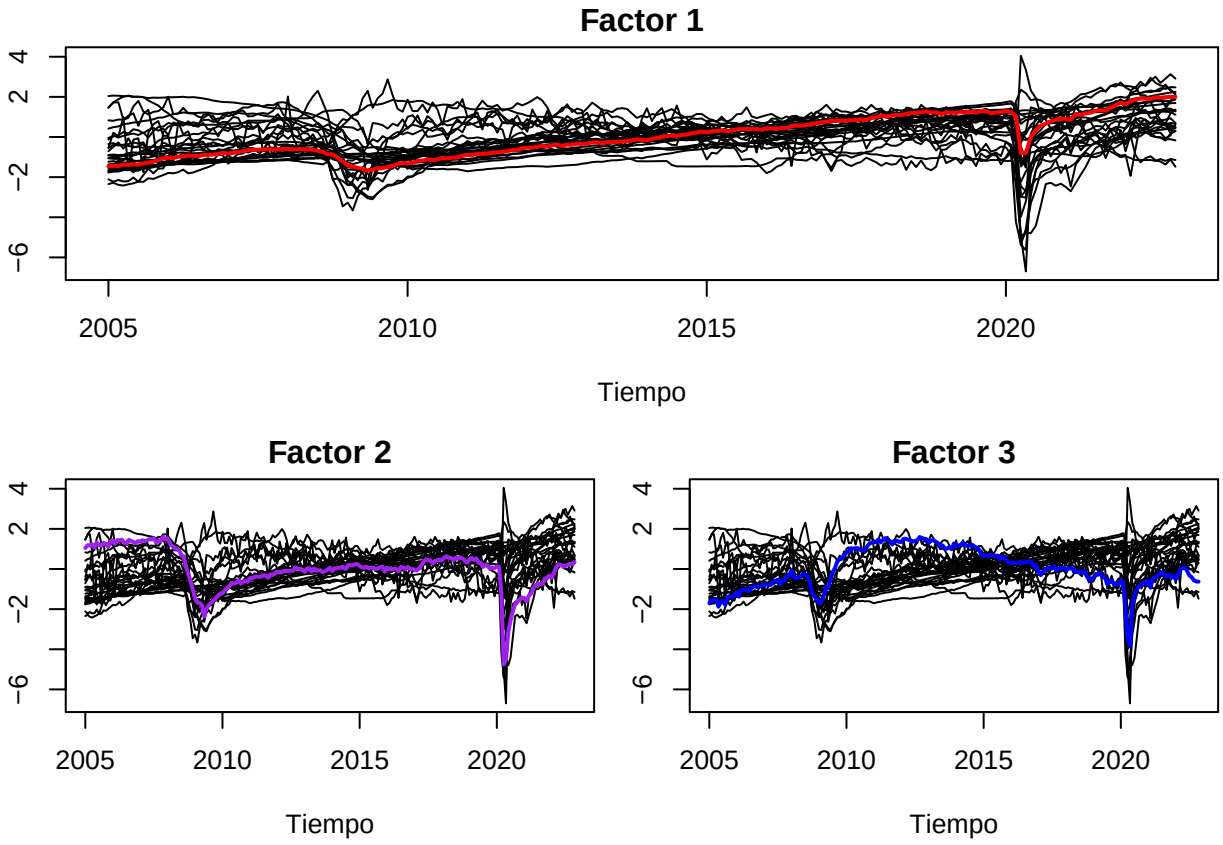
La salida anterior muestra las cargas estimadas de los factores por el método de estimación en niveles. En él se puede apreciar que, el primer factor es una combinación de la gran mayoría de las series, excepto por: CONF_MAN, TIIE_28, OCUP_HOT, GASOLINAS, PEDIDOS_MANU, IMEF, PRECIO y TOTAL_MANUF, ya que sus cargas estimadas son cercanas a 0. Las series TDU y U_US dominan este factor de manera negativa, en cambio, el resto de las series con cargas mayores a 0 lo dominan de forma positiva.

En cuanto al segundo factor, lo dominan principalmente de forma positiva las series de OCUP_HOT, MANUF_USA, CONF_MAN, TOTAL_MANUF, IMEF, IPI_EUA, TIIE_28, PEDIDOS_MANU y TOTAL_CONSTR, por el contrario, de manera negativa se tiene principalmente a las series TDU, U_US, TC, M4, INPC y ANTAD.

En lo que respecta al factor 3, las series que lo dominan de forma positiva son TOTAL_CONSTR, TOTAL_MANUF, TIIE_28 y REMESAS, en sentido opuesto se tiene principalmente a las series de PRECIO, PEDIDOS_MANU, GASOLINAS, IAI, TDU, CONF_MAN e IPC.

```
# Gráfico para los factores
par(mar = c(4.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
ts.plot(tr_sc, col = "black", main = "Factor 1",
```

```
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_a[, 1], col = "red", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 2",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_a[, 2], col = "purple", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 3",
        xlab = "Tiempo", ylab = "Valor")
lines(-fhat_a[, 3], col = "blue", lwd = 2)
```



De acuerdo con la gráfica previa, el primer factor trata de capturar la tendencia de 2010 a 2020 que se percibe en algunas series, por ejemplo, en M, SP500 e IGAE se puede distinguir que después de la caída que se tiene en 2010, la serie vuelve a subir hasta caer nuevamente cerca del año 2020, situación que está replicando este factor; también trata de capturar un poco esas dos situaciones atípicas.

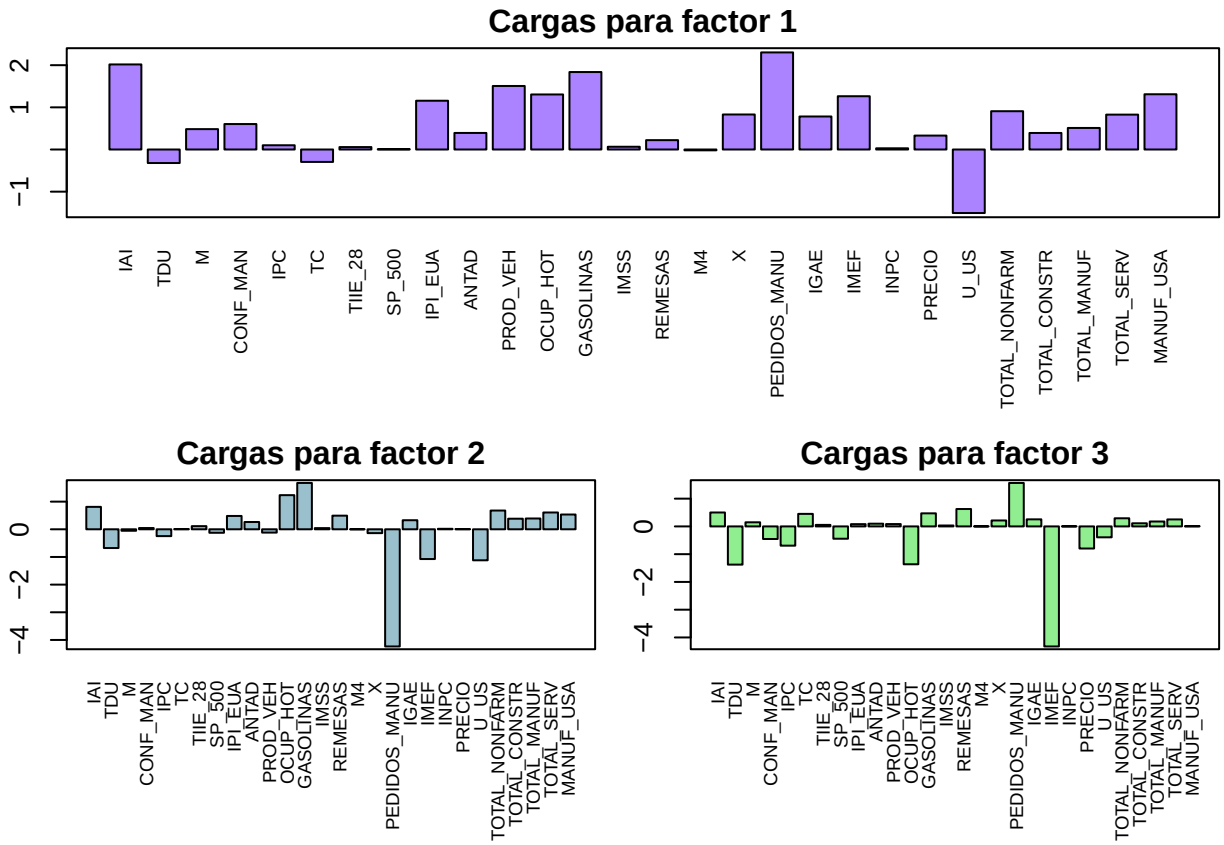
El segundo factor captura la curva inicial que se tiene cerca del año 2010 en las series TOTAL_CONSTR, TOTAL_MANUF y MANUF_USA. También captura de mejor manera esas dos situaciones atípicas que se aprecian cerca de 2010 y de 2020.

Por último, el tercer factor también captura un poco esas dos situaciones atípicas que se tienen cerca de los años 2010 y 2020. A diferencia de los dos factores anteriores, a partir de 2010 tiene un alza importante y después una caída hasta llegar al año de 2020.

Método b) Estimación en diferencias

El código realizado en R se presenta a continuación.

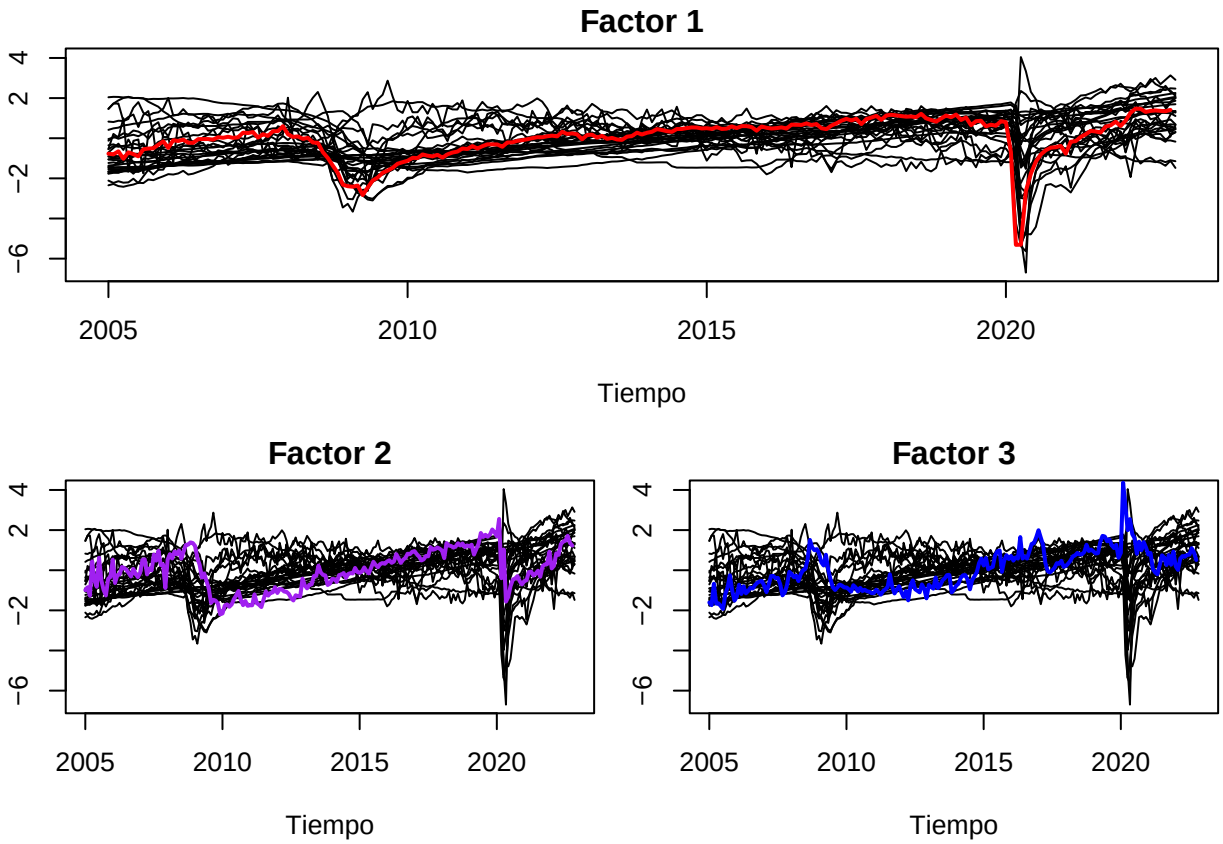
```r
## Método b) Estimación en diferencias
# Ajuste del modelo
pc_dif<-pcfest(diff(tr_sc), r = r)
fhat<-apply(pc_dif$Fhat, 2, cumsum)
# Hacerlo serie de tiempo
fhat_b<-ts(scale(fhat), start = c(2005, 1), frequency = 12)
# Extraer las cargas
Phat_b<-pc_dif$lambda
# Colocar los nombres de columna a los renglones de las cargas
rownames(Phat_b)<-colnames(tr_sc)

# Gráfico de las cargas estimadas
par(mar = c(6.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
barplot(Phat_b[, 1], las = 3, cex.names = 0.75, col = "mediumpurple1",
        main = "Cargas para factor 1", ylim = c(min(Phat_b[, 1]) - 0.1,
                                                max(Phat_b[, 1]) + 0.1)); box();
barplot(Phat_b[, 2], las = 3, cex.names = 0.75, col = "lightblue3",
        main = "Cargas para factor 2", ylim = c(min(Phat_b[, 2]) - 0.1,
                                                max(Phat_b[, 2]) + 0.1)); box();
barplot(Phat_b[, 3], las = 3, cex.names = 0.75, col = "lightgreen",
        main = "Cargas para factor 3", ylim = c(min(Phat_b[, 3]) - 0.1,
                                                max(Phat_b[, 3]) + 0.1)); box()
```



Las gráficas previas contienen las cargas estimadas de los factores a través del método de estimación en diferencias. En él se puede apreciar que, para el primer factor las series que lo dominan principalmente de forma positiva son IAI, PEDIDOS_MANU, GASOLINAS, PROD_VEH, OCUP_HOT, MANUF_USA e IPI_EUA, en cambio, de manera negativa se tiene a U_US.

En lo que respecta al segundo factor, es dominado de manera positiva por las series de OCUP_HOT y GASOLINAS, mientras que, de manera negativa se tiene a la serie de PEDIDOS_MANU, U_US e IMEF. Por último, el tercer factor está dominado de forma positiva por PEDIDOS_MANUF, por el contrario, de forma negativa se tiene a las series de IMEF, TDU y OCUP_HOT.

```
# Gráfico para los factores
par(mar = c(4.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
ts.plot(tr_sc, col = "black", main = "Factor 1",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_b[, 1], col = "red", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 2",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_b[, 2], col = "purple", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 3",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_b[, 3], col = "blue", lwd = 2)
```

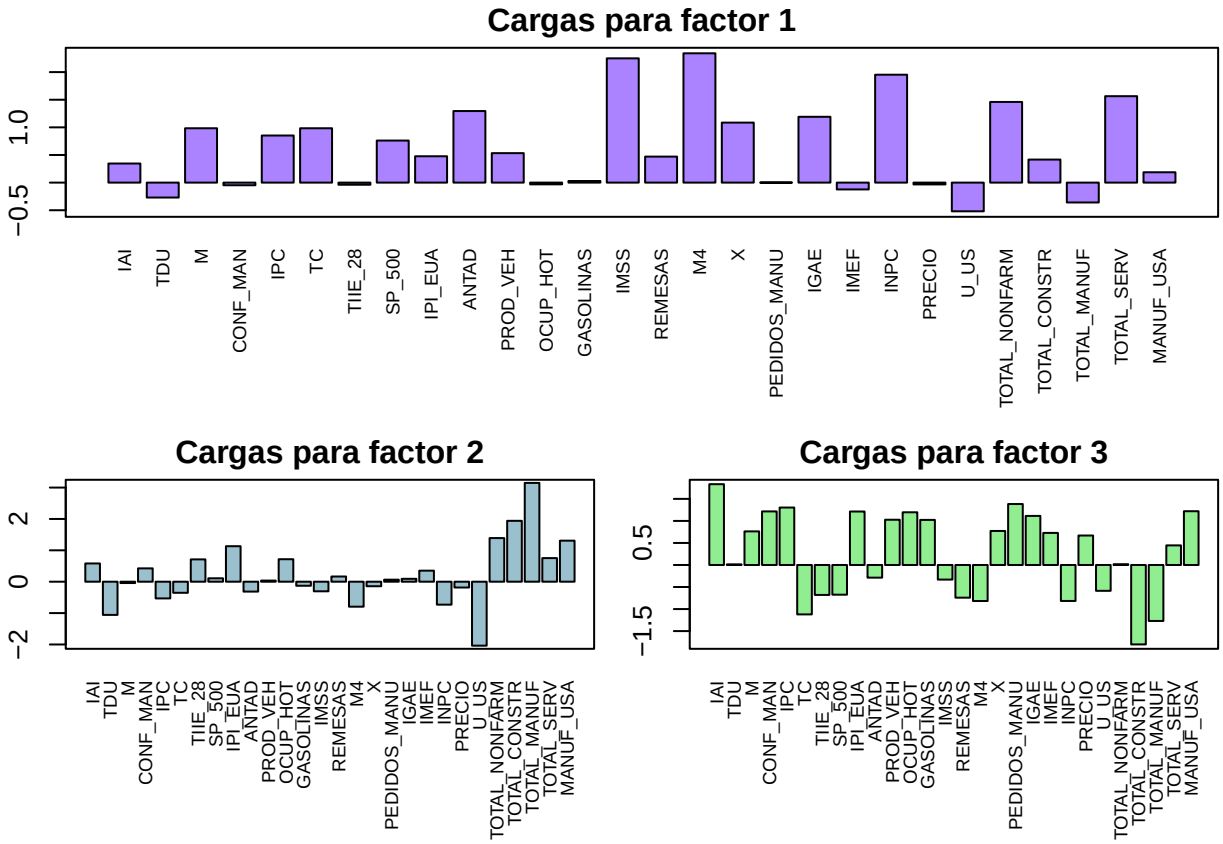


Con base a la salida anterior, el primer factor captura la curva inicial que se tiene hasta aproximadamente el año 2010 en las series TOTAL_CONSTR, TOTAL_MANUF y MANUF_USA. También captura de mejor manera esas dos situaciones atípicas que se aprecian cerca de 2010 y de 2020. Por otra parte, el factor 2 al estar dominado principalmente por la serie de PEDIDOS_MANUF tiende a replicar su comportamiento. De igual forma el factor 3 tiende a replicar el comportamiento, pero de la serie del IMEF.

Método c) Componentes principales generalizados (GPCE)

El código de R se presenta enseguida.

```
## Método c) Componentes Principales Generalizados (GPCE)
# Ajuste del modelo
pc_lev<-pcfest(tr_sc, r = r)
fhat<-pc_lev$Fhat
ehat<-pc_lev$ehat
Sigma_ehat<-t(ehat) %*% ehat
Ygls<-tr_sc %*% diag(diag(Sigma_ehat)^(-1/2))
# Ajuste del modelo
pc_choi<-pcfest(Ygls, r = r)
# Hacerlo serie de tiempo
fhat_c<-ts(scale(pc_choi$Fhat), start = c(2005, 1), frequency = 12)
# Extraer las cargas
Phat_c<-pc_choi$lambda
# Colocar los nombres de columna a los renglones de las cargas
rownames(Phat_c)<-colnames(tr_sc)

# Gráfico de las cargas estimadas
par(mar = c(6.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
barplot(Phat_c[, 1], las = 3, cex.names = 0.75, col = "mediumpurple1",
        main = "Cargas para factor 1", ylim = c(min(Phat_c[, 1]) - 0.1,
                                                max(Phat_c[, 1]) + 0.1))
box()
barplot(Phat_c[, 2], las = 3, cex.names = 0.75, col = "lightblue3",
        main = "Cargas para factor 2", ylim = c(min(Phat_c[, 2]) - 0.1,
                                                max(Phat_c[, 2]) + 0.1))
box()
barplot(Phat_c[, 3], las = 3, cex.names = 0.75, col = "lightgreen",
        main = "Cargas para factor 3", ylim = c(min(Phat_c[, 3]) - 0.1,
                                                max(Phat_c[, 3]) + 0.1))
box()
```

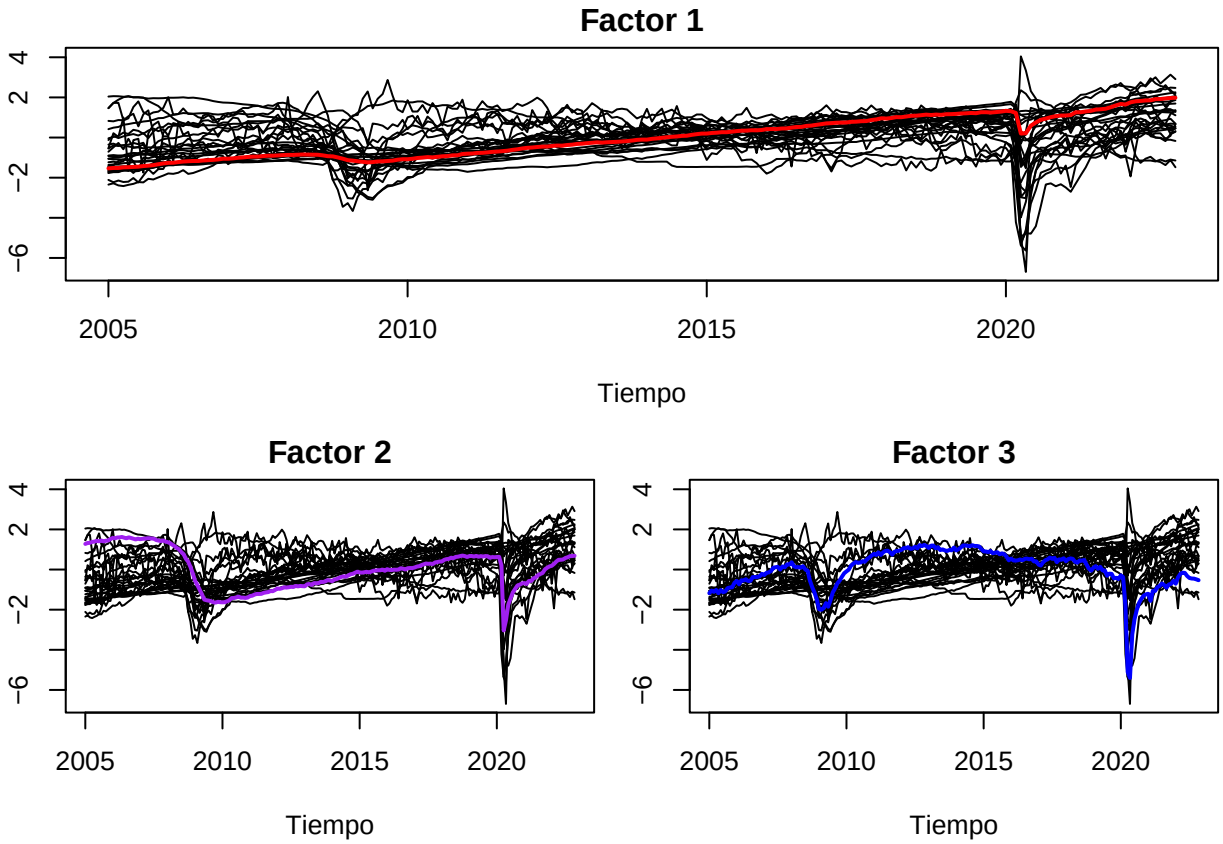


La salida anterior muestra las cargas estimadas de los factores por el método de componentes principales generalizados (GPCE). En él se puede apreciar que, el primer factor se encuentra dominado de forma positiva por las series M4, IMSS, INPC, TOTAL_SERV, TOTAL_NONFARM, ANTAD, entre otras, en cambio, de forma negativa no se tiene una serie que destaque, ya que las cargas son muy cercanas a 0.

En lo que corresponde al segundo factor, las series que lo dominan de forma positiva son TOTAL_MANUF, TOTAL_CONSTR, TOTAL_NONFARM y MANUF_USA, por el contrario, de manera negativa se tiene a U_US, TDU, M4 e INPC.

Por último, para el tercer factor hay varias series que lo dominan de manera positiva, destacando IAI, PEDIDOS_MANU, IPC, CONF_MAN, entre otras, por el contrario, de forma negativa se tiene a las series TOTAL_CONSTR, TOTAL_MANU y TC.

```
# Gráfico para los factores
par(mar = c(4.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
ts.plot(tr_sc, col = "black", main = "Factor 1",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_c[, 1], col = "red", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 2",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_c[, 2], col = "purple", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 3",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_c[, 3], col = "blue", lwd = 2)
```



De acuerdo con la gráfica previa, el primer factor trata de capturar la tendencia de 2010 a 2020 que se percibe en algunas series, por ejemplo, en M, SP500 e IGAE se puede distinguir que después de la caída que se tiene en 2010, la serie vuelve a subir hasta caer nuevamente en aproximadamente 2020, situación que está replicando este factor; también trata de capturar un poco esas dos situaciones atípicas.

El segundo factor captura la curva inicial que se tiene cerca del año 2010 en las series TOTAL_CONSTR, TOTAL_MANUF y MANUF_USA. También captura de mejor manera esas dos situaciones atípicas que se aprecian cerca de 2010 y de 2020.

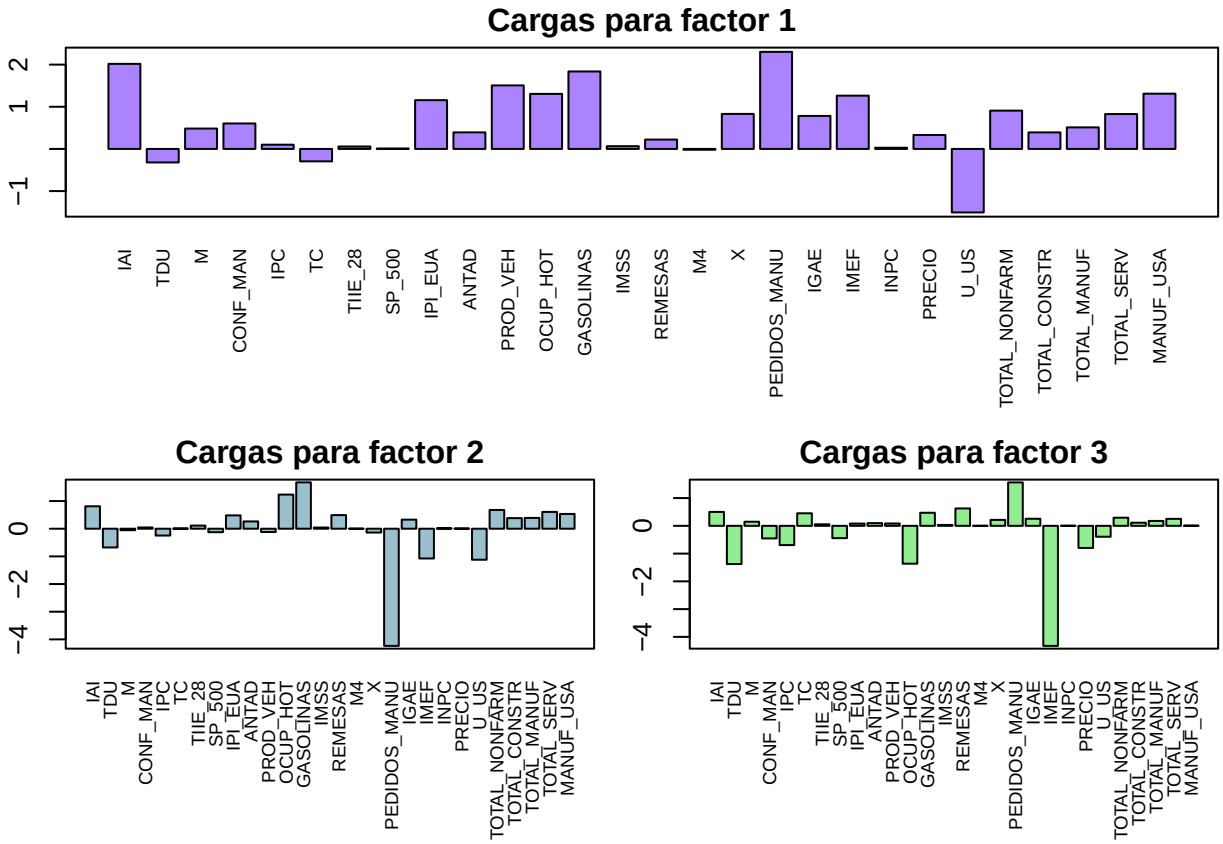
Por último, el tercer factor también captura un poco esas dos situaciones atípicas que se tienen cerca de los años 2010 y 2020. A diferencia de los dos factores anteriores, a partir de 2010 tiene un alza importante y se mantiene constante hasta tener una caída abrupta en el año de 2020.

Método d) Barigozzi

El código implementado en R se muestra enseguida.

```
## Método d) Barigozzi
# Ajustar modelo
pc_dif<-pcfest(diff(tr_sc), r = r)
# Extraer las cargas
Phat_d<-pc_dif$lambda
# Colocar los nombres de columna a los renglones de las cargas
rownames(Phat_d)<-colnames(tr_sc)
# Hacerlo serie de tiempo
fhat<-tr_sc %*% Phat_d/N
fhat_d<-ts(scale(fhat), start = c(2005, 1), frequency = 12)
```

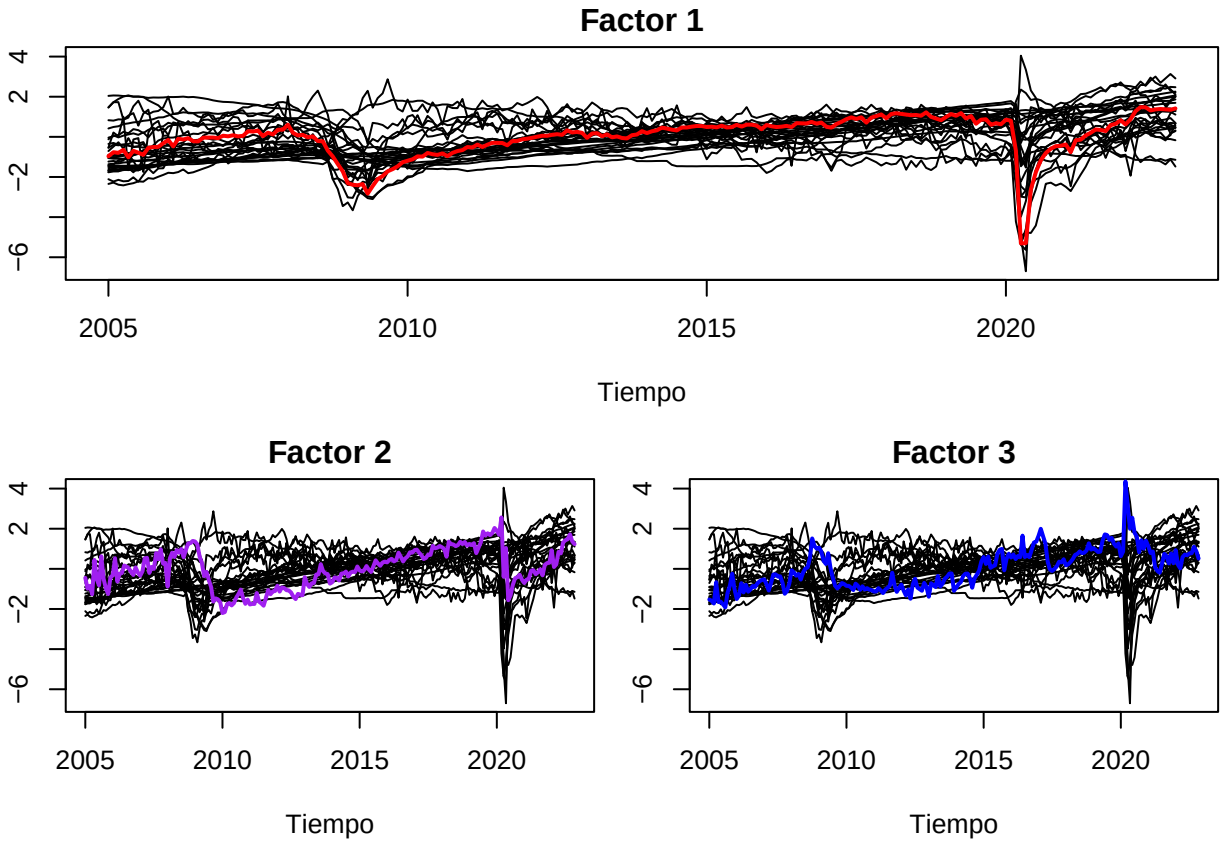
```r
# Gráfico de las cargas estimadas
par(mar = c(6.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
barplot(Phat_d[, 1], las = 3, cex.names = 0.75, col = "mediumpurple1",
        main = "Cargas para factor 1", ylim = c(min(Phat_d[, 1]) - 0.1,
                                                max(Phat_d[, 1]) + 0.1))
box()
barplot(Phat_d[, 2], las = 3, cex.names = 0.75, col = "lightblue3",
        main = "Cargas para factor 2", ylim = c(min(Phat_d[, 2]) - 0.1,
                                                max(Phat_d[, 2]) + 0.1))
box()
barplot(Phat_d[, 3], las = 3, cex.names = 0.75, col = "lightgreen",
        main = "Cargas para factor 3", ylim = c(min(Phat_d[, 3]) - 0.1,
                                                max(Phat_d[, 3]) + 0.1))
box()
```



Las gráficas previas contienen las cargas estimadas de los factores a través del método de estimación de *Barigozzi*. En él se puede observar que, para el primer factor las series que lo dominan de forma positiva son IAI, PEDIDOS_MANU, GASOLINAS, PROD_VEH, OCUP_HOT, MANUF_USA e IPI_EUA, en cambio, de manera negativa se tiene a U_US.

En lo que respecta al segundo factor, es dominado de manera positiva por las series OCUP_HOT y GASOLINAS, por el contrario, de manera negativa se tiene a PEDIDOS_MANU, U_US e IMEF. Por último, el tercer factor está dominado de forma positiva por la serie PEDIDOS_MANU, en cambio, de forma negativa se tiene a IMEF, TDU y OCUP_HOT, .

```r
# Gráfico para los factores
par(mar = c(4.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
ts.plot(tr_sc, col = "black", main = "Factor 1",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_d[, 1], col = "red", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 2",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_d[, 2], col = "purple", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 3",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_d[, 3], col = "blue", lwd = 2)
```



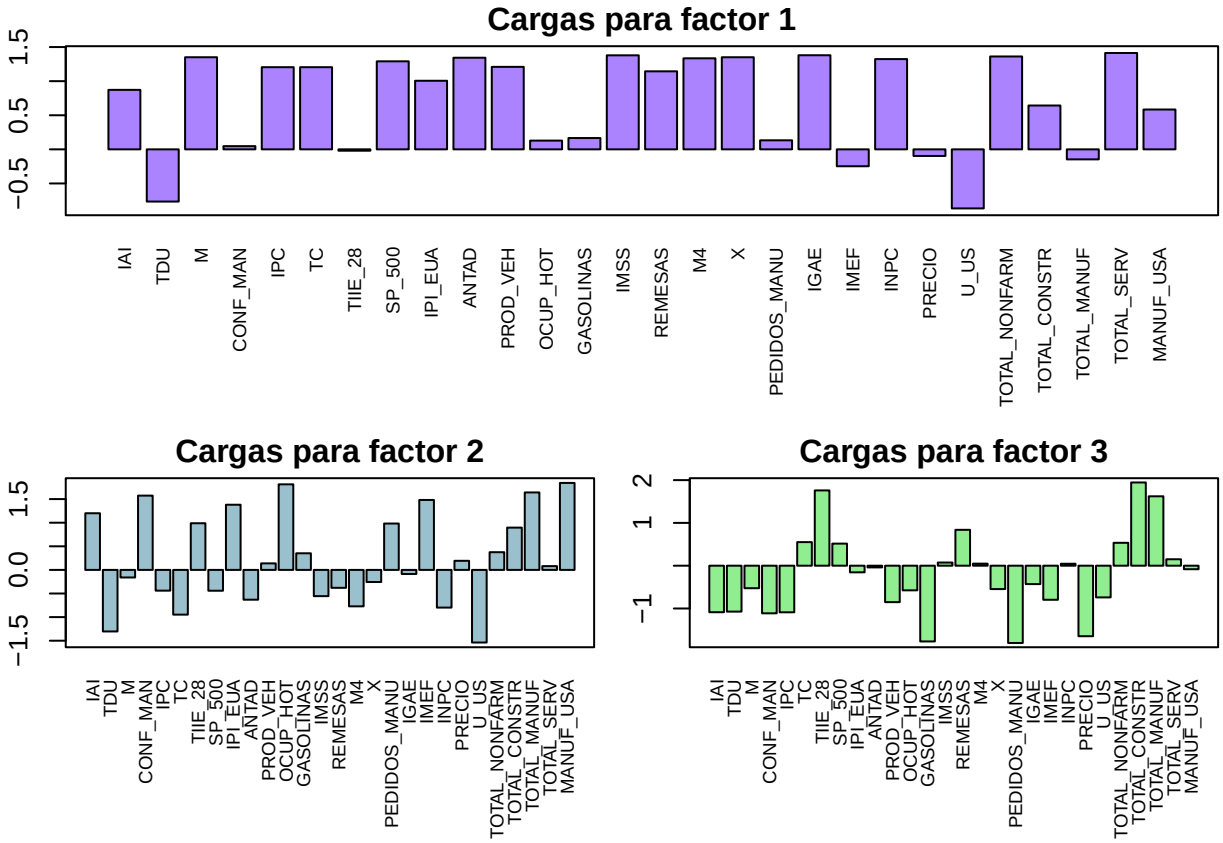
Con base a la salida anterior, el primer factor captura la curva inicial que se tiene cerca del año 2010 en las series TOTAL_CONSTR, TOTAL_MANUF y MANUF_USA. También captura de mejor manera esas dos situaciones atípicas que se aprecian cerca de 2010 y de 2020. Por otra parte, el factor 2 al estar dominado principalmente por la serie de PEDIDOS_MANUF tiende a replicar su comportamiento. De igual forma el factor 3 tiende a replicar el comportamiento, pero de la serie IMEF.

Método e) Factor dinámico (suavizado de Kalman en 2 pasos)

El código utilizado en R es el siguiente.

```r
## Método e) Factor dinámico (Suavizado de Kalman en 2 pasos)
# Ajustar modelo
modelo_fks<-pckf(tr_sc, r)
```

```r
# Hacerlo serie de tiempo
fhat_e<-ts(scale(modelo_fks$Fkf), start = c(2005, 1), frequency = 12)
# Extraer las cargas
Phat_e<-modelo_fks$C
# Colocar los nombres de columna a los renglones de las cargas
rownames(Phat_e)<-colnames(tr_sc)

# Gráfico de las cargas estimadas
par(mar = c(6.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
barplot(Phat_e[, 1], las = 3, cex.names = 0.75, col = "mediumpurple1",
        main = "Cargas para factor 1", ylim = c(min(Phat_e[, 1]) - 0.1,
                                                max(Phat_e[, 1]) + 0.1)); box();
barplot(Phat_e[, 2], las = 3, cex.names = 0.75, col = "lightblue3",
        main = "Cargas para factor 2", ylim = c(min(Phat_e[, 2]) - 0.1,
                                                max(Phat_e[, 2]) + 0.1)); box();
barplot(Phat_e[, 3], las = 3, cex.names = 0.75, col = "lightgreen",
        main = "Cargas para factor 3", ylim = c(min(Phat_e[, 3]) - 0.1,
                                                max(Phat_e[, 3]) + 0.1)); box()
```

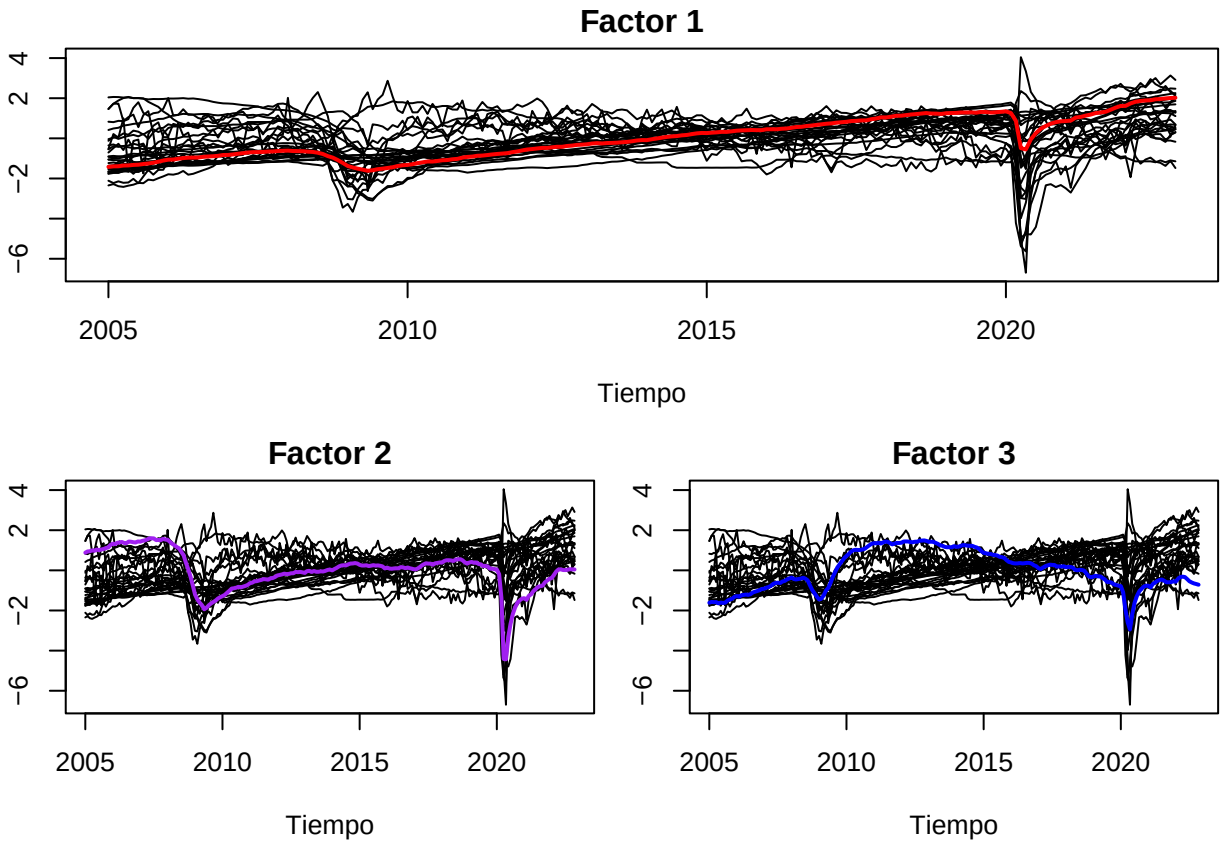


La salida anterior muestra las cargas estimadas de los factores por el método del factor dinámico. En él se puede apreciar que, el primer factor es una combinación de la gran mayoría de las series, excepto por: CONF_MAN, TIIE_28, OCUP_HOT, GASOLINAS, PEDIDOS_MANU, IMEF, PRECIO y TOTAL_MANUF, ya que sus cargas estimadas son cercanas a 0. Las series TDU y U_US dominan este factor de manera negativa, en cambio, el resto de las series con cargas mayores a 0 lo dominan de forma positiva.

En cuanto al segundo factor, lo dominan principalmente de forma positiva las series OCUP_HOT, MANUF_USA, CONF_MAN, TOTAL_MANUF, IMEF, IPI_EUA, TIIE_28, PEDIDOS_MANU y TOTAL_CONSTR, por el contrario, de manera negativa se tiene principalmente a TDU, U_US, TC, M4, INPC y ANTAD.

En lo que respecta al factor 3, las series que lo dominan de forma positiva son TOTAL_CONSTR, TOTAL_MANUF, TIIE_28 y REMESAS, en sentido opuesto se tiene principalmente a PRECIO, PEDIDOS_MANU, GASOLINAS, IAI, TDU, CONF_MAN e IPC.

```
# Gráfico para los factores
par(mar = c(4.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
ts.plot(tr_sc, col = "black", main = "Factor 1",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_e[, 1], col = "red", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 2",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_e[, 2], col = "purple", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 3",
        xlab = "Tiempo", ylab = "Valor")
lines(-fhat_e[, 3], col = "blue", lwd = 2)
```



De acuerdo con la gráfica previa, el primer factor trata de capturar la tendencia de 2010 a 2020 que se percibe en algunas series, por ejemplo, en M, SP500 e IGAE se puede distinguir que después de la caída que se tiene en 2010, la serie vuelve a subir hasta caer nuevamente en aproximadamente 2020, situación que está replicando este factor; también trata de capturar un poco esas dos situaciones atípicas.

El segundo factor captura la curva inicial que se tiene cerca del año 2010 en las series TOTAL_CONSTR, TOTAL_MANUF y MANUF_USA. También captura de mejor manera esas dos situaciones atípicas que se aprecian cerca de 2010 y de 2020.

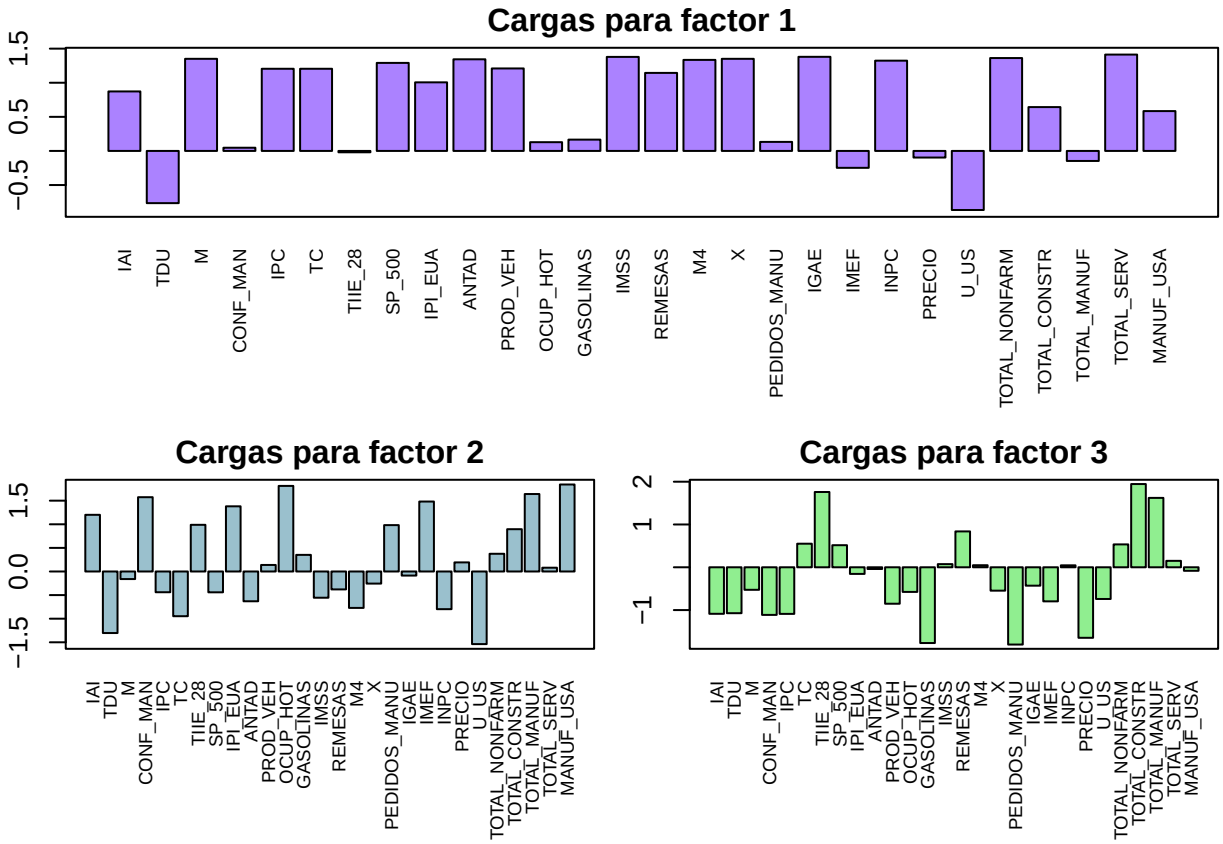
Por último, el tercer factor también captura un poco esas dos situaciones atípicas que se tienen cerca de los años 2010 y 2020. A diferencia de los dos factores anteriores, a partir de 2010 tiene un alza importante y después una caída hasta llegar al año de 2020.

7.4 Elección del modelo de factores dinámicos

De acuerdo con las estimaciones realizadas con los 5 métodos, en todos ellos el primer factor resulta ser una combinación de la mayoría de las series y parece ser de orden de integración uno, el resto de los factores también parece tener un orden de integración de 1. Cuando los factores no son estacionarios una de las sugerencias es utilizar el método del factor dinámico. Por lo tanto, se utilizará este método para los análisis posteriores.

El código utilizado en R para la estimación de los factores es el siguiente.

```
## Método e) Factor dinámico (Suavizado de Kalman en 2 pasos)
# Ajustar modelo
modelo_fks<-pckf(tr_sc, r)
# Hacerlo serie de tiempo
fhat_e<-ts(scale(modelo_fks$Fkf), start = c(2005, 1), frequency = 12)
# Extraer las cargas
Phat_e<-modelo_fks$C
# Colocar los nombres de columna a los renglones de las cargas
rownames(Phat_e)<-colnames(tr_sc)

# Gráfico de las cargas estimadas
par(mar = c(6.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
barplot(Phat_e[, 1], las = 3, cex.names = 0.75, col = "mediumpurple1",
        main = "Cargas para factor 1", ylim = c(min(Phat_e[, 1]) - 0.1,
                                                max(Phat_e[, 1]) + 0.1)); box();
barplot(Phat_e[, 2], las = 3, cex.names = 0.75, col = "lightblue3",
        main = "Cargas para factor 2", ylim = c(min(Phat_e[, 2]) - 0.1,
                                                max(Phat_e[, 2]) + 0.1)); box();
barplot(Phat_e[, 3], las = 3, cex.names = 0.75, col = "lightgreen",
        main = "Cargas para factor 3", ylim = c(min(Phat_e[, 3]) - 0.1,
                                                max(Phat_e[, 3]) + 0.1)); box()
```



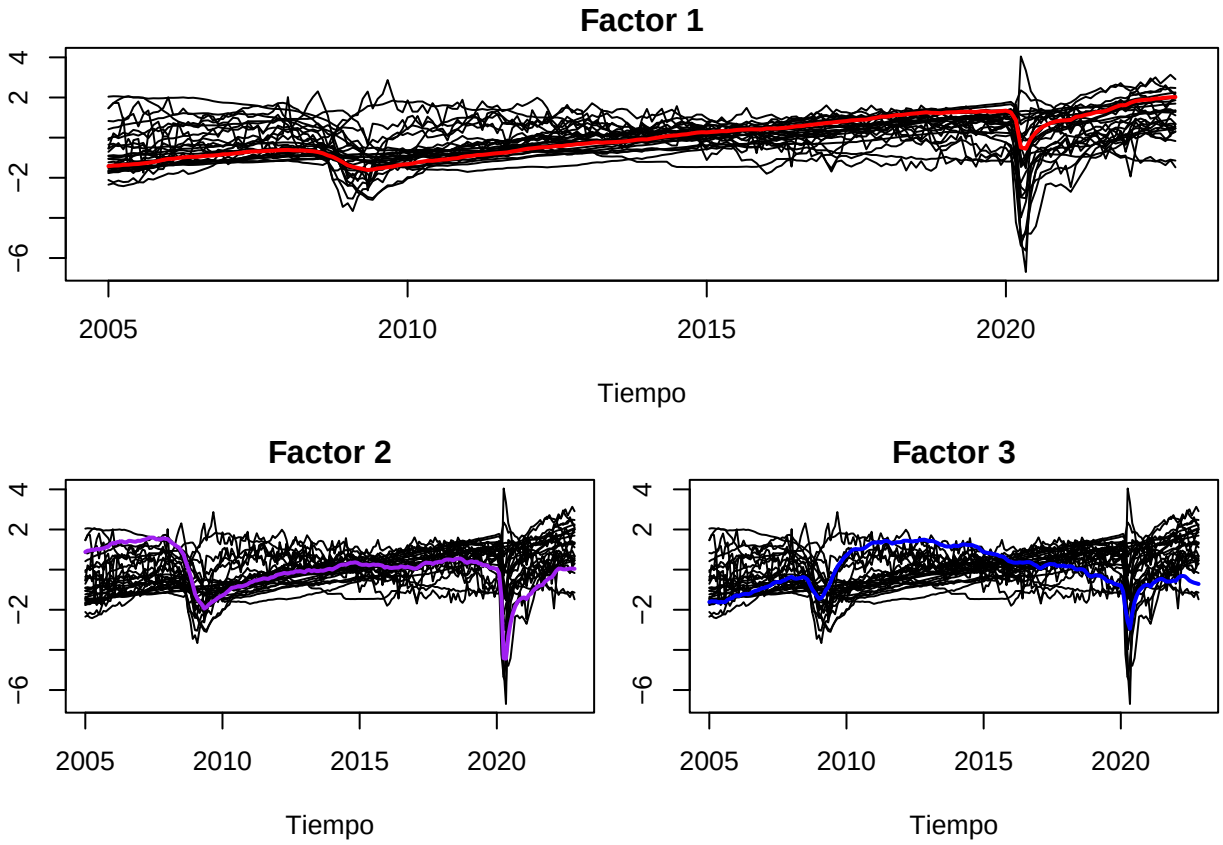
La salida anterior muestra las cargas estimadas de los factores por el método del factor dinámico. En él se puede apreciar que, el primer factor es una combinación de la gran mayoría de las series, excepto por: CONF_MAN, TIIE_28, OCUP_HOT, GASOLINAS, PEDIDOS_MANU, IMEF, PRECIO y TOTAL_MANUF, ya que sus cargas estimadas son cercanas a 0. Las series TDU y U_US dominan este factor de manera negativa, en cambio, el resto de las series con cargas mayores a 0 lo dominan de forma positiva.

En cuanto al segundo factor, lo dominan principalmente de forma positiva las series OCUP_HOT, MANUF_USA, CONF_MAN, TOTAL_MANUF, IMEF, IPI_EUA, TIIE_28, PEDIDOS_MANU y TOTAL_CONSTR, por el contrario, de manera negativa se tiene principalmente a TDU, U_US, TC, M4, INPC y ANTAD.

En lo que respecta al factor 3, las series que lo dominan de forma positiva son TOTAL_CONSTR, TOTAL_MANUF, TIIE_28 y REMESAS, en sentido opuesto se tiene principalmente a PRECIO, PEDIDOS_MANU, GASOLINAS, IAI, TDU, CONF_MAN e IPC.

```
# Gráfico para los factores
par(mar = c(4.5, 2, 1.75, 1))
layout(matrix(c(1, 1, 2, 3), 2, 2, byrow = TRUE))
ts.plot(tr_sc, col = "black", main = "Factor 1",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_e[, 1], col = "red", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 2",
        xlab = "Tiempo", ylab = "Valor")
lines(fhat_e[, 2], col = "purple", lwd = 2)
ts.plot(tr_sc, col = "black", main = "Factor 3",
        xlab = "Tiempo", ylab = "Valor")
```

```
lines(-fhat_e[, 3], col = "blue", lwd = 2)
```



De acuerdo con la gráfica previa, el primer factor trata de capturar la tendencia de 2010 a 2020 que se percibe en algunas series, por ejemplo, en M, SP500 e IGAE se puede distinguir que después de la caída que se tiene en 2010, la serie vuelve a subir hasta caer nuevamente en aproximadamente 2020, situación que está replicando este factor; también trata de capturar un poco esas dos situaciones atípicas.

El segundo factor captura la curva inicial que se tiene cerca del año 2010 en las series TOTAL_CONSTR, TOTAL_MANUF y MANUF_USA. También captura de mejor manera esas dos situaciones atípicas que se aprecian cerca de 2010 y de 2020. Por último, el tercer factor también captura un poco esas dos situaciones atípicas que se tienen cerca de los años 2010 y 2020. A diferencia de los dos factores anteriores, a partir de 2010 tiene un alza importante y después una caída hasta llegar al año de 2020.

Posteriormente, se realizó la prueba aumentada de *Dickey Fuller* con la intención de determinar el orden de integración de los 3 factores estimados. Las hipótesis de la prueba son las siguientes:

H_o : El factor es no estacionario
H_a : El factor es estacionario

La regla de decisión es rechazar la hipótesis nula (H_o) si el valor p es menor que un nivel de significancia de 0.05 ($\alpha = 0.05$). La salida de los valores p de la prueba, tanto para el factor original como diferenciado se muestra a continuación.

```
# Matriz para pruebas ADF
adf_tests<-matrix(NA, ncol(fhat_e), 2)
# Nombres de columnas y renglones
colnames(adf_tests)<-c("Orginal", "Diferenciada")
rownames(adf_tests)<-c(paste0("F", 1:3))
```

```
# Ciclo for para las pruebas
for(i in 1:ncol(fhat_e)){
adf_tests[i, 1]<-adf(fhat_e[, i], "const")$p.value
adf_tests[i, 2]<-adf(diff(fhat_e[, i]), "const")$p.value }

# Se imprime el resultado de las pruebas ADF
kbl(adf_tests, longtable = T, booktabs = T,
    caption = "Resultado de la prueba ADF",
    col.names = c("Original", "Diferenciada"),
    escape = TRUE, digits = 3)
```

Cuadro 3: Resultado de la prueba ADF

	Original	Diferenciada
F1	0.914	0.01
F2	0.138	0.01
F3	0.360	0.01

La tabla anterior contiene el resultado de los valores p de la prueba *ADF* tanto para el factor original como diferenciado. En ella se puede observar que los factores no son estacionarios originalmente, pero sí lo son en las primeras diferencias, por lo que, los tres factores estimados son de orden de integración 1.

Adicionalmente, se aplicó la prueba *panic.f*, función desarrollada por el Dr. Francisco de Jesús Corona. Esta prueba retorna *m number of non-stationary common factors* (m número de factores comunes no estacionarios). El resultado se presenta a continuación.

```
# Número de factores comunes no estacionarios
panic.f(fhat_e)
```

```
# [1] 3
```

Con base a la salida anterior, para el modelo de factores dinámicos ajustado por el método del *Factor dinámico*, el número de factores comunes no estacionarios es de 3.

Después, se analizaron los factores para revisar si capturaron algún tipo de estacionalidad. El método visto en clase consiste en incluir variables ficticias estacionales y comprobar si tienen valores p significativos al calcular la regresión. Si los meses individuales tienen coeficientes significativos, el factor tiene problemas de estacionalidad. El código implementado se muestra enseguida.

```
# Modelo para determinar estacionalidad
for(i in 1:ncol(fhat_e)){
estacionalidad<-lm(fhat_e[, i] ~ c(1:nrow(fhat_e)) + seasonaldummy(fhat_e[, i]))
# Resumen del modelo
print(paste0("F", 1:3)[i])
print(summary(estacionalidad)) }
```

```
# [1] "F1"
#
# Call:
# lm(formula = fhat_e[, i] ~ c(1:nrow(fhat_e)) + seasonaldummy(fhat_e[,
#     i]))
```

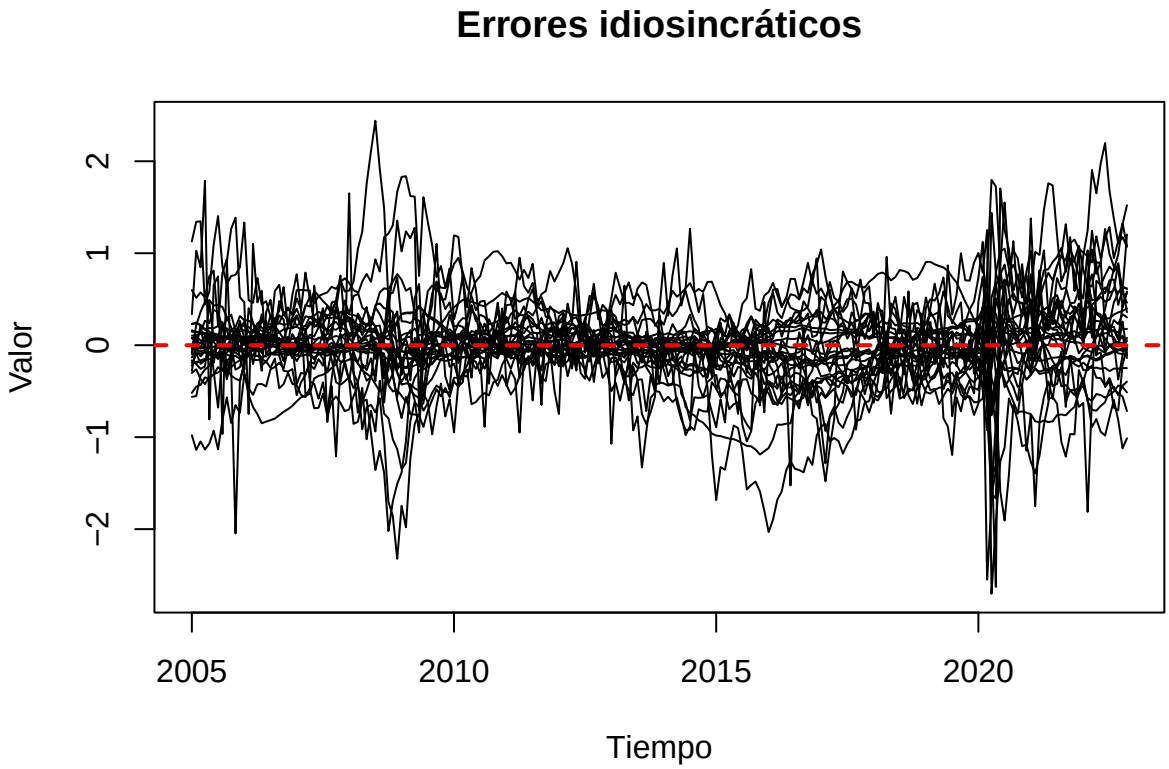
```
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -1.63395 -0.19929  0.09221  0.31479  0.52557
#
# Coefficients:
#                                Estimate Std. Error t value Pr(>|t|)
# (Intercept)                   -1.5810167  0.1071681 -14.753   <2e-16 ***
# c(1:nrow(fhat_e))              0.0148184  0.0004366  33.944   <2e-16 ***
# seasonaldummy(fhat_e[, i])Jan  0.0027749  0.1342178   0.021    0.984
# seasonaldummy(fhat_e[, i])Feb  0.0029866  0.1342114   0.022    0.982
# seasonaldummy(fhat_e[, i])Mar -0.0040470  0.1342064  -0.030    0.976
# seasonaldummy(fhat_e[, i])Apr -0.0770740  0.1342028  -0.574    0.566
# seasonaldummy(fhat_e[, i])May -0.0809292  0.1342007  -0.603    0.547
# seasonaldummy(fhat_e[, i])Jun -0.0484829  0.1342000  -0.361    0.718
# seasonaldummy(fhat_e[, i])Jul -0.0273384  0.1342007  -0.204    0.839
# seasonaldummy(fhat_e[, i])Aug -0.0125510  0.1342028  -0.094    0.926
# seasonaldummy(fhat_e[, i])Sep -0.0065862  0.1342064  -0.049    0.961
# seasonaldummy(fhat_e[, i])Oct  0.0044166  0.1342114   0.033    0.974
# seasonaldummy(fhat_e[, i])Nov  0.0155251  0.1342178   0.116    0.908
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 0.3968 on 202 degrees of freedom
# Multiple R-squared:  0.8514,  Adjusted R-squared:  0.8425
# F-statistic: 96.43 on 12 and 202 DF,  p-value: < 2.2e-16
#
# [1] "F2"
#
# Call:
# lm(formula = fhat_e[, i] ~ c(1:nrow(fhat_e)) + seasonaldummy(fhat_e[,
#     i]))
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -3.8128 -0.4015  0.2988  0.6695  1.2122
#
# Coefficients:
#                                Estimate Std. Error t value Pr(>|t|)
# (Intercept)                    0.722214   0.254310   2.840  0.00497 **
# c(1:nrow(fhat_e))             -0.006420   0.001036  -6.197 3.17e-09 ***
# seasonaldummy(fhat_e[, i])Jan  0.002808   0.318499   0.009  0.99297
# seasonaldummy(fhat_e[, i])Feb  0.003885   0.318484   0.012  0.99028
# seasonaldummy(fhat_e[, i])Mar -0.026274   0.318472  -0.082  0.93433
# seasonaldummy(fhat_e[, i])Apr -0.167095   0.318464  -0.525  0.60037
# seasonaldummy(fhat_e[, i])May -0.156726   0.318458  -0.492  0.62316
# seasonaldummy(fhat_e[, i])Jun -0.081266   0.318457  -0.255  0.79884
# seasonaldummy(fhat_e[, i])Jul -0.028660   0.318458  -0.090  0.92838
# seasonaldummy(fhat_e[, i])Aug  0.009963   0.318464   0.031  0.97507
# seasonaldummy(fhat_e[, i])Sep  0.018884   0.318472   0.059  0.95278
# seasonaldummy(fhat_e[, i])Oct  0.036744   0.318484   0.115  0.90826
# seasonaldummy(fhat_e[, i])Nov  0.043378   0.318499   0.136  0.89180
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#
# Residual standard error: 0.9416 on 202 degrees of freedom
# Multiple R-squared:  0.1631,  Adjusted R-squared:  0.1133
# F-statistic:  3.28 on 12 and 202 DF,  p-value: 0.0002381
#
# [1] "F3"
#
# Call:
# lm(formula = fhat_e[, i] ~ c(1:nrow(fhat_e)) + seasonaldummy(fhat_e[,
#     i]))
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -1.50140 -1.03457  0.05467  0.75026  2.93530
#
# Coefficients:
#                                 Estimate Std. Error t value Pr(>|t|)
# (Intercept)                   -0.0048132  0.2774653  -0.017    0.986
# c(1:nrow(fhat_e))             -0.0006893  0.0011303  -0.610    0.543
# seasonaldummy(fhat_e[, i])Jan  0.1023170  0.3474986   0.294    0.769
# seasonaldummy(fhat_e[, i])Feb  0.1098972  0.3474820   0.316    0.752
# seasonaldummy(fhat_e[, i])Mar  0.1195815  0.3474692   0.344    0.731
# seasonaldummy(fhat_e[, i])Apr  0.1289738  0.3474600   0.371    0.711
# seasonaldummy(fhat_e[, i])May  0.1454188  0.3474545   0.419    0.676
# seasonaldummy(fhat_e[, i])Jun  0.0954056  0.3474526   0.275    0.784
# seasonaldummy(fhat_e[, i])Jul  0.0620112  0.3474545   0.178    0.859
# seasonaldummy(fhat_e[, i])Aug  0.0468075  0.3474600   0.135    0.893
# seasonaldummy(fhat_e[, i])Sep  0.0411116  0.3474692   0.118    0.906
# seasonaldummy(fhat_e[, i])Oct  0.0452264  0.3474820   0.130    0.897
# seasonaldummy(fhat_e[, i])Nov  0.0499873  0.3474986   0.144    0.886
#
# Residual standard error: 1.027 on 202 degrees of freedom
# Multiple R-squared:  0.003718,    Adjusted R-squared:  -0.05547
# F-statistic: 0.06282 on 12 and 202 DF,  p-value: 1
```

Con base a las salidas anteriores, las variables ficticias estacionales ajustadas a cada uno de los factores no resultaron significativos en los modelos de regresión, por lo que los factores no capturaron ningún tipo de estacionalidad.

Posteriormente, se realizó la prueba de estacionariedad a los errores idiosincráticos, pero, antes de comenzar con la prueba formal, se generó el gráfico de los errores el cual debería verse como series estacionarias. El resultado se muestra a continuación.

```
## Probar si los errores idosincráticos son I(0)
# Ajustar modelo
modelo_fks<-pckf(tr_sc, r)
# Extraer el factor
fhat_e<-modelo_fks$Fkf
# Extraer las cargas
Phat_e<-modelo_fks$C
# Cálculo de los errores
ehat<-tr_sc - fhat_e%*%t(Phat_e)
# Prueba visual se debe ver estacionario
ts.plot(ehat, main = "Errores idiosincráticos", xlab = "Tiempo", ylab = "Valor")
abline(h = 0, col = "red", lty = 2, lwd = 2)
```



De acuerdo con la gráfica previa, la gran mayoría de las series de los errores tienen un comportamiento de una serie estacionaria a simple vista, no obstante, se realizó la prueba para la estacionariedad a los errores idiosincráticos, cuyas hipótesis son las siguientes:

H_o : Existe raíz unitaria múltiple en los errores idiosincráticos
H_a : No existe raíz unitaria múltiple en los errores idiosincráticos

La regla de decisión es rechazar H_o si el valor p es menor que 0.05 ($\alpha = 0.05$). El resultado de la prueba se presenta enseguida.

```
# Prueba de estacionariedad a los errores idiosincráticos
pooled.test(ehat)
```

```
#    Pest p.value
#  6.5016  0.0000
```

De acuerdo con la salida anterior, el valor p es de 0, por lo que se rechaza H_o, es decir, existe suficiente evidencia para determinar que no existe raíz unitaria múltiple en los errores idiosincráticos, es decir, que e_t ~ I(0).

7.5 Pronósticos para el INPC

Una vez que se ha revisado que los errores idiosincráticos son I(0), el siguiente paso es generar pronósticos para la variable del INPC con el MFD ajustado. Pero antes de eso, resulta relevante analizar la correlación entre la variable de interés con los 3 factores del MFD.

```
# Matriz de correlaciones de las variables con los factores
cor(tr_sc[, "INPC"], fhat_e)
```

```
#           [,1]       [,2]      [,3]
# [1,] 0.9238736 -0.3832972 0.0267896
```

De acuerdo con la salida anterior, la correlación lineal entre la variable INPC con el factor 1 es de 0.9238736, con el factor 2 es de -0.3832972 y con el factor 3 es de 0.0267896, por lo que, la variable INPC tiene la mayor correlación con el factor 1.

Posteriormente, se realizaron 12 pronósticos para los últimos 12 meses disponibles de la tabla de datos. Para ello se ajustó un modelo VAR, para realizar pronósticos un paso hacia adelante con un horizonte 12 de longitud. Es importante mencionar que, como criterio de información para la identificación del número de rezagos fue seleccionado el criterio de *Hannan - Quinn* (HQ). Asimismo, en cada paso hacia adelante se actualiza el pronóstico. El código implementado se presenta a continuación.

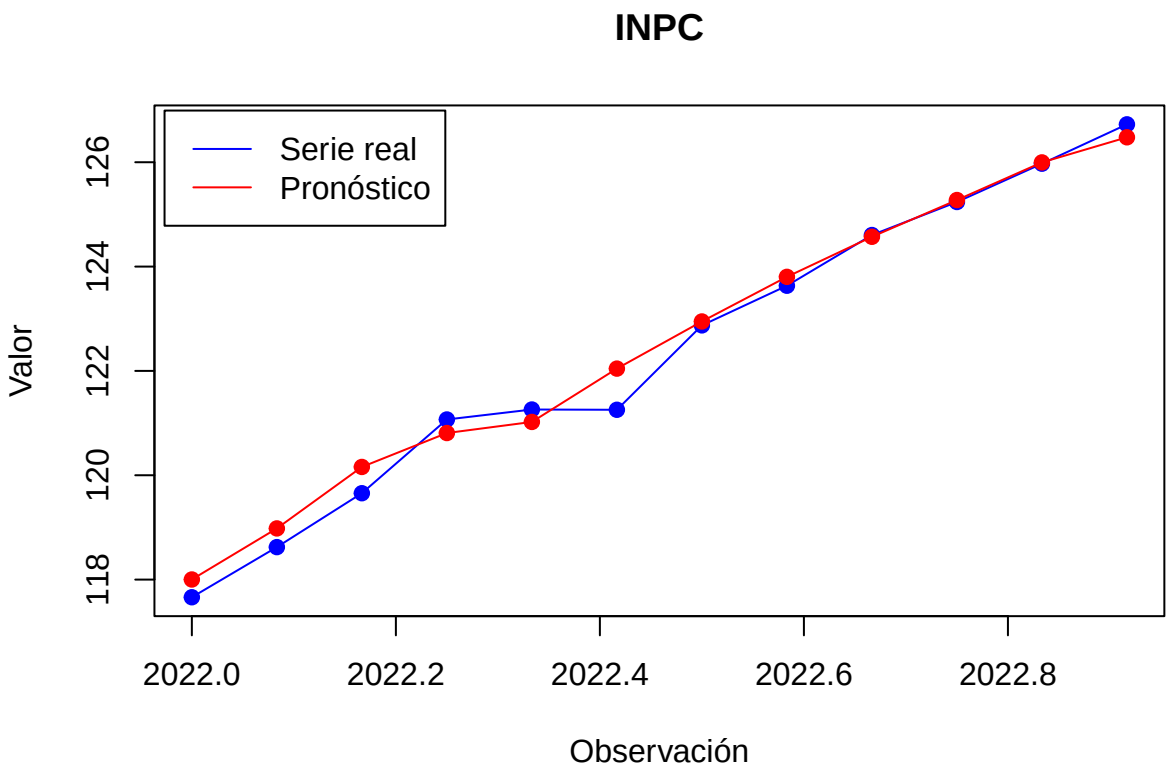
```
# Unir la serie de interés con los factores
Xt<-cbind(tr_sc[, "INPC"], fhat_e)
# Agregar los nombres al conjunto Xt
colnames(Xt)<-c("INPC", paste0("F", 1:r))

# Tamaño de la serie
n<-nrow(Xt)
# Número de pronósticos
H<-11
# Se extraen los datos que serán los observados
observado<-datos$INPC[205:216]
# Tabla resumen de la salida de R
resumen<-data.frame(OBSERVADO = observado, PRONOSTICO = rep(0, H + 1),
                    ERROR = rep(0, H + 1))

# Se crea el ciclo for para los pronósticos
for(h in 1:H){

# Se extraen los datos de interés de la serie
serie<-ts(Xt[1:(n - H - 1 + h), ], start = c(2005, 1), frequency = 12)
# Determinar el número de rezagos
p<-VARselect(serie)$selection["HQ(n)"]
# Ajuste del modelo VAR
modelo<-VAR(serie, p = p)
# Se realiza el pronóstico y se extrae la estimación puntual
pronostico<-forecast(modelo, h = 1)$forecast$INPC$mean
# Se guarda el resultado del pronóstico en la tabla resumen
resumen[h, "PRONOSTICO"]<-pronostico

} # Del ciclo for

## Pronóstico para el mes 12
```

```
serie<-ts(Xt, start = c(2005, 1), frequency = 12)
# Determinar el número de rezagos
p<-VARselect(serie)$selection["HQ(n)"]
# Ajuste del modelo VAR
modelo<-VAR(serie, p = p)
# Se realiza el pronóstico y se extrae la estimación puntual
pronostico<-forecast(modelo, h = 1)$forecast$INPC$mean
# Se guarda el resultado del pronóstico en la tabla resumen
resumen[12, "PRONOSTICO"]<-pronostico

# Calcular la media y la desviación estándar de las variables originales
promedio<-colMeans(tr)
desviacion<-apply(tr, 2, "sd")

# Retornar los valores estandarizados a la serie original
resumen[, "PRONOSTICO"]<-resumen$PRONOSTICO*desviacion["INPC"] + promedio["INPC"]
resumen[, "ERROR"]<-resumen$OBSERVADO - resumen$PRONOSTICO

# Imprimir los resultados
resumen
```

```
#    OBSERVADO PRONOSTICO       ERROR
# 1    118.002   117.6616  0.34039321
# 2    118.981   118.6226  0.35844970
# 3    120.159   119.6555  0.50351697
# 4    120.809   121.0690 -0.25995853
# 5    121.022   121.2601 -0.23807002
# 6    122.044   121.2547  0.78928068
# 7    122.948   122.8730  0.07502108
# 8    123.803   123.6327  0.17026308
# 9    124.571   124.6045 -0.03346842
# 10   125.276   125.2379  0.03814886
# 11   125.997   125.9719  0.02507222
# 12   126.478   126.7263 -0.24830518
```

Para apreciar de mejor manera los resultados obtenidos, a continuación, se presenta la siguiente gráfica temporal.

```
# Transformar los pronósticos a serie de tiempo
observado_ts<-ts(resumen$OBSERVADO, start = c(2022, 1), frequency = 12)
pronostico_ts<-ts(resumen$PRONOSTICO, start = c(2022, 1), frequency = 12)

# Gráfico temporal
plot(pronostico_ts, xlab = "Observación", ylab = "Valor", type = "o", pch = 19,
     col = "blue", main = "INPC")
# Datos pronosticados
points(observado_ts, col = "red", type = "o", pch = 19)
# Leyenda del gráfico
legend("topleft", inset = 0.01, legend = c("Serie real", "Pronóstico"),
       lty = c(1, 1), col = c("blue", "red"))
```



La salida anterior muestra la representación gráfica de los datos observados (línea azul) vs los datos pronosticados (línea roja) para un horizonte H de longitud 12. Como puede apreciarse, los valores pronosticados son bastante similares a los observados, siendo los más parecidos los últimos 5 meses.

Para fines comparativos de los modelos ajustados, se calculó el error de predicción a través del error cuadrático medio (MSE por sus siglas en inglés).

```
# Se realizan los cálculos del MSE
MSE_MFD<-sum(resumen$ERROR^2)/12
MSE_MFD
```

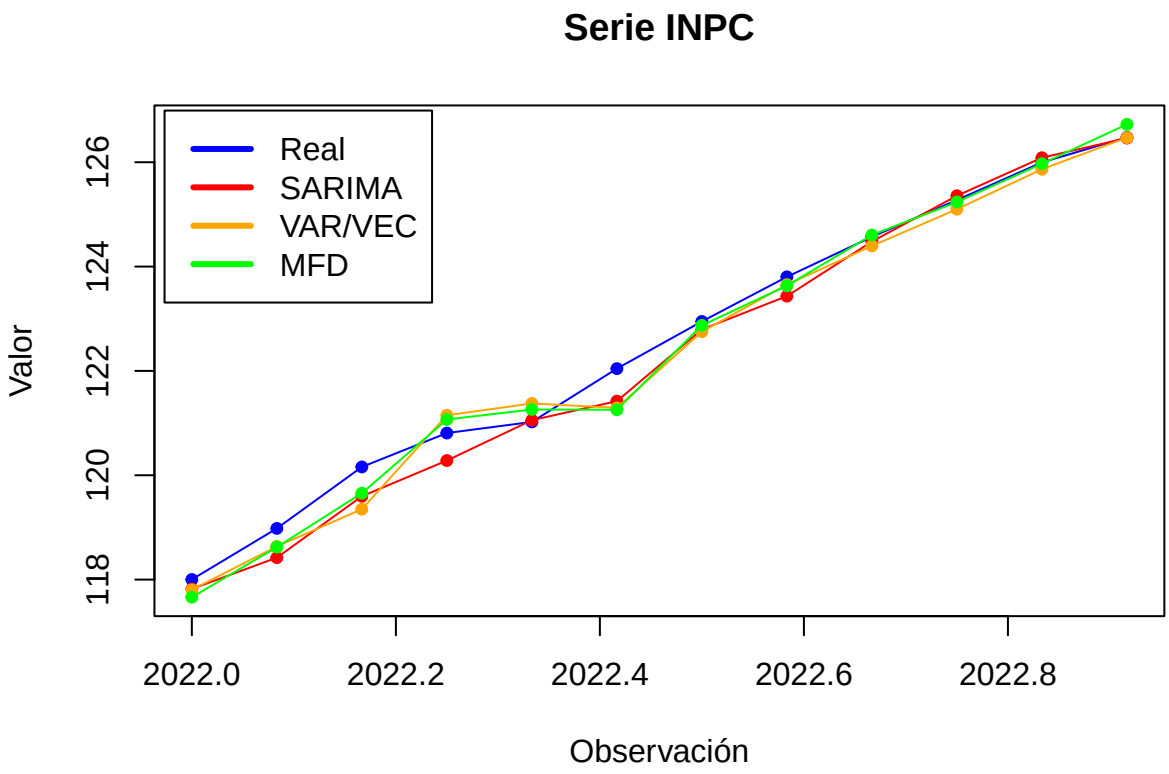
```
# [1] 0.1120483
```

De acuerdo con la salida previa, el valor del *MSE* para el Modelo de Factores Dinámicos ajustado para realizar pronósticos del INPC, fue de 0.1120483

8. Conclusiones y trabajos futuros

La representación visual de los pronósticos obtenidos para el INPC mediante el Modelo SARIMA, VAR/VEC y de Factores Dinámicos se presenta en la próxima gráfica.

```
#################################
# Conclusiones
#################################

# Se limpia el espacio de trabajo
rm(list = ls())
invisible(gc(reset = TRUE))

# Se cargan las librerías
library(readxl)   # Leer archivos xlsx

# Establecer dirección de trabajo
ruta<-("C:/Proyecto de series/")
# Se cargan las funciones
source(paste(ruta, "functions.r", sep = ""))

# Leer datos
datos<-read_excel(paste0(ruta, "Pronósticos INPC.xlsx"), sheet = "Resumen")
# Se convierten a series de tiempo
tr<-ts(as.data.frame(datos[, -1]), start = c(2022, 1), frequency = 12)

# Gráfico temporal
plot(tr[, 1], xlab = "Observación", ylab = "Valor", type = "o", pch = 19,
     col = "blue", main = "Serie INPC", ylim = c(min(tr), max(tr)), cex = 0.75)
# Datos pronosticados
points(tr[, 2], col = "red", type = "o", pch = 19, cex = 0.75)
points(tr[, 3], col = "orange", type = "o", pch = 19, cex = 0.75)
points(tr[, 4], col = "green", type = "o", pch = 19, cex = 0.75)
# Leyenda del gráfico
legend("topleft", inset = 0.01, legend = c("Real", "SARIMA", "VAR/VEC", "MFD"),
       lty = c(1), col = c("blue", "red", "orange", "green"), lwd = 3)
```



De acuerdo con la gráfica previa, visualmente los pronósticos difieren muy poco de la serie real del INPC, la cual está representada con color azul. No obstante, la serie que más se asemeja a la real es la generada por el Modelo de Factores Dinámicos (MFD), después le seguiría la obtenida por el modelo SARIMA y por último la realizada con el modelo VAR/VEC, aunque este último se sugiere tener precaución en los pronósticos, ya que fue el único modelo en el que no se cumplieron los supuestos, por lo que las interpretaciones realizadas deben tomarse con cautela.

En cuanto a las cifras del *MSE*, el MFD arrojó 0.1120483, le sigue el modelo SARIMA con 0.1262209 y, por último, el modelo VAR/VEC con 0.146699, lo que implica que el MFD obtuvo mejores pronósticos del INPC para el año 2022. Con estos resultados se concluye que, al agregar información valiosa en las series de tiempo se puede obtener una mejor precisión en los pronósticos. No obstante, el uso conjunto de los tres modelos propuestos podría servir como punto de referencia para la comparación de resultados y evaluar la consistencia de los pronósticos.

Como futuras líneas de trabajo, se pretende continuar con la generación de pronósticos del INPC agregando otras series con la intención de mejorar la precisión. Asimismo, implementar los pronósticos en la planeación operativa de las encuestas y censos que realiza el INEGI. Existen otros proyectos en los que se desea implementar las series de tiempo, por ejemplo, en la codificación de encuestas para determinar los avances esperados hasta cierto día, pronósticos para otros indicadores económicos y el número de servicios que se esperarían a lo largo del año en el centro de atención a usuarios del propio Instituto.

9. Referencias

- C. Rella, J. (2020). Modelo Factorial Dinámico para la Economía Gallega. Universidad de Coruña.
- Corona, F. (2023). Modelo de factores dinámicos. CIMAT - INEGI.
- Corona, F. (2023). Vectores Autorregresivos. CIMAT - INEGI.
- Corona, F. (2023). Vectores de Corrección de Error. CIMAT - INEGI.
- Instituto Nacional de Estadística y Geografía (INEGI). (2019). Indicador Oportuno de la Actividad Económica. Síntesis metodológica. INEGI.
- Instituto Nacional de Estadística y Geografía (INEGI). (2023). Preguntas frecuentes del Índice Nacional de Precios al Consumidor (INPC). INEGI.