

# Understanding X13-ARIMA-SEATS models

November 1, 2024

Submitted By

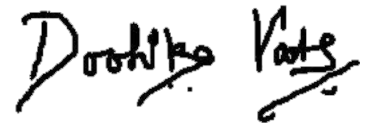

Under the guidance of

**Mr. Rohit Jangid (210872)**  
BS Statistics and Data Science  
Indian Institute of Technology Kanpur

**Prof. Dootika Vats**  
Dept. of Mathematics and Statistics  
Indian Institute of Technology Kanpur

Signature

Signature

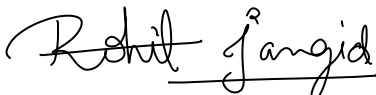


Department of Mathematics and Statistics,  
Indian Institute of Technology, Kanpur

## DECLARATION

I hereby declare that the work presented in the project report entitled 'Understanding X13-ARIMA-SEATS models' is written by me in my own words and contains my own or borrowed ideas. At places, where ideas and words are borrowed from other sources, proper references, and acknowledgments, as applicable, have been provided. To the best of my knowledge, this work does not emanate from or resemble work created by person(s) other than those mentioned and acknowledged herein.

Name: Rohit Jangid (210872)

Signature 

Date: 1 NOV 2024

# 1 Abstract

The Ministry of Statistics and Programme Implementation (MoSPI) in India is responsible for disseminating key economic indicators such as the Gross Domestic Product (GDP) and the Consumer Price Index (CPI). These indicators often exhibit seasonal variations due to factors like holidays and agricultural cycles, which can obscure the true underlying economic trends. Currently, MoSPI does not apply seasonal adjustments to its indices, potentially leading to misinterpretations of economic conditions.

This report addresses the need for effective seasonal adjustment techniques by exploring the **X13-ARIMA-SEATS** model developed by the U.S. Census Bureau. The objectives of this project are threefold: to provide a detailed technical explanation of the **X13-ARIMA-SEATS** model for MoSPI, to assess its appropriateness for Indian economic data, and to equip MoSPI with the tools necessary for correct implementation.

We analyze the monthly combined CPI data of India from January 2013 to October 2024 to demonstrate the application of the **X13-ARIMA-SEATS** model. The report delves into the theoretical foundations of the model, including focusing mainly on the regARIMA framework.

Our findings indicate that although the **X13-ARIMA-SEATS** software is a powerful and suitable tool for the seasonal adjustment but certain changes and adjustments must be made to make it suitable for Indian economic time series data. The project has significantly advanced our understanding of the model's complexities and provided insights into its practical implementation. However, further work is needed to fully comprehend the SEATS algorithm and the X-13 seasonal adjustment method.

In conclusion, this report enhances the understanding of seasonal adjustment techniques and provides a foundation for MoSPI to implement the **X13-ARIMA-SEATS** model. Future work will focus on into the SEATS and X-13 procedures and extending the analysis to additional datasets. This endeavor aims to improve the accuracy and reliability of India's economic indicators, aiding policymakers and stakeholders in making informed decisions.

## Contents

<b>1</b>	<b>Abstract</b>	<b>2</b>
<b>2</b>	<b>Introduction</b>	<b>3</b>
<b>3</b>	<b>X-13ARIMA-SEATS</b>	<b>4</b>
3.1	SEATS and X-13 . . . . .	4
3.2	RegARIMA and TRAMO . . . . .	4
3.2.1	Workflow Explanation . . . . .	5
3.3	Using X13-ARIMA-SEATS . . . . .	6
3.3.1	Spec File and Key Specifications . . . . .	6
3.3.2	Commonly used specs in X13-ARIMA-SEATS . . . . .	7
<b>4</b>	<b>Background Mathematical Details</b>	<b>8</b>
4.1	Time Series Models . . . . .	8
4.2	Time Series Decomposition . . . . .	10
4.3	Ljung-Box Test for Model Residuals . . . . .	11
4.4	The Hannan-Rissanen Estimation Method . . . . .	11
4.5	Information Criteria for Model Selection . . . . .	12
<b>5</b>	<b>RegARIMA</b>	<b>14</b>
5.1	Regression Variables . . . . .	14
5.1.1	Stock and Flow Series . . . . .	15
5.1.2	Trading Day Regression Variables . . . . .	15
5.1.3	Holiday Effect Variables . . . . .	16
5.1.4	Length-of-Month, Length-of-Quarter, and Leap Year Variables . . . . .	17
5.1.5	Outlier Regressors . . . . .	19
5.1.6	User-Defined Regressors and the <code>usertype</code> Argument . . . . .	19

5.1.7	Change of Regime Regressors . . . . .	20
5.1.8	AIC-Based Regressor Selection Procedure . . . . .	21
5.2	Transformation . . . . .	22
5.2.1	Power or Function Transform . . . . .	22
5.2.2	Length of Month (LoM) and Length of Quarter (LoQ) Adjustments . . . . .	24
5.2.3	User-Specified Adjustments . . . . .	24
5.3	Outlier Detection . . . . .	25
5.3.1	Outlier Flagging Process . . . . .	28
5.3.2	AddOne Method . . . . .	30
5.3.3	AddAll Method . . . . .	31
5.4	Automatic Model Selection . . . . .	32
5.4.1	Default Model Selection . . . . .	32
5.4.2	Identification of Differencing Orders . . . . .	33
5.4.3	Identification of ARMA Model Orders . . . . .	35
5.4.4	Final Selection of the ARIMA Model . . . . .	38
<b>6</b>	<b>Conclusion</b>	<b>41</b>
<b>A</b>	<b>Appendix A</b>	<b>42</b>
A.1	X13-ARIMA-SEATS Output . . . . .	42
A.2	R Implementation . . . . .	85

## 2 Introduction

The Ministry of Statistics and Programme Implementation (MoSPI) is responsible for assembling and sharing key country indexes such as the Gross Domestic Product (GDP) and the Consumer Price Index (CPI). GDP represents the overall economic activity, while CPI measures inflation trends. Both indicators provide critical insights into a country's economic performance. Seasonal variations, due to factors like holidays and agriculture, often obscure the true trends. This necessitates seasonal adjustment, a statistical technique to remove these effects and reveal the underlying patterns in time series data.

Currently, MoSPI does not apply seasonal adjustments to its indices. This project's goal is to address the need for seasonal adjustment techniques using the widely adopted **X13-ARIMA-SEATS** model, developed by the U.S. Census Bureau. Our objective is to provide clarity on the technical details of this model and equip MoSPI with the tools for correct implementation. Economic indicators such as GDP and CPI are often influenced by seasonal factors like holidays, agricultural cycles, and other periodic events. These factors introduce systematic patterns in the data, which, if not accounted for, can mislead decision-makers about the actual trends in the economy. For instance, a rise in consumer spending around holidays may inflate the perception of economic growth, which is temporary.

The **X-13-ARIMA-SEATS** model is a powerful statistical tool designed to remove such seasonal effects. It is based on ARIMA (AutoRegressive Integrated Moving Average) modeling combined with signal extraction techniques. This model enables users to separate the trend, seasonal, and irregular components from time series data. Our project aims to:

- Provide a detailed technical explanation of the **X-13-ARIMA-SEATS** model for MoSPI.
- Test the appropriateness of the model on Indian economic data.
- Deliver an R package to apply the model on Indian data for seasonal adjustment.

We have made significant progress in understanding the ARIMA framework and its integration with signal extraction for seasonal adjustments. We are currently testing the model on sample datasets and preparing the technical document for submission to MoSPI. The dataset being used for testing purposes is the monthly combined CPI data of India from 2013 January till 2024 October.

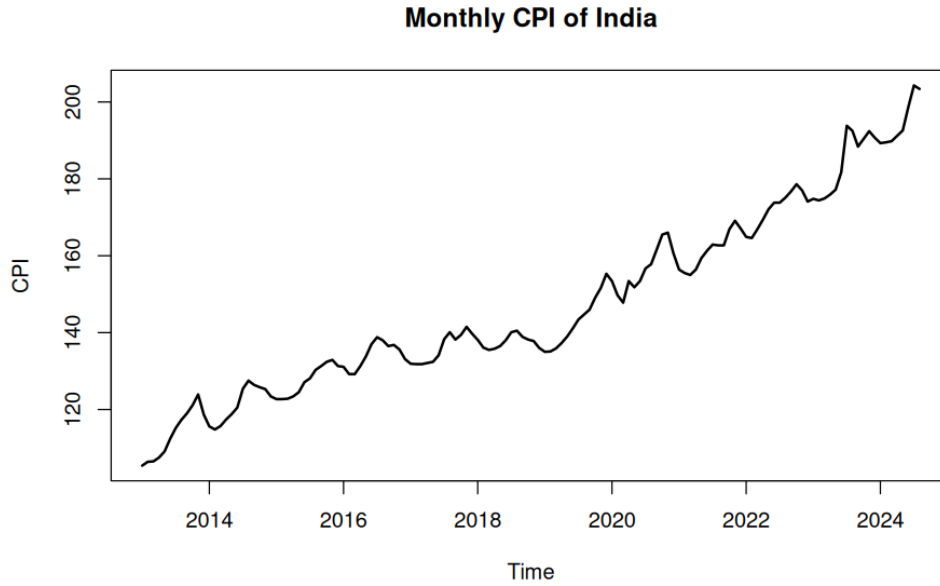


Figure 1: Plot showing the dataset being used for analysis.

### 3 X-13ARIMA-SEATS

The **X13-ARIMA-SEATS** is a comprehensive seasonal adjustment software developed by the U.S. Census Bureau [14]. It is widely used for time series analysis to remove seasonal variations from data and identify underlying trends. The software integrates two powerful algorithms: the X-11 method and SEATS (Signal Extraction in ARIMA Time Series). These techniques allow users to adjust for seasonal effects and extract trend and irregular components.

#### 3.1 SEATS and X-13

SEATS is a model-based approach that uses ARIMA modeling to decompose a time series into its trend, seasonal, and irregular components. It was originally developed by Agustin Maravall and Victor Gómez at the Bank of Spain [7] [8]. SEATS provides a statistically rigorous way to account for seasonal patterns and produce smoother trends [13].

X-13, on the other hand, builds on the earlier X-11 and X-12 methods, which use filter-based techniques for seasonal adjustment. The **X13-ARIMA-SEATS** software combines the strengths of both approaches, giving users flexibility in choosing between model-based and filter-based seasonal adjustments [6].

#### 3.2 RegARIMA and TRAMO

The regARIMA model used in **X13-ARIMA-SEATS** is based on TRAMO (Time Series Regression with ARIMA Noise, Missing Observations, and Outliers). TRAMO handles preprocessing tasks such as outlier detection, modeling calendar effects, and managing missing observations. It fits a regression model with ARIMA errors to clean the data before applying seasonal adjustment. The software is able to extend time series data through forecasting and backcasting, improving the accuracy of seasonal adjustments near the boundaries of the data [13] [7].

The **X13-ARIMA-SEATS** program operates in several stages to transform raw time series data into its seasonal, trend, and irregular components. The main steps involve reading the raw data, applying the regARIMA model for transformations, handling outliers and missing values, and then using either SEATS or X-11 to extract the seasonal and other components.

### 3.2.1 Workflow Explanation

[13]The general workflow of the **X13-ARIMA-SEATS** program can be broken down as follows:

- **Raw Data Input:** The process begins with inputting the raw time series data, which is defined in the ‘series’ spec of the specification file.
- **regARIMA Model:** The regARIMA model is applied to the data. This involves:
  - **Transformations:** Automatically or manually defined transformations (e.g., logarithmic or differencing).
  - **Handling Outliers and Missing Values:** Identifies and adjusts outliers, and fills missing data where applicable.
  - **Model Fitting:** Fits an ARIMA model to the transformed data, accounting for these adjustments.

The output from the regARIMA model is the linearized series, which is free of seasonal effects, outliers, and missing values.

- **Seasonal Adjustment:** This linearized series is passed through either the SEATS or X-11 algorithm to extract the seasonal, trend, and irregular components.
  - **SEATS:** A model-based approach for decomposing the series.
  - **X-11:** A filter-based approach for seasonal decomposition.
- **Final Output:** The program provides the seasonal component, trend component, and irregular component of the time series.

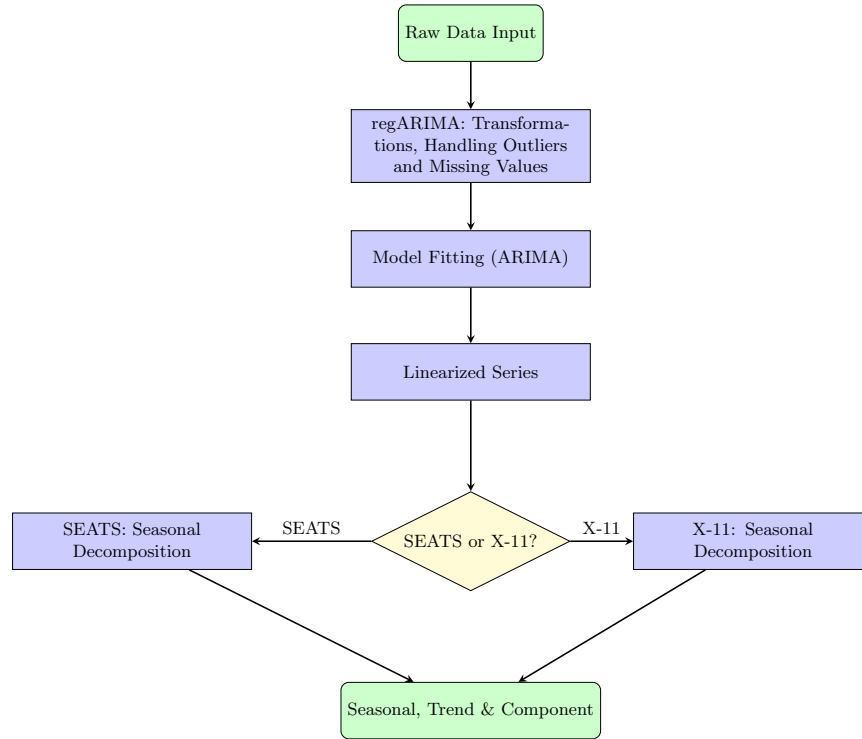


Figure 2: Flowchart of the Seasonal Adjustment Process

### 3.3 Using X13-ARIMA-SEATS

The software can be executed using the command line by specifying the path to the input specification (spec) file. A generic command to run X13-ARIMA-SEATS is:

```
path\x13as path\filename
```

The spec file contains the necessary instructions for running the software, including details about the time series data, ARIMA models, and output preferences. It is a simple text file with a '.spec' extension.

#### 3.3.1 Spec File and Key Specifications

A spec file consists of various specifications (specs) that control the flow of execution. Here is an example spec file for demonstration purpose.

```
series{
  title = "Consumer Food Price Index - All India Combined"
  start = 2013.01
  span = (2013.01, 2024.08)
  data = (
    105.4 106.4 106.5 107.5 109.1 112.4 115.2 117.3 119.0 121.1 123.9 118.7
    115.6 114.8 115.7 117.4 118.8 120.5 125.4 127.5 126.4 125.8 125.3 123.4
    122.7 122.7 122.8 123.4 124.5 127.1 128.1 130.3 131.3 132.4 132.9 131.3
    131.1 129.2 129.2 131.3 133.8 137.0 138.8 138.0 136.5 136.8 135.6 133.1
    131.9 131.8 131.8 132.1 132.4 134.1 138.3 140.1 138.2 139.4 141.5 139.7
    138.1 136.1 135.5 135.8 136.5 138.0 140.1 140.5 138.9 138.2 137.8 136.0
    135.0 135.1 135.9 137.3 139.0 141.1 143.4 144.7 146.0 149.1 151.6 155.3
    153.4 149.7 147.8 153.4 151.8 153.4 156.7 157.8 161.6 165.5 166.0 160.6
    156.4 155.5 155.0 156.4 159.4 161.3 162.9 162.7 162.7 166.9 169.1 167.1
    164.9 164.6 166.9 169.4 172.1 173.8 173.8 175.1 176.7 178.6 177.0 174.1
    174.8 174.4 174.9 175.9 177.2 181.7 193.8 192.5 188.4 190.4 192.4 190.7
    189.3 189.5 189.8 191.2 192.6 198.7 204.3 203.4
  )
}

transform{
  function = auto
}

automdl{maxorder = (3, ) }

outlier{types = (ls ao)}

estimate{
  save = residuals
}

regression {
  variables = (const, td)
  user = (diwali)
  start = 2013.01
  data = (
    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0405 -0.0405 0.0
    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0
    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.6595 0.6595 0.0
    0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0
  )
}
```

```

0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.3405 -0.3405  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0 -0.3595  0.3595  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.3405 -0.3405  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0 -0.6595  0.6595  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0 -0.0595  0.0595  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.3405 -0.3405  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0 -0.6595  0.6595  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.3405 -0.3405  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.3405 -0.3405  0.0)
print = rmx
}

seats{}

```

### 3.3.2 Commonly used specs in X13-ARIMA-SEATS

1. **SERIES** The SERIES spec defines the input time series data for analysis. It includes details such as the starting and ending dates, frequency, and data type. This spec is crucial as it tells the software what time series to use for seasonal adjustment and other analyses. The SERIES spec also allows users to set the precision of the data and handle missing values appropriately.
2. **AUTOMODEL** The AUTOMODEL spec automates the selection of the best-fitting ARIMA model by comparing several candidate models and selecting the one with the lowest AIC (Akaike Information Criterion). This is particularly useful when the user is unsure of the correct model structure. The procedure is closely based on TRAMO, a method developed by Gómez and Maravall (2000) [7]  
The method is detailed in Gómez and Maravall's paper, "Automatic Modeling Methods for Univariate Series" in *A Course in Time Series*, edited by D. Peña, G. C. Tiao, and R. S. Tsay, New York: J. Wiley and Sons, 2000 [11] [8].
3. **Estimate** This spec controls the estimation method used for fitting the ARIMA model. Maximum likelihood estimation (MLE) is typically used, but users can modify various estimation settings such as the number of iterations and convergence criteria.
4. **Outlier** The Outlier spec detects and adjusts for outliers in the data. Three types of outliers are commonly detected: Additive Outliers (AO), Level Shifts (LS), and Transitory Changes (TC). The software automatically identifies these outliers and includes them as regressors in the model.
5. **Regression** The Regression spec allows users to specify the regression variables used in the regARIMA model. These variables can include predefined effects, such as trading day and holiday adjustments, or user-defined regressors [13].
6. **Transform** The Transform spec handles transformations of the data, such as logarithmic transformations, to stabilize the variance or make the series more stationary. This is particularly useful when dealing with time series that exhibit non-constant variance.
7. **SEATS** The SEATS spec controls the use of the SEATS algorithm for seasonal adjustment. This includes options for model selection, output settings, and diagnostics for the decomposition of the time series into its components.
8. **X-11** The X-11 spec is used when applying the X-11 seasonal adjustment method. This spec provides options for controlling seasonal filters, diagnostics, and the handling of trading day effects.

## 4 Background Mathematical Details

The definition and theorem mentioned in this section have been taken from "Time Series Analysis: Forecasting and Control" by *Box, George E. P. and Jenkins, Gwilym M. and Reinsel, Gregory C. and Ljung, Greta M.* [3]

**Definition 4.1.** A time series  $\{X_t\}$  is an ordered collection (indexed by time) of random variables.

**Definition 4.2.** A time series is called stationary if the joint distribution of any finite collection of data points is time-independent. Formally, for all positive integers  $h, n_1, n_2, \dots, n_k$ ,

$$X_{n_1}, X_{n_2}, \dots, X_{n_k} \stackrel{d}{=} X_{n_1+h}, X_{n_2+h}, \dots, X_{n_k+h}.$$

**Definition 4.3.** A time series is called covariance-stationary if the following conditions are satisfied for all positive integers  $h, n, n_1$ , and  $n_2$ :

$$\mathbb{E}[X_n] = \mu < \infty$$

$$\text{Cov}(X_{n_1}, X_{n_2}) = \text{Cov}(X_{n_1+h}, X_{n_2+h}) < \infty$$

All stationary time series are covariance-stationary, but the converse is not generally true.

**Definition 4.4.** A time series is called a Gaussian process if the joint distribution of any finite collection of data points follows a multivariate normal distribution.

**Theorem 4.1.** A covariance-stationary Gaussian process is stationary.

**Definition 4.5.** For a covariance-stationary time series, the autocovariance function is defined as:

$$\rho(h) = \text{Cov}(X_n, X_{n+h}),$$

where  $h$  is called the lag.

**Definition 4.6.** White noise  $\varepsilon_t$  is a sequence of uncorrelated random variables with zero mean and constant variance  $\sigma^2$ .

### 4.1 Time Series Models

This section provides the mathematical foundations for time series modeling, starting with the basic models like  $\text{AR}(p)$  and  $\text{MA}(q)$ , and progressing to more complex models such as  $\text{ARMA}(p, q)$ ,  $\text{ARIMA}(p, d, q)$ , and  $\text{ARIMA}(p, d, q)(P, D, Q)_s$ .

#### 1. $\text{AR}(p)$ Process

**Definition 4.7.** A time series  $\{X_t\}$  is said to follow an  $\text{AR}(p)$  process if it satisfies the equation:

$$X_t = \sum_{i=1}^p \phi_i X_{t-i} + \varepsilon_t$$

where  $\phi_1, \dots, \phi_p$  are the parameters of the model, and  $\varepsilon_t$  is white noise with mean zero and variance  $\sigma^2$ .

Here,  $p$  is called the order of the AR process. The  $\text{AR}(p)$  process can be equivalently written using the backshift operator  $B$  as:

$$X_t = \sum_{i=1}^p \phi_i B^i X_t + \varepsilon_t$$

where the backshift operator  $B$  is defined by:

$$B^i X_t = X_{t-i}$$

In an  $\text{AR}(p)$  process the current value of the series is a linear sum of the previous values and an independent error term.



## 2. MA( $q$ ) Process

**Definition 4.8.** A time series  $\{X_t\}$  is said to follow an MA( $q$ ) process if it satisfies the equation:

$$X_t = \sum_{i=1}^q \theta_i \varepsilon_{t-i} + \varepsilon_t$$

where  $\theta_1, \dots, \theta_q$  are the coefficients of the model, and  $\varepsilon_t$ 's are white noise error terms.

Here,  $q$  is called the order of the MA process. The MA( $q$ ) process can be equivalently written in terms of the backshift operator  $B$  as:

$$X_t = (1 + \sum_{i=1}^q \theta_i B^i) \varepsilon_t.$$

An MA( $q$ ) process can be thought of as a weighted moving sum of a white noise series.

## 3. ARMA( $p, q$ ) Process

**Definition 4.9.** A time series  $\{X_t\}$  is said to follow an ARMA( $p, q$ ) process if it satisfies the equation:

$$X_t = \sum_{i=1}^p \phi_i X_{t-i} + \sum_{i=1}^q \theta_i \varepsilon_{t-i} + \varepsilon_t$$

where  $\phi_1, \dots, \phi_p$  and  $\theta_1, \dots, \theta_q$  are the coefficients of the model, and  $\varepsilon_t$  are white noise error terms.

Here,  $p, q$  are called the order of the ARMA process. The ARMA( $p, q$ ) process can be equivalently written in terms of the backshift operator  $B$  as:

$$\left(1 - \sum_{i=1}^p \phi_i B^i\right) X_t = (1 + \sum_{i=1}^q \theta_i B^i) \varepsilon_t$$

or equivalently as:

$$\phi(B)X_t = \theta(B)\varepsilon_t$$

ARMA( $p, q$ ) process is a combination of the AR and MA models which means the current value is a linear sum of the previous values and a weighted sum of some white noise process.

**Theorem 4.2.** All MA( $q$ ) processes of finite order  $q$  are covariance stationary and  $\rho(h) = 0$  for  $|h| > q$ .

**Theorem 4.3.** An AR( $p$ ) process  $\{X_t\}$  defined by  $\Phi(B)X_t = \varepsilon_t$  is covariance stationary if the roots of the polynomial  $\Phi(z) = 0$  lie outside the unit circle.

**Theorem 4.4.** An ARMA( $p, q$ ) process  $\{X_t\}$  defined by  $\Phi(B)X_t = \Theta(B)\varepsilon_t$  is covariance stationary if the roots of the polynomial  $\Phi(z) = 0$  lie outside the unit circle.

## 4. ARIMA( $p, d, q$ ) Process

We define the differencing operator  $\nabla$  as  $\nabla X_t = X_t - X_{t-1}$  or  $\nabla = 1 - B$ .

**Definition 4.10.** A time series  $X_t$  is said to be an ARIMA( $p, d, q$ ) process if  $Y_t = \nabla^d X_t$  is a stationary ARMA( $p, q$ ) process.

## 5. ARIMA( $p, d, q$ )( $P, D, Q$ )<sub>s</sub> Process

The general ARIMA model can be extended to include seasonal factors. The seasonal AR and MA components are represented by polynomials with seasonal lags. The general ARIMA( $p, d, q$ )( $P, D, Q$ )<sub>s</sub> model is given by:

$$\phi(B)\Phi(B^s)(1-B)^d(1-B^s)^D z_t = \theta(B)\Theta(B^s)a_t$$

where:

- $\phi(B)$  and  $\theta(B)$  are the non-seasonal autoregressive (AR) and moving average (MA) polynomials.
- $\Phi(B^s)$  and  $\Theta(B^s)$  are the seasonal AR and MA polynomials with seasonal period  $s$ .
- $(1 - B)^d$  represents the non-seasonal differencing operator.
- $(1 - B^s)^D$  represents the seasonal differencing operator.
- $a_t$  is white noise with mean zero and variance  $\sigma^2$ .

## 4.2 Time Series Decomposition

A time series can often be broken down into three components [13]:

1. **Trend Component** ( $m_t$ ): The long-term movement or direction in the data.
2. **Seasonal Component** ( $s_t$ ): The repeating, periodic fluctuation (e.g., annual cycles).
3. **Random Component** ( $e_t$ ): The stochastic, unpredictable part of the series.

The decomposition can be done additively or multiplicatively:

- **Additive Decomposition:**  $X_t = m_t + s_t + e_t$
- **Multiplicative Decomposition:**  $X_t = m_t \cdot s_t \cdot e_t$

Seasonal adjustment is the process of removing the seasonal component from a time series to analyze the underlying trend and cycle independently. It is essential for analyzing economic data that exhibits seasonal patterns, such as consumer spending, which often increases before holidays like Christmas.

In many practical applications, the observed time series can be influenced by both deterministic effects (such as calendar effects, trends, or interventions) and stochastic components (captured by ARIMA processes). The Regression with ARIMA Noise (RegARIMA) model combines a linear regression model with ARIMA errors. This approach allows us to model both the deterministic structure of the time series and the stochastic dependencies (e.g., autocorrelation) present in the error terms. The general form of a regression model with ARIMA errors is given by:

$$y_t = X_t\beta + z_t$$

where:

- $y_t$  is the observed time series at time  $t$ .
- $X_t$  is a vector of regression variables (e.g., calendar effects, external regressors).
- $\beta$  is the vector of coefficients corresponding to the regression variables.
- $z_t$  is the error term, which follows an ARIMA process.

The residuals  $z_t$  are assumed to follow an ARIMA( $p, d, q$ ) process, which captures the autocorrelation and the non-stationarity of the data. The ARIMA process for  $z_t$  is expressed as:

$$\phi(B)\Phi(B^s)(1 - B)^d(1 - B^s)^D z_t = \theta(B)\Theta(B^s)a_t$$

Combining the regression component and the ARIMA noise model, we get the complete RegARIMA model:

$$\phi(B)\Phi(B^s)(1 - B)^d(1 - B^s)^D (y_t - X_t\beta) = \theta(B)\Theta(B^s)a_t$$

This model provides a comprehensive approach to handle both deterministic and stochastic components in the time series. The regression variables  $X_t$  account for known, external influences, while the ARIMA model for  $z_t$  captures the autocorrelation and non-stationarity in the data.

### 4.3 Ljung-Box Test for Model Residuals

[2] The Ljung-Box test is a statistical diagnostic tool used to assess whether the residuals of a time series model exhibit autocorrelation. This test helps in validating the adequacy of models such as ARIMA in time series analysis. The main objective of the Ljung-Box test is to determine if residuals (errors) from a time series model are independently distributed (i.e., they exhibit no serial autocorrelation up to a certain lag). If autocorrelation is detected, this indicates possible model inadequacy or misspecification, suggesting that adjustments or alternative models may be needed for accurate forecasting. Typically, a sufficient sample size is required for reliable conclusions. The hypotheses for the Ljung-Box test are as follows:

- **Null Hypothesis ( $H_0$ ):** There is no autocorrelation in residuals up to the specified lag (the residuals are independently distributed).
- **Alternative Hypothesis ( $H_a$ ):** There is autocorrelation in residuals, suggesting that the model may not adequately capture the structure of the data.

Rejecting  $H_0$  suggests that the model may be misspecified or incomplete. The Ljung-Box test statistic  $Q$  is calculated as:

$$Q = n(n+2) \sum_{k=1}^m \frac{\hat{r}_k^2}{n-k}$$

where:

- $n$  is the number of observations,
- $\hat{r}_k$  is the sample autocorrelation at lag  $k$ ,
- $m$  is the number of lags being tested.

The test statistic  $Q$  follows an approximate chi-square ( $\chi^2$ ) distribution with  $h$  degrees of freedom, where  $h$  is the number of parameters estimated in the model (such as  $p$  and  $q$  in ARIMA models). At a given significance level  $\alpha$ , the critical value is derived from the chi-square distribution. If  $Q$  exceeds the critical value, the null hypothesis  $H_0$  is rejected, indicating significant autocorrelation in the residuals and suggesting that the model may be inadequate.

### 4.4 The Hannan-Rissanen Estimation Method

[7] The Hannan-Rissanen (HR) estimation method is an efficient algorithm for estimating the parameters of an ARMA( $p, q$ ) model. It provides a practical approach to model identification and estimation by utilizing linear regression techniques, which are computationally less intensive than full maximum likelihood estimation but still yield consistent and asymptotically efficient estimators. This is useful in the automatic model selection procedure where we need to fit a lot of models. The HR estimation method involves a three-step procedure:

1. **Preliminary Estimation of Innovations:** Fit a high-order autoregressive (AR) model to the time series data to obtain preliminary estimates of the residuals (innovations).
2. **Linear Regression for Parameter Estimation:** Use the estimated innovations from the first step to perform a linear regression and estimate the AR and MA parameters of the model.
3. **Refinement of Estimates (Optional):** Improve the parameter estimates by correcting biases and enhancing efficiency, although this step is often optional for model selection purposes.

**Estimation of ARMA Model Parameters** After selecting the ARMA( $p, q$ ) model orders, the next step is to estimate the model parameters. The estimation process involves preliminary estimation of innovations, parameter estimation through linear regression, and optional refinement of estimates.

**Step 1: Preliminary Estimation of Innovations** Given a stationary time series  $\{Y_t\}$  of length  $N$ , we aim to fit an ARMA( $p, q$ ) model. We begin by fitting a high-order autoregressive (AR) model of order  $\tilde{p}$ :

$$Y_t = \sum_{j=1}^{\tilde{p}} \tilde{\phi}_j Y_{t-j} + \epsilon_t, \quad t = \tilde{p} + 1, \tilde{p} + 2, \dots, N, \quad (1)$$

where  $\tilde{\phi}_j$  are the AR coefficients, and  $\epsilon_t$  represents the residuals (innovations). The order  $\tilde{p}$  is chosen to be large enough to capture the underlying dynamics, commonly set as:

$$\tilde{p} = \max \{ \lfloor \log(N) \rfloor, 2 \max\{p, q\} \}. \quad (2)$$

The AR coefficients  $\tilde{\phi}_j$  are estimated using methods like the Yule-Walker equations or the Burg algorithm. The residuals  $\hat{\epsilon}_t$  obtained from (1) serve as preliminary estimates of the innovations required for the next step.

**Step 2: Linear Regression for Parameter Estimation** We aim to estimate the parameters of the ARMA( $p, q$ ) model:

$$Y_t = \sum_{i=1}^p \phi_i Y_{t-i} + \epsilon_t + \sum_{j=1}^q \theta_j \epsilon_{t-j}, \quad t = n + 1, n + 2, \dots, N, \quad (3)$$

where  $n = \max\{\tilde{p}, p, q\}$ .

Since the innovations  $\epsilon_t$  and their past values are not observable, we use the preliminary estimates  $\hat{\epsilon}_t$  from Step 1. Rearranging (3), we set up the linear regression equation:

$$\hat{\epsilon}_t = Y_t - \sum_{i=1}^p \phi_i Y_{t-i} - \sum_{j=1}^q \theta_j \hat{\epsilon}_{t-j}. \quad (4)$$

We perform a regression of  $\hat{\epsilon}_t$  on the past values  $Y_{t-i}$  and the estimated innovations  $\hat{\epsilon}_{t-j}$  to obtain estimates  $\hat{\phi}_i$  and  $\hat{\theta}_j$  of the AR and MA parameters, respectively. The residual sum of squares from this regression is used to estimate the variance of the innovations:

$$\hat{\sigma}_\epsilon^2 = \frac{1}{N - n} \sum_{t=n+1}^N \left( \hat{\epsilon}_t - \hat{\epsilon}_t^{\text{predicted}} \right)^2, \quad (5)$$

where  $\hat{\epsilon}_t^{\text{predicted}}$  are the fitted values from the regression in (4).

**Step 3: Refinement of Estimates (Optional)** To improve the parameter estimates and correct for biases, we can refine the estimates by iterating the procedure:

1. Use the estimates  $\hat{\phi}_i$  and  $\hat{\theta}_j$  to filter the series and obtain updated innovations  $\hat{\epsilon}_t$ .
2. Re-estimate the parameters using the updated innovations by repeating Step 2.

This iterative process enhances the efficiency of the estimators but may not be necessary for model selection purposes.

## 4.5 Information Criteria for Model Selection

[13] [10] Selecting the most suitable time series model requires a balance between goodness of fit and model complexity. Information criteria offer a systematic approach to compare models by quantifying this trade-off. The X13-ARIMA-SEATS program utilizes several such criteria, including the Akaike Information Criterion (AIC), the corrected AIC (AICC), the Bayesian Information Criterion (BIC), and the Hannan-Quinn criterion (HQ). These criteria help in identifying models that best explain the data without overfitting.

Let  $L_N$  denote the maximized log-likelihood of a model based on  $N$  effective observations after differencing, and  $n_p$  represent the number of estimated parameters, including the white noise variance. The information criteria are defined as follows:

- **Akaike Information Criterion (AIC):**

$$\text{AIC}_N = -2L_N + 2n_p$$

- **Corrected Akaike Information Criterion (AICC):**

$$\text{AICC}_N = -2L_N + 2n_p \left( \frac{N}{N - n_p - 1} \right)$$

- **Bayesian Information Criterion (BIC):**

$$\text{BIC}_N = -2L_N + n_p \log N$$

- **Hannan-Quinn Criterion (HQ):**

$$\text{HQ}_N = -2L_N + 2n_p \log(\log N)$$

These formulas combine the model's fit (through the log-likelihood  $L_N$ ) with a penalty term that increases with the number of parameters  $n_p$  and, for some criteria, the sample size  $N$ .

**Model Comparison Using Information Criteria** When comparing models, the one with the lower value of the chosen information criterion is generally preferred. For instance, if we have two models—Model 1 and Model 2—with corresponding log-likelihoods  $L_N^{(1)}$  and  $L_N^{(2)}$  and numbers of parameters  $n_p^{(1)}$  and  $n_p^{(2)}$ , the difference in their AIC values is:

$$\Delta \text{AIC} = \text{AIC}_N^{(1)} - \text{AIC}_N^{(2)} = -2 \left( L_N^{(1)} - L_N^{(2)} \right) + 2 \left( n_p^{(2)} - n_p^{(1)} \right)$$

A positive  $\Delta \text{AIC}$  suggests that Model 2 is preferred over Model 1 according to the AIC criterion.

**Asymptotic Properties and Type I Error Probabilities** In the context of nested models—where one model is a special case of another—the difference in log-likelihoods multiplied by  $-2$  follows, asymptotically, a chi-squared distribution with degrees of freedom equal to the difference in the number of parameters:

$$-2 \left( L_N^{(1)} - L_N^{(2)} \right) \sim \chi_{n_p^{(2)} - n_p^{(1)}}^2$$

This property allows us to assess the probability of incorrectly favoring a more complex model (Type I error). Specifically, the asymptotic probability that the more complex model (Model 2) will be incorrectly preferred is:

$$P(\Delta \text{AIC} > 0) = P \left( \chi_{n_p^{(2)} - n_p^{(1)}}^2 > 2(n_p^{(2)} - n_p^{(1)}) \right)$$

This expression quantifies the risk of overfitting due to the addition of unnecessary parameters.

**Practical Considerations in Model Selection** When applying information criteria:

- **Balance Fit and Complexity:** While a lower information criterion value indicates a better model, it's essential to ensure that the improvement in fit justifies the increase in complexity.
- **Thresholds for Model Differences:** Small differences in information criterion values may not be practically significant. For AICC, differences greater than 3 are often considered substantial evidence favoring one model over another.
- **Multiple Model Comparison:** In cases involving multiple models, it's advisable to consider models with information criterion values close to the minimum. This approach acknowledges model uncertainty and avoids over-reliance on a single metric.

- **Contextual Understanding:** Information criteria should be used alongside domain knowledge and other diagnostic tools. They are based on asymptotic approximations and may not fully capture the nuances of model adequacy in finite samples.

In seasonal adjustment and time series modeling, information criteria assist in decisions such as including trading day effects or Easter effects. For example, when testing for trading day effects using the `aictest` option in `X13-ARIMA-SEATS`, setting an appropriate `aicdiff` value can control the Type I error probability, influencing whether additional regressors are included.

## 5 RegARIMA

The RegARIMA model in the `X13-ARIMA-SEATS` software is an advanced tool for handling regression effects, detecting outliers, and fitting ARIMA models to time series data. The process includes several key steps, each involving tests and decisions based on the time series characteristics.

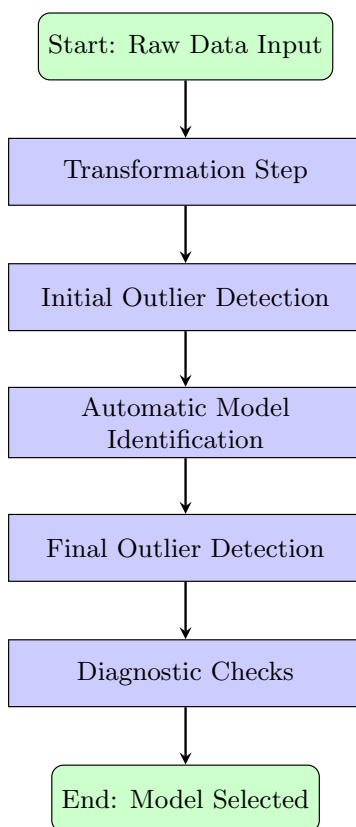


Figure 3: Flowchart Illustrating the Steps Taken During the RegARIMA Process

### 5.1 Regression Variables

[13] [1] [7] Regression variables in the `X-13ARIMA-SEATS` program are essential components used to model and adjust for effects that are external to the inherent properties of the time series data. They allow for the inclusion of deterministic factors such as trading day effects, holiday effects, outliers, and other user-defined variables that can influence the behavior of the series. Incorporating these variables enhances the accuracy of the regARIMA model by accounting for known patterns and anomalies, leading to better forecasting and seasonal adjustment.

The program supports a variety of predefined regression variables, each designed to model specific effects in time series data. The main types include:

- **Trading Day Variables:** Adjust for variations due to the differing number of weekdays in each month.
- **Holiday Effect Variables:** Model the impact of holidays that do not occur on the same date each year.
- **Length-of-Month (LOM), Length-of-Quarter (LOQ), and Leap Year (LPYEAR) Variables:** Account for variations in the number of days or months in each period.
- **Outlier Regressors:** Capture abrupt changes in the series due to one-time events.
- **User-Defined Regressors:** Custom variables defined by the user to model specific effects.
- **Change of Regime Regressors:** Model structural changes in the series at specified points in time.

### 5.1.1 Stock and Flow Series

Understanding the nature of the time series is crucial when selecting regression variables. Time series data can be classified as either **stock** or **flow** series:

- **Flow Series:** Represent accumulations over a period, such as total sales in a month. They measure the activity or transactions occurring within a specific time frame. Examples of flow series include the Consumer Price Index (CPI) and Gross Domestic Product (GDP), as they represent the aggregate value over a period.
- **Stock Series:** Measurements taken at a specific point in time, reflecting the level of a variable at that moment. For instance, inventory levels at the end of each month or the balance of a bank account on a particular day are stock series.

The distinction between stock and flow series affects the selection of regression variables, especially trading day and holiday effect variables, as the impact of these factors can differ between stock and flow measurements.

### 5.1.2 Trading Day Regression Variables

[1]Trading day effects arise due to the variation in the number and distribution of weekdays in each month or quarter. Economic activities often differ between weekdays and weekends, and even among different weekdays. For instance, consumer spending might be higher on weekends, or industrial production might be lower on certain weekdays due to maintenance schedules. These variations can introduce systematic patterns in the time series data that need to be accounted for in the model. The **X13-ARIMA-SEATS** program provides several predefined trading day regression variables to model these effects:

- **td:** Includes six day-of-week contrasts for flow series.
- **tdnolpyear:** Similar to **td** but does not adjust for leap year effects.
- **td1coef:** Uses a single trading day contrast variable for flow series.
- **tdstock[w]:** For stock series, models day-of-week effects based on the  $w$ -th day of the month.
- **tdstock1coef[w]:** Similar to **tdstock[w]** but uses a single coefficient model.

**Flow Series Trading Day Variables** [13]For flow series like CPI and GDP, which accumulate values over a period, the trading day effect is modeled using contrasts of weekdays:

- **td:** This option includes six variables representing the differences between the number of each weekday and Sundays in a month. The variables are defined as:

$$T_i = (\text{Number of days of type } i) - (\text{Number of Sundays})$$

where  $i$  ranges from Monday to Saturday. This approach accounts for the effect of each weekday relative to Sunday.

- **tdnolpyear**: Similar to **td** but excludes adjustments for leap year effects, useful when leap year adjustments are handled separately.
- **td1coef**: Simplifies the trading day effect into a single variable representing the contrast between weekdays and weekends:

$$T = (\text{Number of weekdays}) - \frac{5}{2}(\text{Number of Saturdays and Sundays})$$

This assumes that all weekdays have a similar effect, and Saturdays and Sundays have a different but consistent effect.

- **Leap Year Adjustment**: When using **td** or **td1coef**, the program automatically adjusts for leap year effects unless specified otherwise. Leap years can affect the number of weekdays in a month, especially February.

**Stock Series Trading Day Variables** For stock series, which measure the value at a specific point in time, the trading day effect depends on the day of the week corresponding to that point:

- **tdstock[w]**: Models the effect based on the  $w$ -th day of each month. If stocks are recorded on the last day of the month, **w** is set to 31. The variables are contrasts between each day of the week and Sundays, similar to the flow series but adjusted for the specific day.
- **tdstock1coef[w]**: Uses a single coefficient to model the trading day effect for stock series:

$$I(w)_t = -\frac{1}{5}D(w)1_t - \frac{1}{5}D(w)2_t + \frac{1}{5}D(w)3_t + \frac{3}{5}D(w)4_t + D(w)5_t$$

where  $D(w)i_t$  is a dummy variable indicating whether the  $w$ -th day of month  $t$  is a specific weekday.

- **Adjustments for Month Length**: Stock series trading day variables take into account the varying lengths of months by adjusting  $w$  if it exceeds the number of days in the month.

## Considerations When Using Trading Day Variables

- **Selection of Variables**: The choice between **td**, **td1coef**, and their stock counterparts depends on the nature of the series and the desired model complexity.
- **Interaction with Other Variables**: Trading day variables should be carefully combined with other regressors, such as holiday effects or length-of-month adjustments, to avoid multicollinearity or overfitting.
- **Flow vs. Stock Series**: Ensure that the appropriate trading day variables are selected based on whether the series is a flow or stock series, as the effects and definitions differ.

### 5.1.3 Holiday Effect Variables

Holidays can significantly impact economic activities, causing deviations in time series data. Holidays that do not occur on the same date each year, such as Easter or Diwali, require special attention because their effects shift between months or quarters over time. The program includes predefined variables to model these effects:

- **easter[w]**: Models the Easter effect for flow series, where  $w$  specifies the window length in days before Easter.
- **labor[w]**: Captures the effect of Labor Day.
- **thank[w]**: Models the effect of Thanksgiving.



**Easter Effect Variables** The date of Easter varies between March 22 and April 25, affecting economic activities differently each year. The `easter[w]` variable adjusts for this by modeling the impact over a window of  $w$  days before Easter. The variable for month  $t$  is defined as:

$$E(w, t) = \frac{1}{w} \times \text{Number of days within } w \text{ days before Easter that fall in month } t$$

This variable is deseasonalized by subtracting long-term (500 year) monthly means to isolate the Easter effect. When including the Easter effect in the model:

- Choose an appropriate window length  $w$  based on the expected duration of the Easter impact on the series.
- Use the `easter[w]` variable in the `regression` spec:

```
regression {
  variables = (easter[8])
}
```

**Modeling the Diwali Effect** In the Indian context, the Easter effect is not significant. However, festivals like Diwali have a substantial impact on economic activities due to increased consumer spending, production adjustments, and other economic behaviors surrounding the festival. Similar to the Easter regressor, a Diwali regressor can be created to model this effect. To create a Diwali regressor:

1. **Define the Affected Period:** Determine the window around Diwali during which economic activities are influenced. For example,  $w$  days before or after the festival date.
2. **Calculate Monthly Proportions:** For each year, calculate the proportion of the affected period that falls within each month. This accounts for the shifting dates of Diwali according to the lunar calendar.
3. **Construct the Regressor:** Use these proportions as the regressor values for the corresponding months in the time series.
4. **Deseasonalize the Regressor:** Subtract long-term monthly means to remove the average seasonal pattern associated with Diwali from the regressor.
5. **Include in the Model:** Add the Diwali regressor to the `regression` spec as a user-defined variable:

The code to generate the Diwali regressor is provided in the appendix, detailing the steps to calculate the regressor values based on historical Diwali dates and the affected period. The choice of  $w$  should reflect the duration of Diwali's impact on the economic activities relevant to the series. The program also includes variables for other holidays, primarily relevant in specific contexts:

- `labor[w]`: Models the effect of Labor Day, significant in countries where this holiday impacts economic activities. The variable is constructed similarly to the Easter regressor, accounting for the days before Labor Day.
- `thank[w]`: Models the effect of Thanksgiving, primarily relevant to the United States. This variable accounts for the period from a specified number of days before Thanksgiving through December 24.

#### 5.1.4 Length-of-Month, Length-of-Quarter, and Leap Year Variables

Variations in the number of days in a month or quarter can significantly influence time series data, especially for flow series where the total amount accumulated over the period depends on its length. To account for these variations, the `X13-ARIMA-SEATS` program provides variables that adjust for the length of the period, ensuring that comparisons across time are meaningful by normalizing the data.

**Length-of-Month (LOM) Variable** The `lom` variable adjusts for the varying lengths of months in monthly flow series. It is defined as:

$$\text{LOM}_t = m_t - \bar{m}$$

where:

- $m_t$  is the number of days in month  $t$ .
- $\bar{m} = 30.4375$  is the average length of a month, calculated as the average number of days in a year (365.25) divided by 12.

Including the `lom` variable in the regression model accounts for the fact that months with more days may naturally have higher values due to their length, rather than an underlying change in economic activity. To include the length-of-month adjustment in the model, specify the `lom` variable in the `regression spec`:

```
regression {
  variables = (lom)
}
```

**Length-of-Quarter (LOQ) Variable** For quarterly flow series, the `loq` variable serves a similar purpose, adjusting for the varying number of days in each quarter. It is defined as:

$$\text{LOQ}_t = q_t - \bar{q}$$

where:

- $q_t$  is the number of days in quarter  $t$ .
- $\bar{q} = 91.3125$  is the average length of a quarter, calculated as the average number of days in a year divided by 4.

**Leap Year (LPYEAR) Variable** The presence of an extra day in February during leap years can affect time series data, particularly for flow series. The `lpyear` variable adjusts for this effect. It is defined as:

$$\text{LPYEAR}_t = \begin{cases} 0.75 & \text{if } t \text{ is a leap year February} \\ -0.25 & \text{if } t \text{ is a non-leap year February} \\ 0 & \text{otherwise} \end{cases}$$

This variable accounts for the fact that February in leap years has 29 days, potentially increasing the accumulated value for that month compared to non-leap years.

#### Considerations when using LOM, LOQ, and LPYEAR variables

- **Compatibility with Trading Day Variables:** When using `td` or `td1coef`, the program automatically handles length-of-month and leap year adjustments at the time of transformation step as given in the transformation section. Including `lom`, `loq`, or `lpyear` in addition may lead to multicollinearity or over-adjustment.
- **Series Type:** These variables are primarily relevant for flow series, where the amount accumulated is directly affected by the length of the period.
- **Avoiding Conflicts:** Do not include both `lom` and `lpyear` in the model unless appropriate. If `td` is specified, avoid including `lom` or `loq` because the model will adjust for these in the transformation step automatically. If one wants to use both trading day variables and length of month variable together then use `tdnolpyear` instead of `td` which does not automatically pre-adjust the series for the length of month or quarter.

An example of including length-of-month and leap year variables in the regression spec:

```
regression {
    variables = (tdnolpyear lom)
}
```

In this example, `tdnolpyear` includes the six trading day contrast variables without leap year adjustments, and `lom` is responsible for the effect of varying lengths of months.

### 5.1.5 Outlier Regressors

[13] [7] [12] [4] [5] Outlier regressors are used to model sudden, unexpected changes in a time series due to events such as strikes, natural disasters, policy changes, or other anomalies. These regressors are crucial for improving the model fit and ensuring that unusual observations do not distort the estimation of other model parameters. A detailed explanation of outlier detection and modeling is provided in the outlier detection subsection.

### 5.1.6 User-Defined Regressors and the `usertype` Argument

Users may have specific effects they wish to model that are not covered by the predefined regression variables. The program allows for the inclusion of custom, user-defined regressors. The `usertype` argument assigns a type to each user-defined regressor, enabling the program to treat them similarly to predefined variables and to apply appropriate processing and reporting.

The `usertype` argument assigns a specific type to each user-defined regressor, which can be any of the predefined types such as:

- `holiday`, `holiday2`, `holiday3`, etc.
- `ao`, `ls`, `so` (for outlier types)
- `td` (for trading day effects)
- `seasonal` (for seasonal effects)

By specifying the type, the program can:

- Apply appropriate processing, such as deseasonalization.
- Include the regressor in the correct group for statistical testing and reporting.
- Control whether the effect is adjusted out of the series prior to seasonal adjustment.

Assigning different types to multiple user-defined regressors:

```
regression {
    user = (holiday_new ao_event)
    usertype = (holiday ao)
    file = "user_regressors.dat"
    format = "datevalue"
}
```

### Considerations When Using User-Defined Regressors

- **Data Coverage:** Ensure that the regressor data spans the entire period of the series, including any forecasts or backcasts.
- **Type Assignment:** Assign the appropriate type using `usertype` to enable correct processing and interpretation.
- **Interactions with Predefined Regressors:** Be cautious of potential multicollinearity when combining user-defined regressors with predefined ones of the same type.

### 5.1.7 Change of Regime Regressors

Change of regime regressors are used to model structural changes in a time series at specific dates, allowing for different parameter values before and after the change point. These changes can be due to policy shifts, market reforms, economic crises, or any event that fundamentally alters the behavior of the series. By including change of regime regressors, analysts can capture these shifts and improve the model's accuracy during periods of transition.

There are two main types of change of regime regressors in the **X-13ARIMA-SEATS** program:

- **Full Change of Regime Regressor:** Captures a change in the effect of a regressor starting at a specified date, modeling different effects before and after the change.
- **Partial Change of Regime Regressor:** Models the effect of a regressor only before or after a specified date.

Change of regime regressors are specified by appending the change date to the regressor name using slashes:

- **Full Change of Regime:** `reg/date/`
- **Partial Change (Zero Before Date):** `reg//date/`
- **Partial Change (Zero On and After Date):** `reg/date//`

Here, `reg` represents the name of the regression variable (e.g., `td`, `easter[8]`), and `date` is the change point specified in `YYYY.MM` format.

**Example: Modeling a Structural Change in Trading Day Effects** Suppose there is evidence that the trading day effects in a flow series changed starting in January 2010. To model this change, you can specify a full change of regime regressor:

```
regression {  
    variables = (td/2010.01/)  
}
```

This specification includes both the standard trading day variables and additional regressors to capture the change in trading day effects from January 2010 onward.

**Interpreting Change of Regime Coefficients** The coefficients resulting from change of regime regressors provide insights into how the effect of a variable changes over time:

- **Basic Regressor Coefficient ( $a_j$ ):** Represents the effect of the regressor after the change date.
- **Change Regressor Coefficient ( $b_j$ ):** Represents the difference in the regressor's effect before the change date compared to after.

Statistically significant  $b_j$  coefficients indicate a notable change in the effect of the regressor at the specified date.

#### Considerations When Using Change of Regime Regressors

- **Compatibility:** Change of regime regressors can be used with most predefined regression variables, including trading day variables, holiday effects, and length-of-month adjustments.
- **Singularity Issues:** Ensure that the inclusion of change of regime regressors does not introduce multicollinearity or singularity into the model.
- **Model Complexity:** Adding change of regime regressors increases the number of parameters. Use statistical tests (i.e. `aictest`) to justify their inclusion.

### 5.1.8 AIC-Based Regressor Selection Procedure

Model selection involves choosing the appropriate variables to include in the regression to achieve a balance between model fit and complexity. The Akaike Information Criterion (AIC) is a widely used measure for this purpose, and the X-13ARIMA-SEATS program incorporates an AIC-based procedure to assist in regressor selection.

**Implementing the AIC Test in the Program** The `aictest` argument specifies which regressors to test for inclusion based on their impact on the AICC. The `aicdiff` or `pvaictest` arguments set the criteria for inclusion:

- `aictest`: Lists the regressors to be tested (e.g., `td`, `easter`).
- `aicdiff`: Sets the minimum difference in AICC required for a regressor to be included. Default is 0.
- `pvaictest`: Specifies a p-value to generate a critical value for the AICC difference.

**Testing Procedure** The AIC-based testing procedure follows these steps:

1. **Initial Model**: Estimate the model without the specified regressors and compute  $AICC_{\text{without}}$ .
2. **Extended Model**: Add the regressors to the model, estimate it, and compute  $AICC_{\text{with}}$ .
3. **Calculate AICC Difference**: Compute  $\Delta AICC = AICC_{\text{with}} - AICC_{\text{without}}$ .
4. **Decision Rule**: Include the regressors if  $\Delta AICC + \Delta_{AIC} < 0$ , where  $\Delta_{AIC}$  is the value specified in `aicdiff`.

**Sequential Testing Order** When multiple regressors are specified in `aictest`, they are tested sequentially in the following order:

1. **Trading Day Regressors**
2. **Length-of-Month/Quarter or Leap Year Regressors**
3. **Holiday Regressors** (e.g., Easter)
4. **User-Defined Regressors**

#### Special Cases and Considerations

- **Window Length for Holidays**: The window length for holiday regressors like Easter is determined by the `variables` argument if already specified, or defaults to standard values (e.g.,  $w = 1, 8, 15$ ) during testing.
- **Compatibility**: Ensure that regressors specified in `aictest` can be included with other regressors in the model. For example, do not test `lom` if `td` is already in `variables`, as they cannot be specified together.

**Example Usage** Testing whether to include trading day and Easter regressors:

```
regression {  
  aictest = (td easter)  
}
```

In this example:

- The program tests models with and without trading day and Easter regressors.
- For Easter, it tests multiple window lengths and selects the one with the lowest AICC, provided it meets the inclusion criterion.
- Only regressors that improve the model according to the AICC difference are included.

### Using aicdiff and pvaictest

- **aicdiff**: Directly sets the critical value for the AICC difference. A positive value makes inclusion more stringent.
- **pvaictest**: Specifies a p-value (e.g., 0.01) to calculate  $\Delta_{AIC}$  based on the chi-squared distribution, considering the difference in degrees of freedom.

### Interpreting Results

After the AIC test procedure:

- Review the regressors included in the final model.
- Examine statistical significance and diagnostics to ensure the model's adequacy.
- Be cautious of overfitting, especially when including multiple regressors.

Regression variables are vital components in modeling time series data using the **X-13ARIMA-SEATS** program. By incorporating various types of regressors—such as trading day effects, holiday effects, length-of-period adjustments, outlier corrections, user-defined variables, and change of regime effects—analysts can capture a wide range of external influences on the series. The AIC-based regressor selection procedure aids in building parsimonious models by including only those variables that significantly improve the model's fit without unnecessary complexity.

Understanding and appropriately applying these regression variables enhances the accuracy of the RegARIMA model, leading to better forecasting and more reliable seasonal adjustments. In the context of specific economies, such as India, customizing holiday effect variables (e.g., modeling the Diwali effect) ensures that culturally significant events are accurately reflected in the analysis. By thoroughly considering the interactions among regressors and employing statistical testing procedures, analysts can develop robust models that provide valuable insights into the underlying patterns and trends in time series data.

## 5.2 Transformation

When performing RegARIMA, the program can apply several types of transformations to the data. These transformations stabilize variance and ensure that the time series meets the assumptions required for ARIMA modeling. The TRANSFORM spec allows the user to specify the type of transformation or to let the program automatically select the best transformation.

### 5.2.1 Power or Function Transform

The user can specify a power transformation, which applies the Box-Cox transformation to the data. Let  $Y_t$  be the original series and  $y_t$  be the transformed series. The Box-Cox transformation is defined as:

$$y_t = \begin{cases} \log(Y_t) & \text{if } \lambda = 0 \\ \lambda^2 + \frac{Y_t^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \end{cases}$$

By default,  $\lambda = 1$ , meaning no transformation is applied. This is specified using the **power** option in the TRANSFORM spec. Alternatively, the user can specify a function transformation. The **function = auto** option applies a test to check whether a log transformation or no transformation is more appropriate. The following functions can be specified:

#### Test for Checking Log Transform vs No Transform

The program tests whether a log transformation is appropriate using the following procedure:

1. A default ARIMA (0, 1, 1)(0, 1, 1) model is fit on both the untransformed and log-transformed series. All user-specified regressors are included in both models. If the user has specified a model in the ARIMA spec then that model is used instead of the default model.
2. The AICC value is calculated for both fitted models.

value	transformation	range for $Y_t$	equivalent power argument
none	$Y_t$	all values	power = 1
log	$\log(Y_t)$	$Y_t > 0$ for all $t$	power = 0
sqrt	$0.25 + 2(\sqrt{Y_t} - 1)$	$Y_t \geq 0$ for all $t$	power = 0.5
inverse	$2 - \left(\frac{1}{Y_t}\right)$	$Y_t \neq 0$ for all $t$	power = -1
logistic	$\log\left(\frac{Y_t}{1-Y_t}\right)$	$0 < Y_t < 1$ for all $t$	no equivalent

Table 1: Transformations Available Using the function Argument for Transform

3. If  $\text{AICC}_{\text{nolog}} - \text{AICC}_{\text{log}} \leq \text{AICC}_{\text{diff}}$ , then no transformation is favored. The default for  $\text{AICC}_{\text{diff}}$  is  $-2$ , meaning that negative values of  $\text{AICC}_{\text{diff}}$  favor the log transformation.

### Calculation of AICC

The AICC is a modified version of the Akaike Information Criterion (AIC) that adjusts for small sample sizes. Let  $Y_1, Y_2, \dots, Y_n$  be the original data. The AICC is calculated as:

$$\text{AICC}_N = -2L_N + 2n_p \left( \frac{N}{N - n_p - 1} \right)$$

where:

- $L_N$  is the log-likelihood of the model.
- $n_p$  is the number of parameters.
- $N = n - d - sD$  is the effective number of observations, where  $d$  is the non-seasonal differencing order,  $D$  is the seasonal differencing order, and  $s$  is the seasonal period (12 for monthly or 4 for quarterly data).

### Log-Likelihood Adjustment After Transformation

Assuming  $Y = (Y_1, Y_2, \dots, Y_N)'$  is the data after differencing and  $X$  is the regression matrix we have

$$\text{AICC}_N = -2L_N + 2n_p \left( \frac{1 + n_p}{N} \right)$$

$$L_N = L_N(\phi, \theta, \Phi, \Theta, \sigma^2, \beta | Y, X)$$

is the log likelihood of the data from the regARMA model:

$$L_N(\phi, \theta, \Phi, \Theta, \sigma^2 | Y, X) = -\frac{1}{2} \log(2\pi|\Omega|) - \frac{1}{2} (Y - X\beta)' \Omega^{-1} (Y - X\beta)$$

$\Omega(\phi, \theta, \Phi, \Theta, \sigma^2, \beta)$  is the covariance matrix of the data  $Y$  following the  $ARIMA(p, d, q)(P, D, Q)_s$  process with Gaussian errors. If a log transformation is applied, the log-likelihood must be adjusted to account for the transformation. The log-likelihood after transformation is calculated as:

$$L_N^* = \tilde{L}_N + \sum \log \left| \frac{df(Y_t)}{dY_t} \right|$$

where:

- $\tilde{L}_N$  is the log-likelihood for the transformed data.
- The summation term adjusts for the Jacobian of the transformation.

This adjustment ensures that the log-likelihoods of the transformed and untransformed models are comparable, which is important for calculating information criteria like AICC and BIC.

*Proof for Log-Likelihood Adjustment*

Let  $X \sim F(x|\theta)$  be a random variable, and let  $Y = f(X)$ . If  $f_X(x|\theta)$  is the probability density function (pdf) of  $X$ , then the pdf of  $Y$  is given by:

$$f_Y(y) = f_X(f^{-1}(y)|\theta) \cdot \left| \frac{dx}{dy} \right|$$

Taking the logarithm:

$$\log f_Y(y|\theta) = \log f_X(x|\theta) + \log \left| \frac{dx}{dy} \right|$$

$$L(\theta|X) = L(\theta|Y) + \log \left| \frac{df(X)}{dX} \right|$$

For i.i.d observations, the log-likelihood is adjusted as follows:

$$L(\theta|X = (X_1, X_2 \dots X_N)') = L(\theta|Y = (Y_1, Y_2 \dots Y_N)') + \sum_{i=1}^n \log \left| \frac{df(X_i)}{dX_i} \right|$$

### 5.2.2 Length of Month (LoM) and Length of Quarter (LoQ) Adjustments

For monthly or quarterly series, users can specify adjustments for the length of the month (LoM) or the length of the quarter (LoQ). Many time series, such as GDP, are cumulative sums over time, so longer months or quarters may have larger values. These adjustments correct for this deterministic effect. The LoM adjustment factor is calculated as:

$$\text{LoM}_t = \frac{m_t - \bar{m}}{\bar{m}}$$

where

- $m_t$  is the number of days in month  $t$ .
- $\bar{m} = 30.4375$  is the average length of a month, accounting for leap years.

The Leap year correction factors for each time point (for monthly data) are calculated as follows: Note that 28.25 is the average length of February.

$$\text{LeapYear}_t = \begin{cases} \frac{28}{28.25} & \text{for 28-day months} \\ \frac{29}{28.25} & \text{for 29-day months (leap year February)} \\ 1 & \text{for all other months} \end{cases}$$

### 5.2.3 User-Specified Adjustments

Users can also provide custom adjustment factors for each time point using the `mode` option in the `TRANSFORM` spec. If `mode` is set to `ratio`, the adjustment factors  $C_1, C_2, \dots, C_N$  are provided by the user, and the adjustment is performed as follows:

$$f(x_i) = \frac{x_i}{C_i}$$

or

$$f(x_i) = x_i - C_i$$

if `mode` is set to `diff`.



### 5.3 Outlier Detection

[13] [12] [4] [5] The **X13-ARIMA-SEATS** program provides capabilities for detecting and adjusting various types of outliers. The program can detect the following types of outliers:

1. **Additive Outlier (AO)**: This is an unexpected high or low value at a given time point.
2. **Level Shift Outlier (LS)**: This occurs when the mean of a series changes abruptly.
3. **Temporary Change (TC)**: This occurs when the series mean changes temporarily and shows a decay to the previous mean.

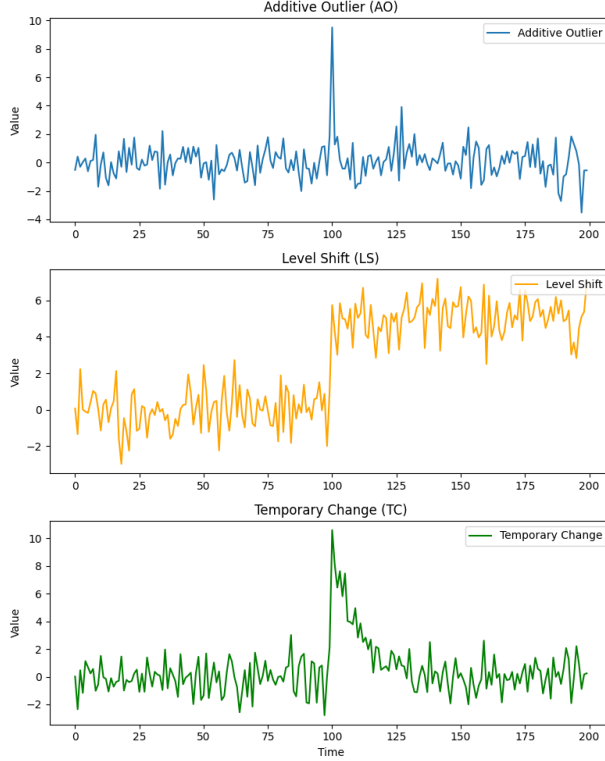


Figure 4: Visual Representation of Outlier Types: Additive Outlier (AO), Level Shift (LS), and Temporary Change (TC)

An outlier at a time point  $t_0$  can be detected using regression variables. Additive outliers at  $t_0$  can be detected as follows:

Suppose  $Y = (Y_1, Y_2, \dots, Y_N)^T$  is our time series with outliers that follows a regARIMA  $(p, d, q)(P, D, Q)_s$  process. Let  $X_t$  be the vector of regression variables for the  $t$ -th observation. The model can be written as:

$$Y_t = X_t^T \beta + Z_t$$

where  $Z_t$  follows an ARIMA  $(p, d, q)(P, D, Q)_s$  process.

To detect an outlier, say an additive outlier at time  $t_0$ , we add a new regression variable  $AO_t^{(t_0)}$  to the  $X_t$  vector, so:

$$\tilde{X}_t^T = [X_t^T \quad AO_t^{(t_0)}]^T$$

where  $AO_t^{(t_0)}$  is defined as:

$$AO_t^{(t_0)} = \begin{cases} 1, & \text{if } t = t_0 \\ 0, & \text{if } t \neq t_0 \end{cases}$$

This regressor contributes to  $Y$  only at time point  $t_0$ . The equation for the time series with an additive outlier (AO) included becomes:

$$Y_t = \tilde{X}_t^T \tilde{\beta} + \tilde{Z}_t$$

Now, we can fit the new model and check if the coefficient corresponding to the regressor  $AO_{(t_0)}$  is significantly different from 0. The program calculates the  $t$ -value for the regression coefficient and compares it with a critical value. The critical values are taken from a table, which contains the critical values used for different values of  $N$ . This table is based on the researcher's experience. Similarly, the other two types of outliers can be modeled using such regressors.

- **Level Shift (LS):** We can define a regressor as follows:

$$LS_{(t_0)}^t = \begin{cases} -1, & t < t_0 \\ 0, & t \geq t_0 \end{cases}$$

- **Temporary Shift Change (TC):** We define a regressor as follows:

$$TC_{(t_0)}^t = \begin{cases} 0, & t < t_0 \\ \alpha^{(t-t_0)}, & t \geq t_0 \end{cases}$$

where  $\alpha$  is called the decay rate.

$L_t^{(a)}$  signifies that the mean before time  $t_0$  was less than the mean after time  $t_0$ . The critical decay rate for  $TC_{(t_0)}$  can be specified by the user. The default value is  $\alpha = (0.7)^{1/s}$  ( $s$  is the number of observations in a year). The regressors in the regression matrix may look like this:

	AO1953.02	LS1953.04	TC1953.03
1953.01	0	-1	0
1953.02	1	-1	0
1953.03	0	-1	1
1953.04	0	0	0.5
1953.05	0	0	0.25
1953.06	0	0	0.125

here  $\alpha = 0.5$ .

When an outlier is detected, it can be adjusted by using the corresponding regressor in the model. The user can specify the type of outlier they want the program to detect automatically using the **type** argument of the **outlier\_spec**.

```
outlier{
  types= (ls ao tc)
}
```

The default for types argument in the outlier spec is **types = (ls ao)**, but the user can provide any combination of ao, ls, and tc.

The automatic outlier detection procedure discussed here is used to detect only Additive Outliers (AO), Level Shifts (LS), and Temporary Changes (TC) but **X13-ARIMA-SEATS** also provides functionality to check for other types of outliers at specific time points provided by the user. For instance, if a user suspects a temporary level shift from time  $t_0$  to  $t_1$ , indicating that the mean of the series changes temporarily between these points, they can specify the corresponding time range for the regressor in the regression specification as follows:

```
regression{variables = ts1952.03-1954.06}
```

This approach differs from the automatic outlier detection procedure because the specific time points for potential outliers must be predefined by the user. In contrast, the automatic detection process tests for the presence of a given type of outlier at all possible time points.

There are two algorithms available for automatic outlier detection:

1. **Addone Method:** Tests one potential outlier at a time.
2. **AddAll Method:** Tests all potential outliers at once.

The default method is **Addone**.

Regression effect	Variable definition(s)
<b>Seasonal Outlier at <math>t_0</math></b> sodate $_{t_0}$	$SO_t^{(t_0)} = \begin{cases} 0, & \text{for } t \geq t_0 \\ -1, & \text{for } t < t_0 \\ 1/(s-1), & \text{otherwise} \end{cases}$
<b>Ramp, to <math>t_1</math></b> rpdate $_{t_0}$ -date $_{t_1}$	$RP_t^{(t_0, t_1)} = \begin{cases} t_0 - t_1, & \text{for } t \leq t_0 \\ t - t_1, & \text{for } t_0 < t < t_1 \\ 0, & \text{for } t \geq t_1 \end{cases}$
<b>Quadratic Ramp (Increasing), to <math>t_1</math></b> qidate $_{t_0}$ -date $_{t_1}$	$QI_t^{(t_0, t_1)} = \begin{cases} -(t_1 - t_0)^2, & \text{for } t \leq t_0 \\ (t - t_0)^2 - (t_1 - t_0)^2, & \text{for } t_0 < t < t_1 \\ 0, & \text{for } t \geq t_1 \end{cases}$
<b>Quadratic Ramp (Decreasing), to <math>t_1</math></b> qddate $_{t_0}$ -date $_{t_1}$	$QD_t^{(t_0, t_1)} = \begin{cases} -(t_1 - t_0)^2, & \text{for } t \leq t_0 \\ -(t - t_0)^2 + (t_1 - t_0)^2, & \text{for } t_0 < t < t_1 \\ 0, & \text{for } t \geq t_1 \end{cases}$
<b>Temporary Level Shift, to <math>t_1</math></b> tl1date $_{t_0}$ -date $_{t_1}$	$TL_t^{(t_0, t_1)} = \begin{cases} 1, & \text{for } t_0 \leq t \leq t_1 \\ 0, & \text{otherwise} \end{cases}$
<b>Additive Outlier Sequence, to <math>t_1</math></b> aosdate $_{t_0}$ -date $_{t_1}$	This adds a sequence of AO variables (as defined previously) beginning at $t_0$ and ending on $t_1$ . For example, aos1950.jan-1950.oct is equivalent to listing ao1950.jan ao1950.feb ao1950.mar ... ao1950.oct individually. For convenience, 0.0 represents the end of the series; e.g., aos2020.jan-0.0 would add a sequence of AO variables beginning at January 2020.
<b>Level Shift Sequence, to <math>t_1</math></b> lssdate $_{t_0}$ -date $_{t_1}$	The level shift counterpart to AOS, this adds a sequence of LS variables (as defined previously) beginning at $t_0$ and ending on $t_1$ .

Table 2: Outlier regression effects and their variable definitions

### 5.3.1 Outlier Flagging Process

Let  $Y = Y_1, \dots, Y_N$  represent the time series, and let

$$X = \begin{bmatrix} X_1^T \\ X_2^T \\ \vdots \\ X_N^T \end{bmatrix}$$

represent the original regression matrix. Suppose we are detecting level shifts and additive outliers. The process of outlier flagging involves adding a regression variable corresponding to an outlier at a given time point to the model and fitting the model to find parameter estimates. The  $t$ -value for an additive outlier  $AO_{(t_0)}$  is defined as:

$$t_{AO(t_0)} = \frac{\beta_{AO(t_0)}}{\sqrt{\text{Var}(\beta_{AO(t_0)})}}$$

where  $t_{AO(t_0)}$  is the  $t$ -value for the regression coefficient  $\beta_{AO(t_0)}$  corresponding to the regressor  $AO^{(t_0)}$ . If  $|t_{AO(t_0)}|$  exceeds the critical value, the point is flagged as an outlier. Proceeding with the detection at the next time point, we add  $AO^{(t_0+1)}$  to the original set of regressors  $X_t$  and follow the same process. This process is repeated across all time points and outlier types to make a list of flagged outliers. If during this process the regression matrix becomes singular we ignore it and move to the next time point.

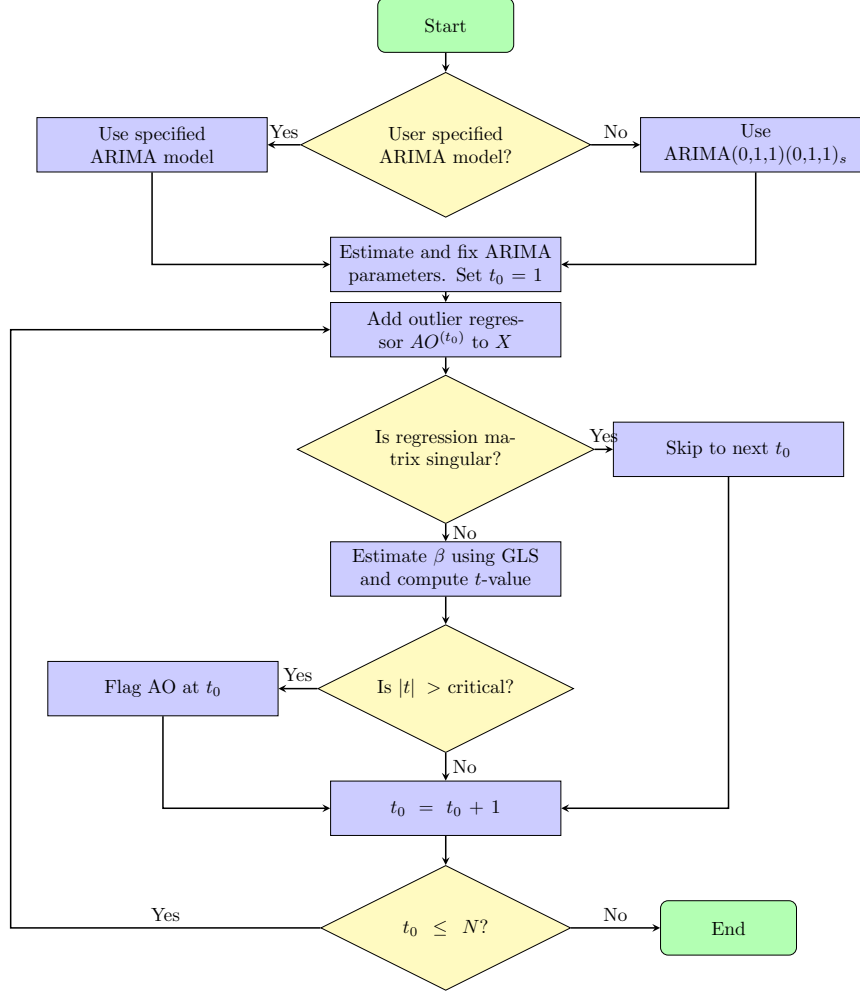


Figure 5: Flowchart of the Additive Outlier Flagging Process

In outlier detection, fitting the model repeatedly for all outliers can be time- and resource-intensive. To address this, the program employs the following strategy:  
We aim to fit the model:

$$Y_t = X\beta + Z_t$$

where  $Z_t \sim \text{ARIMA}(p, d, q)(P, D, Q)$ .

This involves estimating two sets of parameters:

- $\beta$ : the regression parameters
- Parameters of the ARIMA process

If the user has specified a particular ARIMA  $(p, d, q)(P, D, Q)$  model, it is used to estimate the parameters of the ARIMA part in  $Y_t = X\beta + Z_t$ , otherwise the default model ARIMA(0, 1, 1)(0, 1, 1)<sub>s</sub> is used. The model parameters are then fixed at the estimated values. During the flagging process, when new regression variables are added, we calculate only the regression parameter estimates,  $\beta$ , using Generalized Least Squares (GLS) estimation as follows: The log-likelihood maximization reduces to a weighted least squares estimation:

$$L_N = -\frac{1}{2} \log(2\pi\Omega) - \frac{1}{2} (Y - X\beta)^T \Omega^{-1} (Y - X\beta)$$

Maximizing  $L_N$  is equivalent to minimizing:

$$\min_{\beta} (Y - X\beta)^T \Omega^{-1} (Y - X\beta)$$

Since  $\Omega$  depends only on the parameters of the ARIMA model (which are fixed), there is a closed-form solution for  $\beta$ :

$$\hat{\beta} = (X^T \Omega^{-1} X)^{-1} X^T \Omega^{-\frac{1}{2}} Y$$

where  $\Omega^{-\frac{1}{2}}$  is a matrix such that  $(\Omega^{-\frac{1}{2}})(\Omega^{-\frac{1}{2}}) = \Omega^{-1}$ , and  $\Omega^{-\frac{1}{2}}$  exists if  $\Omega$  is positive definite.

### 5.3.2 AddOne Method

It is the default method of outlier detection which consists of identifying outliers and adding the most significant one to the model. This process is repeated till no outliers are left. The following is an outline of the algorithm.

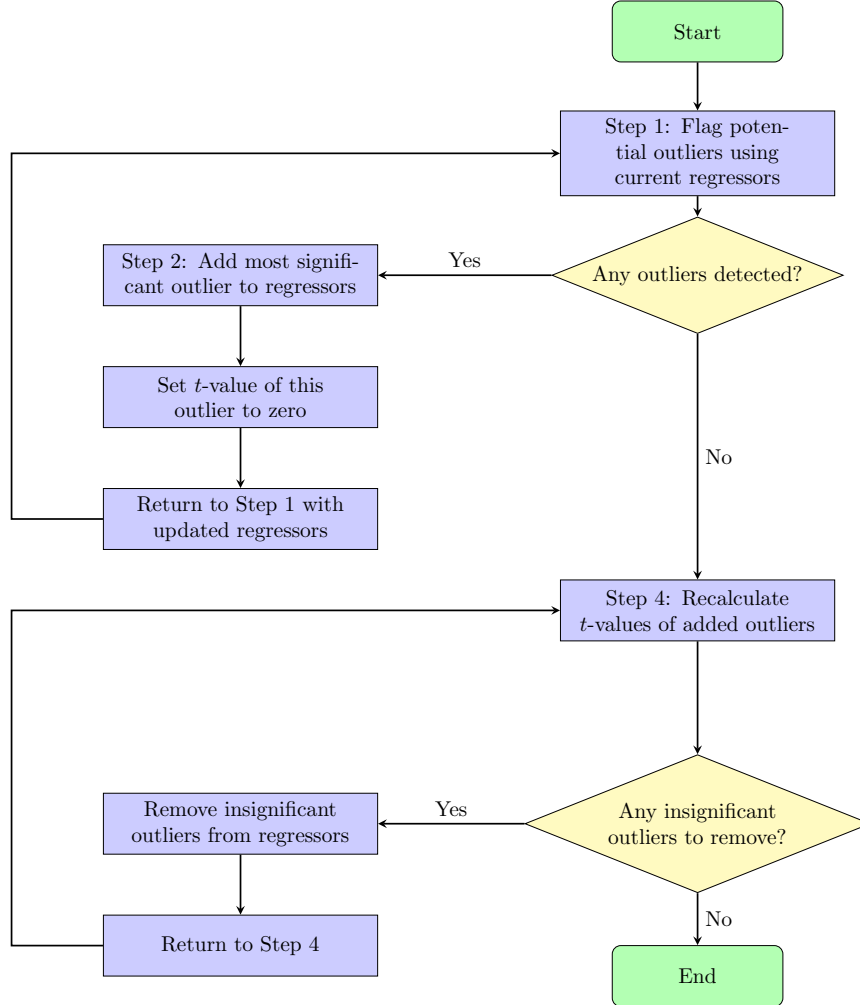


Figure 6: Flowchart of the AddOne Method for Outlier Detection

1. **Step 1:** Flag all potential outliers using the current set of regression variables.
2. **Step 2:** Identify the most significant outlier and add it to the list of regression variables. This outlier remains in the regression matrix for subsequent iterations.
3. **Step 3:** If no new outliers are detected, proceed to Step 4. Otherwise, return to Step 1 with the updated list of regressors. Note that in subsequent iterations, the  $t$ -value for this flagged outlier will be set to zero to prevent reflagging.

4. **Step 4:** Now that no new outliers are identified, we remove previously added outliers that have become insignificant. The model is refitted, and  $t$ -values for each previously flagged outlier are recalculated. Outliers with  $t$ -values below the critical threshold are flagged for removal.
5. **Step 5:** If outliers flagged in Step 4 are found, remove them from the set of regressors and return to Step 4. Otherwise, the process stops.

### 5.3.3 AddAll Method

The steps in the **AddAll** method are the same as those in the **AddOne** method, except that during the forward pass, all flagged regressors are added to the regression variable list rather than only the regressor with the highest  $t$ -value. However, in the backward pass, flagged outliers are still removed one at a time.

#### Remarks

1. Different methods may result in different sets of outliers. The choice of ARIMA model may also affect the outliers detected. One should be aware that certain combinations of outliers produce arithmetically equivalent effects. For example, the following are equivalent:

- (i) An Additive Outlier (AO) at time  $t_0$  followed by a Level Shift (LS) at  $t_0 + 1$ .
- (ii) Level Shifts (LS) at both  $t_0$  and  $t_0 + 1$ .
- (iii) Both an AO and an LS at  $t_0$ .

However, an LS at  $t_0$  followed by an AO at  $t_0 + 1$  is not equivalent to these other combinations. Because during seasonal extraction AOs are assigned to the irregular component and LSs to the trend-cycle, one might prefer one combination of equivalent outliers over another based on the intended interpretation.

2. Certain outliers cannot be distinguished or calculated at specific data points:
  - An LS at the first data point cannot be estimated since the level of the series prior to the given data is unknown. Therefore, no LS test statistic is calculated for the first data point.
  - An LS at the last data point cannot be distinguished from an AO there, and an LS at the second data point cannot be distinguished from an AO at the first data point. Hence, LS statistics are calculated for the second and last data points only if AOs are not also being detected.
  - A temporary change (TC) outlier at the last data point cannot be distinguished from an AO there, so no TC statistic is calculated for the last data point if an AO is also being detected.

LS and TC test statistics that are not calculated due to these limitations are set to zero during the flagging process.

3. When a model contains two or more level shifts (including those obtained from outlier detection as well as any specified in the regression spec), **X-13ARIMA-SEATS** can optionally produce  $t$ -statistics for testing the null hypothesis that each sequence of two, three, etc., successive level shifts cancels to yield a net effect of zero beyond the last level shift in the sequence. The  $t$ -statistics are computed as the sums of the estimated parameters for each sequence of successive level shifts divided by the appropriate standard error. An insignificant  $t$ -statistic (e.g., one less than 2 in magnitude) fails to reject the null hypothesis that the corresponding level shifts offset each other.
  - Two successive level shifts cancel if the sum of their corresponding regression parameters is zero, which can be re-expressed as a temporary level shift starting at the time of the first level shift and ending one time point before the second level shift.
  - Similarly, three successive level shifts cancel if the sum of their three regression parameters is zero, and these can be re-expressed as two adjacent temporary level shifts.

There is a user-specified limit on the number of successive level shifts in the sequences tested. These cancellation tests help users assess the impacts of level shifts in a model. If one or more of these  $t$ -statistics are insignificant, the user might consider re-specifying the model with the relevant level shift regression variables replaced by appropriate temporary level shift variables.

4. During the forward pass, a robust estimate of  $\sigma$  is used, which is calculated as follows: [4] [13] [9]

$$\hat{\sigma} = 1.48 \times \text{median}\{|Z_t - \tilde{Z}_t| : t \in [1, 2, \dots, N]\}$$

where  $Z_t$  represents the observed residuals, and  $\tilde{Z}_t$  are the fitted values. This robust estimate helps to limit the influence of outliers during the initial outlier detection process.

5. During the backward pass, the usual mean square error (MSE) is used to estimate  $\sigma$ :

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{t=1}^N (Z_t - \tilde{Z}_t)^2$$

This approach provides a standard estimate of variance once the initial set of outliers has been flagged and potentially mitigates the effects of outliers on the variance estimate.

## 5.4 Automatic Model Selection

[10] [13] [7] The `automdl` specification in `X13-ARIMA-SEATS` allows for automatic model selection based on a modified version of the TRAMO program developed by Gomez and Maravall. The procedure can be outlined as follows:

1. **Default Model Estimation:** A default model, usually the airline model  $\text{ARIMA}(0\ 1\ 1)(0\ 1\ 1)_s$ , is estimated. Initial outlier detection, regressor significance tests, and residual diagnostics are performed.
2. **Differencing Order Identification:** Empirical unit root tests are conducted to determine the appropriate differencing orders for the model.
3. **ARMA Model Order Identification:** An iterative procedure is applied to determine the orders of the AR and MA parameters.
4. **Comparison of Identified Model with Default Model:** The identified model is compared with the default model using diagnostic measures.
5. **Final Model Checks:** The chosen model undergoes further diagnostics to ensure adequacy.

### 5.4.1 Default Model Selection

The first step of the automatic outlier procedure is to estimate a default model. For monthly and quarterly series, this is initially an *airline* model:  $\text{ARIMA}(0\ 1\ 1)(0\ 1\ 1)_s$ .

The default model is used to perform several tasks:

- If tests for trading day, Easter, or user-defined regressors are requested by the user in the regression specification, an initial check for the significance of these effects is performed using the default model.
- The `X-13ARIMA-SEATS` program's `aictest` option is utilized to assess the significance of the regressors AICC criteria discussed before.

The procedure then checks the significance of including a constant term in the `regARIMA` model. This is done by fitting the model without a constant term and a  $t$ -statistic for the mean of the model residuals is generated and compared against a critical value of 1.96. Once these tests are complete, the program performs automatic outlier identification if specified by the user in the outlier specification. After outlier identification, the trading day, Easter, and constant regressors are reassessed for significance:

- $t$ -tests are generated.



- A critical value of 1.96 is used to determine if the regressors are significant, except for the constant regressor, which uses the value specified in `armalimit` argument of the `automdl` spec. Default value of `armalimit` is 1.0.
- For the trading day regressor, at least one of the regressors must have a critical value greater than 1.96.

Note that this test is conducted for trading day and Easter regressors only if the `aictest` argument is provided in the regression specification; the constant regressor is always tested.

After determining the regression part of the default model, the program generates residual diagnostics for this model, which include:

- The Ljung-Box  $Q$  statistic for the model residuals (at lag 24 for monthly series or lag 16 for quarterly series).
- The confidence coefficient of the Ljung-Box  $Q$  statistic.
- A  $t$ -value for the mean of the regARIMA model residuals.
- An estimate of the residual standard error.

The confidence coefficient is defined as  $1 - p$ -value of the Ljung-Box  $Q$  statistic, as described in Lehman (1986). The TRAMO documentation (Gómez and Maravall, 1996) refers to the confidence coefficient as the significance level. These diagnostics will later be compared to those of the model selected by the automatic model identification procedure. The model identified by this procedure must show some improvement over the default model in these residual diagnostics; otherwise, the program will accept the default model.

Just before the model identification phase begins, the program removes the regression effects estimated by the default model from the original series. It is this adjusted series, rather than the original series, that is used in the model identification routines. This approach aims to robustify the model identification process, ensuring that the choice of differencing and model orders are not unduly affected by outliers, calendar effects, and other regression effects. In the TRAMO documentation, this adjusted series is referred to as the *linearized series*.

#### 5.4.2 Identification of Differencing Orders

The purpose of differencing is to make the series stationary. We want to find the appropriate order of differencing so that the differenced series becomes stationary. In the model equation of ARIMA differencing order is represented by a differencing polynomial on the left hand side of the following form:

$$\phi(B)\Phi(B)\delta(B)z_t = \theta(B)\Theta(B)\epsilon_t$$

$$\delta(B) = (1 - B)^d(1 - B^s)^D$$

Notice that the roots of these polynomials play a very important role. The conditions for stationarity and invertibility for an ARIMA process are checked using the roots. The ARMA part of the model is stationary if all the roots of the AR polynomials i.e.  $\phi(B)$  and  $\Phi(B)$  have a modulus greater than 1. If the modulus of a root of the AR polynomials is equal to 1 or very close to 1 then it suggests that that factor should be a part of the differencing polynomial, not the AR polynomials. (The AR roots can never have modulus less than one as the series would explode). This means if we fit an AR(2) model to a series and one of the roots of the AR polynomial comes out to be 1 or close to 1 then we should rather fit an ARIMA(1, 1, 0) model. This will ensure that the unit root is incorporated by the differencing and the ARMA part remains stationary. Following this spirit the program attempts to identify an appropriate order of differencing for the linearized series computed earlier. The maximum permissible values of regular and seasonal differencing are 2 and 1 respectively. This is achieved by performing a series of unit root tests and fitting different ARMA models to the (possibly differenced) linearized series. The estimation of these models utilizes the Hannan-Rissanen method.

**Step 1:** The first stage of the procedure involves fitting an  $\text{ARIMA}(2, 0, 0)(1, 0, 0)_s$  model to the linearized series using the Hannan-Rissanen method. The real AR

roots of the estimated model are then examined. A root is considered a unit root if the modulus of the root is less than 1.042. If such a root is identified, the corresponding order of differencing (seasonal or nonseasonal) is increased by one. If the Hannan-Rissanen procedure estimates a model with roots inside the unit circle, the X-13ARIMA-SEATS program re-estimates the model using exact maximum likelihood estimation. The modulus test described above is then applied to the resulting estimates to determine the necessity of additional differencing. **Step 2:** If differencing was identified in Step 1, the linearized series is differenced accordingly at the start of Step 2. An ARMA  $(1, 1)(1, 1)_s$  model is then fitted to the differenced series. The roots of the AR polynomial of this model are examined to determine if they are close to one. If no root close to 1 is found then we are done and the differencing identified in step one is chosen for the final model.

1. If an AR coefficient meets the criterion of being close to one, the program checks for a common factor in the corresponding AR and MA polynomials of the ARMA model that can be canceled.
2. If no such cancellation exists, the differencing order is altered by increasing the appropriate differencing order. The linearized series is then differenced using the new set of differencing orders.
3. The ARMA $(1, 1)(1, 1)_s$  model is re-fitted to the newly differenced series, and the program checks for any additional necessary differencing.
4. This process repeats iteratively until no further differencing is required.

Once the differencing orders are established, a  $t$ -statistic for the mean term of the fully differenced series is generated. This statistic is based on either the sample mean (if no differencing is identified) or by adding a constant term to the regARIMA model. The critical value for the test is determined based on the number of observations in the series. This step determines whether we will use an intercept term or not.

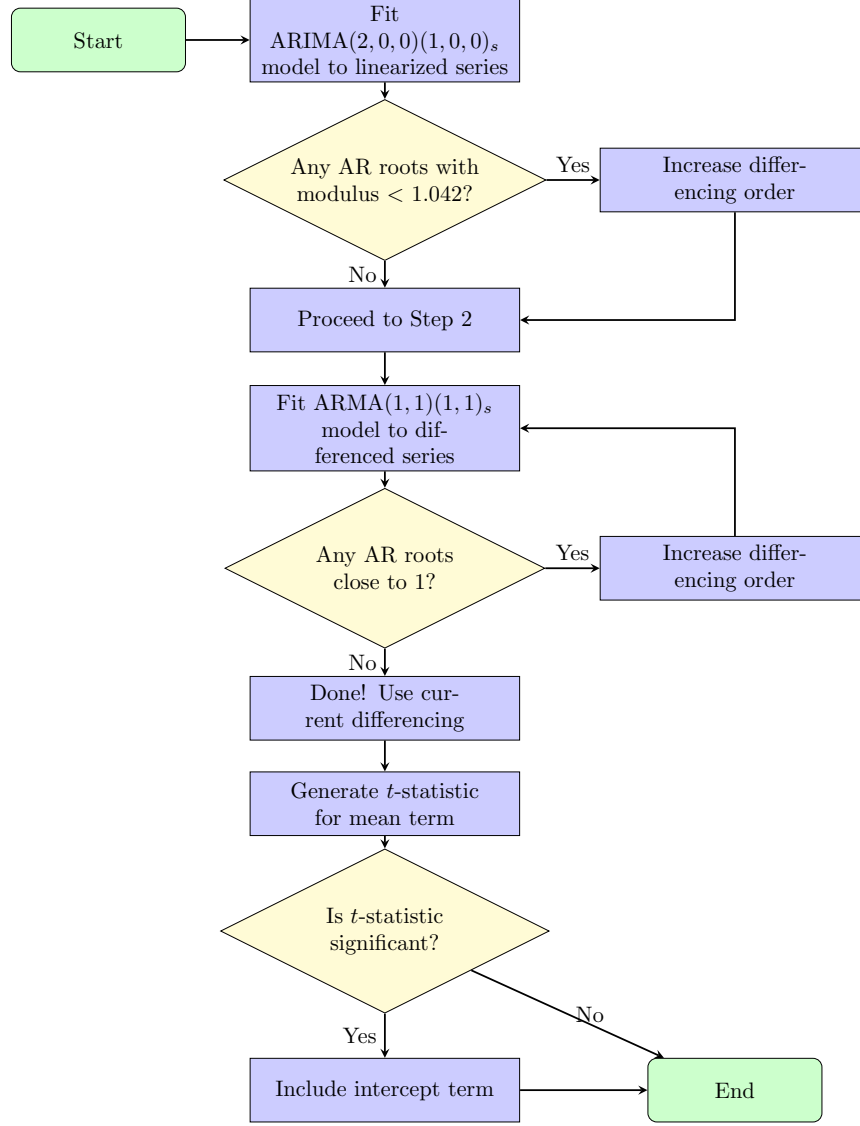


Figure 7: Flowchart for Identifying Appropriate Differencing Orders in ARIMA Modeling

### 5.4.3 Identification of ARMA Model Orders

With the appropriate differencing orders determined, the next step involves identifying the orders of the ARMA model components. This process aims to select the best-fitting ARMA model by comparing various candidate models based on their Bayesian Information Criterion (BIC) values. The BIC is a criterion for model selection among a finite set of models. It is based on the likelihood function and includes a penalty term for the number of parameters in the model to discourage overfitting. The classical BIC formula used in the X-13ARIMA-SEATS output is:

$$\text{BIC} = -2L_N + n_p \cdot \ln N,$$

where:

- $L_N$  is the maximized value of the log-likelihood function evaluated over  $N$  observations.
- $n_p$  is the number of estimated parameters in the model, including the white noise variance.
- $N$  is the number of observations remaining after applying the model's differencing operations.

For the automatic model identification procedure, a variant of BIC used by TRAMO is employed:

$$\text{BIC2} = \frac{-2L_N + n_p \cdot \ln N}{N}.$$

This adjusted criterion allows for a fair comparison of models with different numbers of parameters and observations. The identification process consists of a three-stage procedure, as detailed in Gomez and Maravall (2000) [7], to efficiently explore potential ARMA model orders up to specified maximum orders for regular and seasonal components.

#### Stage 1: Initial Seasonal Order Selection

1. **Model Specification:** Consider ARIMA models of the form  $(3, d, 0)(P, D, Q)_s$ , where:
  - $d$  and  $D$  are the previously determined regular and seasonal differencing orders, respectively.
  - $P$  and  $Q$  range from 0 up to the maximum seasonal order  $m_s$  (default is 1).
  - The regular AR order is fixed at 3, and the regular MA order is fixed at 0 for initial estimation.
2. **Model Estimation:** For each combination of  $P$  and  $Q$ , estimate the model parameters using the exact maximum likelihood.
3. **BIC2 Calculation:** Compute the BIC2 value for each model.
4. **Selection of Seasonal Orders:** Select the pair  $(P, Q)$  that results in the lowest BIC2 value.

#### Stage 2: Nonseasonal Order Selection

1. **Model Specification:** Using the selected  $P$  and  $Q$  from Stage 1, consider ARIMA models of the form  $(p, d, q)(P, D, Q)_s$ , where:
  - $p$  and  $q$  range from 0 up to the maximum regular order  $m_r$  (default is 2).
2. **Model Estimation:** For each combination of  $p$  and  $q$ , estimate the model parameters.
3. **BIC2 Calculation:** Compute the BIC2 value for each model.
4. **Selection of Nonseasonal Orders:** Select the pair  $(p, q)$  that results in the lowest BIC2 value.

#### Stage 3: Refinement of Seasonal Orders

1. **Model Specification:** With the selected  $p$  and  $q$  from Stage 2, reconsider ARIMA models of the form  $(p, d, q)(P, D, Q)_s$ , varying  $P$  and  $Q$  within the specified seasonal order limits (from 0 to  $m_s$ ).
2. **Exception:** If no seasonal AR term ( $P = 0$ ) was selected in Stage 1 and seasonal differencing is present ( $D > 0$ ), only  $P = 0$  is considered, and  $Q$  is varied.
3. **Model Estimation:** Estimate the model parameters for each combination of  $P$  and  $Q$ .
4. **BIC2 Calculation:** Compute the BIC2 value for each model.
5. **Selection of Seasonal Orders:** Select the pair  $(P, Q)$  that results in the lowest BIC2 value.

#### Model Selection and Final Checks

During the ARMA order selection process, X13-ARIMA-SEATS keeps track of the models with the five smallest BIC2 values. Once the identification phase is over, the program compares the BIC2 of the best model with those of the other four models to determine if there are models with BIC2s that are "close enough" such that there is no significant difference between the models. The criteria for being "close enough" depend on several factors, including the length of the series and the magnitude of the differences between the BIC2 values. (No further information about this was available.) If the program finds a model that is close enough to the best model, it also checks whether the alternative model is more parsimonious, especially in the seasonal operator, than the best model. Parsimony refers to the simplicity of the model—the model

with fewer parameters is considered more parsimonious. If the alternative model is more parsimonious, the program will accept it over the model with the lowest BIC2.

The program also checks for *model balance*. A model is said to be more balanced than a competing model if the absolute difference between the total orders of the AR (including differencing) and MA operators is smaller. In mathematical terms, for a model with AR order  $p$ , differencing order  $d$ , and MA order  $q$ , the total orders are  $p + d$  and  $q$ , respectively. A balanced model minimizes  $|(p + d) - q|$ . While balanced models are useful for model-based seasonal adjustment, it is unclear whether this criterion is beneficial for the types of operations performed by **X13-ARIMA-SEATS**. This is because it can introduce a small bias toward mixed ARMA models, which can be difficult to estimate due to near cancellation of terms. **X13-ARIMA-SEATS** makes checking for model balance optional at this stage; by default, it does not test for model balance.

If the identified model is different from the default model, the program redoes several steps that determined the regressors of the default model:

- **Outlier Regressors:** Outlier regressors identified for the default model are removed from the identified model. Outlier detection is then re-performed using the identified model to ensure that any new outliers are appropriately accounted for.
- **Regressor Testing:** If the user has specified Akaike Information Criterion (AIC) testing of trading day effects, Easter effects, or user-defined regressors, this testing is redone using the identified model. This ensures that the inclusion or exclusion of these regressors is optimal for the new model.

Subsequently, outlier identification is redone for the identified model. By repeating these steps, the program ensures that all aspects of the model specification are consistent and optimized for the selected ARIMA structure.

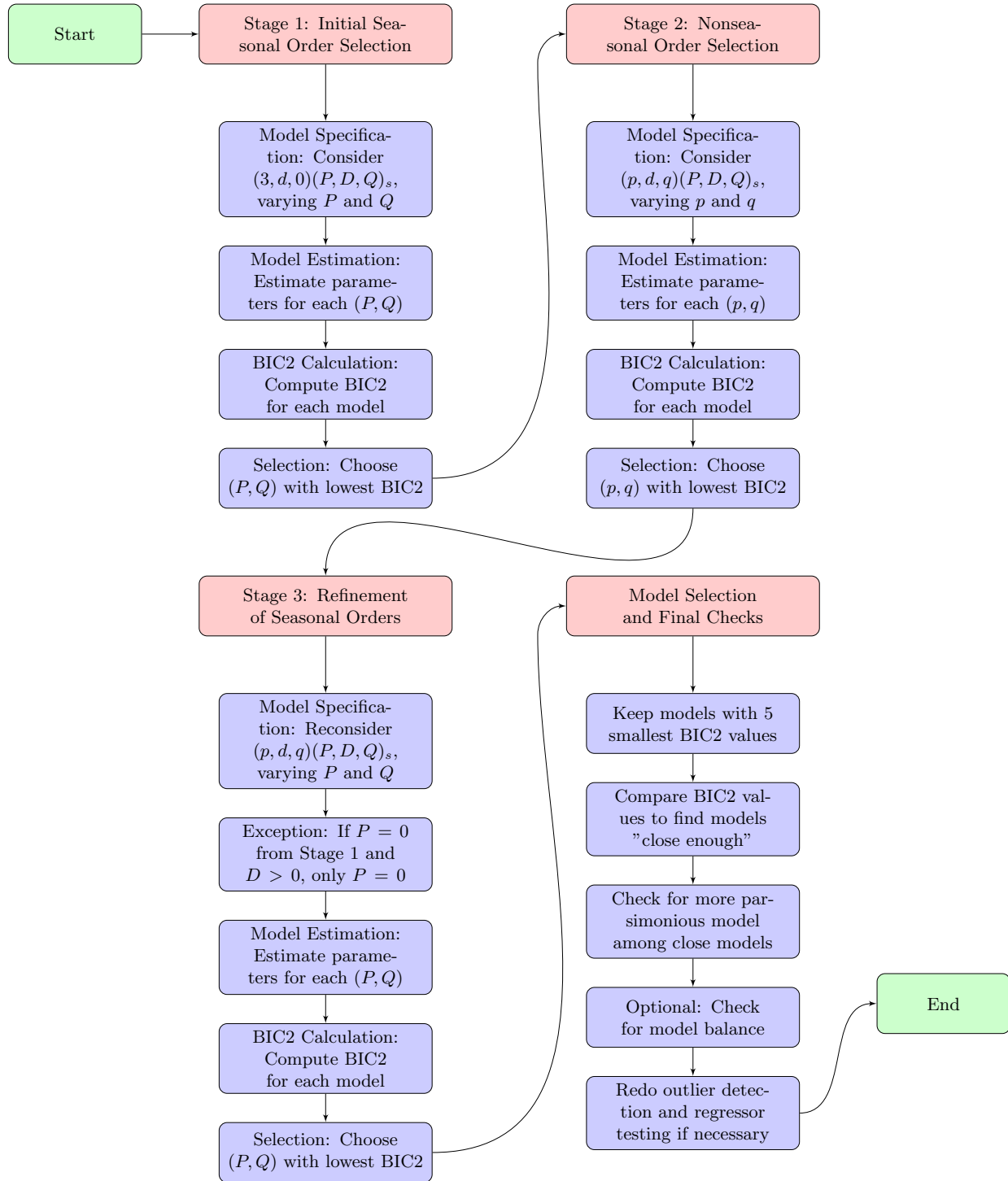


Figure 8: Flowchart for Identification of ARMA Model Orders

#### 5.4.4 Final Selection of the ARIMA Model

After identifying potential ARIMA models, the next crucial step is to compare the automatically identified model with the default model to ensure the most suitable model is selected for analysis. This comparison involves rigorous diagnostic checks and adherence to specific criteria that consider model parsimony and

statistical adequacy.

### [13] Comparison of Identified Model with Default Model

If the automatically identified model differs from the default model, a detailed comparison of residual diagnostics is conducted. The key statistics used for this comparison are:

- $Q_A$ : Confidence coefficient of the Ljung-Box Q statistic for the automatically identified model (evaluated at lag 24 for monthly series or lag 16 for quarterly series).
- $Q_D$ : Confidence coefficient of the Ljung-Box Q statistic for the default model.
- $RSE_A$ : Residual standard error of the automatically identified model.
- $RSE_D$ : Residual standard error of the default model.

The default model is preferred over the automatically identified model if the following conditions are satisfied:

1. The number of outliers identified in the default model is less than or equal to that in the automatically identified model, *and*
2. Any one of the following holds:
  - (a)  $Q_A < 0.95$  and  $Q_D < 0.75$  and  $RSE_D < RSE_A$ .
  - (b)  $Q_A > 0.95$  and  $Q_D < 0.95$  (applicable only on the first pass).
  - (c)  $Q_A < 0.95$  and  $Q_D < 0.75$  and  $Q_D < Q_A$  and  $RSE_D < RSE_A \times 1.013$ .
  - (d)  $Q_A \geq 0.95$  and  $Q_D < 0.95$  and  $RSE_D < RSE_A \times 1.013$ .
  - (e) The automatic model is  $ARIMA(1, 0, 1)(0, 1, 1)_s$  or  $ARIMA(1, 0, 0)(0, 1, 1)_s$  and the estimated AR coefficient  $\phi_1 \geq 0.82$ .
  - (f) The automatic model is  $ARIMA(0, 1, 1)(1, 0, 1)_s$  or  $ARIMA(0, 1, 1)(1, 0, 0)_s$  and the estimated seasonal AR coefficient  $\phi_s \geq 0.65$ .

These conditions prioritize models that have better residual diagnostics and fewer outliers while ensuring that the model is not overfitted.

### Model Acceptability and Adjustment

The acceptability of the preferred model is further evaluated using the confidence coefficient of the Ljung-Box Q statistic. If this coefficient exceeds a default threshold of 0.975, it suggests the presence of autocorrelation in the residuals, indicating potential model inadequacy. To address this, the program adjusts the critical value (CV) for automatic outlier detection by reducing it based on a user-specified parameter `reducecv`:

$$CV_r = (1 - \text{reducecv}) \times CV,$$

where  $CV_r$  is the reduced critical value. The adjusted critical value cannot be less than 2.8 to prevent excessive outlier detection. The program then re-estimates the model and re-identifies outliers using  $CV_r$ . If no outliers were previously identified, re-identification is performed without re-running the automatic model identification.

After re-estimation, diagnostics are generated for the revised model and compared to those of the previous preferred model. The Ljung-Box Q test is performed again, and if the confidence coefficient remains above 0.99, indicating persistent autocorrelation, the program defaults to a simplified model:

$$ARIMA(3, d, 1)(0, D, 1)_s,$$

and attempts outlier identification for this new model.

### Re-evaluation of Regressors

The program re-evaluates the significance of regressors such as trading day effects, Easter effects, and constant terms. T-statistics are calculated, and a critical value of 1.96 is used to determine significance (except for the constant term, which uses the value specified in `armalimit`). For the trading day regressor, at least one of the individual regressors must have a t-statistic exceeding 1.96 to be considered significant.

## Final Model Checks

After selecting a final model, several tests are performed to ensure model adequacy:

### 1. Checking for Unit Roots in AR Polynomials:

The model is inspected for unit roots in the AR polynomials to detect possible over-differencing. A unit root is identified if the modulus of any AR root is less than or equal to 1.05. If a unit root is found, the order of the corresponding AR polynomial is reduced, and the differencing order is increased. The model is then re-estimated, and diagnostics are updated.

### 2. Checking for Unit Roots in Nonseasonal MA Polynomials:

The sum of the nonseasonal MA coefficients is calculated. If this sum is within 0.001 of one, indicating a unit root, the order of regular differencing is decreased by one, and the order of the MA polynomial is reduced by one. A constant term is added to the model (if not already present) and tested for significance.

### 3. Optional Checking for Unit Roots in Seasonal MA Polynomials:

An optional test involves checking the sum of the seasonal MA coefficients. If this sum is within 0.001 of one, the order of seasonal differencing is decreased by one, and the MA order is reduced accordingly. Seasonal regressors are added to the model and evaluated for significance using an F-test.

### 4. Re-estimation and Outlier Detection:

After adjusting for unit roots, the model is re-estimated. If automatic outlier detection is specified, it is re-performed, and diagnostics are regenerated.

### 5. Assessment of Model Residuals:

If no constant term is present in the model, the mean of the residuals is tested. A significant t-statistic (absolute value greater than 2.5) leads to the inclusion of a constant term in the model.

### 6. Testing for Insignificant ARMA Parameters:

To simplify the model, the highest-order AR and MA coefficients (both seasonal and nonseasonal) are tested for significance. A coefficient is deemed significant if:

- Its t-statistic exceeds the value specified in `armalimit`.
- Its absolute value is greater than 0.15 (if the series has up to 150 observations) or 0.10 (if more than 150 observations).

If more than one insignificant coefficient is identified for a given parameter type and outlier detection is enabled, the outlier critical value is reduced (not below 2.8), and the model identification process is retried. If only one insignificant coefficient is found or the critical value cannot be reduced further, the order of the model is reduced by removing insignificant coefficients, and the model is re-estimated.

### 7. Constraints on Model Simplification:

The program ensures that the model retains at least one ARMA parameter; it will not remove the only ARMA parameter, even if it is insignificant. Additionally, no ARMA coefficients are eliminated if a unit root is detected (i.e., the modulus of any root is less than 1.053).



## 6 Conclusion

In this project, our goal was to thoroughly understand and document the functionalities and mathematical foundations of the **X13-ARIMA-SEATS** software, with a particular focus on its RegARIMA modeling and seasonal adjustment procedures. We aimed to assimilate the dispersed information from various references into a coherent and comprehensive guide that can serve as a valuable resource for future researchers and practitioners working with time series data. We made significant progress in demystifying the complex processes involved in time series analysis using **X13-ARIMA-SEATS**. By meticulously exploring the specifications, commonly used options, and the underlying mathematical theories—including ARIMA modeling, time series decomposition, and outlier detection techniques—we have created a detailed report that enhances understanding and accessibility of these sophisticated procedures.

One of the most time-consuming aspects of this project was gathering and synthesizing information from multiple sources. The intricate details of the procedures were often sparsely documented and lacked comprehensive explanations, which necessitated extensive research and careful interpretation to accurately convey the functionalities and theoretical concepts involved.

Despite the progress made, there is still work to be done. Specifically, we intend to delve deeper into the **SEATS** algorithm and the **X-13** seasonal adjustment method to gain a complete understanding of their processes and applications. Further exploration of these areas will not only augment our knowledge but also enhance the utility of this report as an authoritative guide. Overall, this project has been an enriching experience that has broadened our expertise in time series analysis and statistical modeling. It has highlighted the importance of thorough documentation and the value of creating resources that bridge the gap between complex methodologies and practical applications. We are motivated to continue this project, aspiring to fully comprehend the **SEATS** and **X-13** procedures and to share this knowledge with the broader community.

## References

- [1] William R. Bell and Steven C. Hillmer. Modeling time series with calendar variation. *Journal of the American Statistical Association*, 78(383):526–534, 1983.
- [2] G. E. P. Box and David A. Pierce. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American Statistical Association*, 65(332):1509–1526, 1970.
- [3] George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, and Greta M. Ljung. *Time Series Analysis: Forecasting and Control*. Wiley, Hoboken, NJ, 5 edition, 2016.
- [4] In-Chan Chang, George C. Tiao, and Ching S. Chen. Estimation of time series parameters in the presence of outliers. *Technometrics*, 30(2):193–204, 1988.
- [5] Ching S. Chen and L. M. Liu. Joint estimation of model parameters and outlier effects in time series. *Journal of the American Statistical Association*, 88(421):284–297, 1993.
- [6] David F. Findley, Brian C. Monsell, William R. Bell, Mark C. Otto, and Bor-Chung Chen. New capabilities and methods of the x-12-arima seasonal-adjustment program. *Journal of Business & Economic Statistics*, 16(2):127–177, 1998.
- [7] Victor Gómez and Agustín Maravall. *TRAMO and SEATS: User Instructions*, 1997. Time Series Analysis and Signal Extraction.
- [8] Victor Gómez and Agustín Maravall. Seasonal adjustment and signal extraction in economic time series. *Journal of Business & Economic Statistics*, 16(3):285–296, 1998.
- [9] Frank R. Hampel, Elvezio M. Ronchetti, Peter J. Rousseeuw, and Werner A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. Wiley, New York, 2005.
- [10] Agustín Maravall and Victor Gómez. Automatic modeling procedure for time series analysis. *Computational Statistics & Data Analysis*, 30:87–120, 2001.

- [11] Daniel Peña, George C. Tiao, and Ruey S. Tsay. *A Course in Time Series Analysis*. Wiley, New York, 2001.
- [12] Ruey S. Tsay. Time series model specification in the presence of outliers. *Journal of the American Statistical Association*, 81(393):132–141, 1986.
- [13] U.S. Census Bureau. *X-13ARIMA-SEATS Reference Manual*, 2024. Official Manual for X-13ARIMA-SEATS Software.
- [14] U.S. Census Bureau. X-13arima-seats seasonal adjustment program, 2024. Accessed: 2024-11-01.

## A Appendix A

Here I provide an example run of X13-ARIMA-SEATS. After that, I provide my attempt at replicating the process and results.

### A.1 X13-ARIMA-SEATS Output

Reading input spec file from ./X-13-ARIMA-SEATS/x13as/specfiles/tramo\_only.spc

[Previous Table](#) | [Index](#) | [Next Table](#)

U. S. Department of Commerce, U.  
S. Census Bureau

X-13ARIMA-SEATS monthly regARIMA model  
estimation Method,  
Release Version 1.1 Build 61

This software application provides an enhanced version of Statistics Canada's X-11-ARIMA extension (Dagum, 1980) of the X-11 variant of the Census Method II of Shiskin, Young and Musgrave (1967).

It also provides an ARIMA model-based method following Hillmer and Tiao (1982) and Burman (1980) that is very similar to the update of the method of SEATS (Gómez and Maravall, 1996) produced at the Bank of Spain by G. Caporello and A. Maravall for TSW (Caporello and Maravall, 2004). The present application includes additional enhancements.

X-13ARIMA-SEATS includes an automatic ARIMA model selection procedure based largely on the procedure of Gómez and Maravall (1998) as implemented in TRAMO (1996) and subsequent revisions.

Primary Programmers: Brian Monsell, Mark Otto and, for the ARIMA model-based signal extraction, Gianluca Caporello and Víctor Gómez

**Series Title-** Consumer Food Price Index - All India Combined  
**Series Name-** CPI  
Nov 1, 2024 05.14.12

- Period covered- 1st month,2013 to 8th month,2024
- Spectral plot of the original series generated

**FILE SAVE REQUESTS** (\* indicates file exists and will be overwritten)

./X-13-ARIMA-SEATS/x13as/specfiles/CPI.html\* program output file  
./X-13-ARIMA-SEATS/x13as/specfiles/CPI\_err.html\* program error file

LINKS TO OTHER HTML FILES

- [Error file for CPI](#)
- [Log entry for CPI](#)

[Previous Table](#) | [Index](#) | [Next Table](#)

Contents of spc file ./X-13-ARIMA-SEATS/x13as/specfiles/tramo\_only.spc

```
Line #
-----
1: series{
2:   title = "Consumer Food Price Index - All India Combined"
3:   start = 2013.01
```

Index for CPI.html

- [Program Header](#)
- [Content of input specification file](#)
- [AIC test for transformation](#)
- [A 1 Time series data \(for the span analyzed\)](#)
- [Prior-adjusted and transformed data](#)
- [Automatic ARIMA model selection](#)
- [Outlier detection](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 1](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 2](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 3](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 4](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 5](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 6](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Outlier identification: Backward deletion pass 1](#)
- [Regression model](#)
- [ARIMA model](#)
- [Final Additive Outlier t-values](#)
- [Final Level Change Outlier t-values](#)
- [Outlier detection](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 1](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 2](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 3](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 4](#)
- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Regression model](#)
- [ARIMA model](#)
- [Outlier identification: Forward addition pass 5](#)

```

4: span = (2013.01, 2024.08)
5: data = (
6:   105.4 106.4 106.5 107.5 109.1 112.4 115.2 117.3 119.0 121.1 12
7:   115.6 114.8 115.7 117.4 118.8 120.5 125.4 127.5 126.4 125.8 12
8:   122.7 122.7 122.8 123.4 124.5 127.1 128.1 130.3 131.3 132.4 13
9:   131.1 129.2 129.2 131.3 133.8 137.0 138.8 138.0 136.5 136.8 13
10:  131.9 131.8 131.8 132.1 132.4 134.1 138.3 140.1 138.2 139.4 14
11:  138.1 136.1 135.5 135.8 136.5 138.0 140.1 140.5 138.9 138.2 13
12:  135.0 135.1 135.9 137.3 139.0 141.1 143.4 144.7 146.0 149.1 15
13:  153.4 149.7 147.8 153.4 151.8 153.4 156.7 157.8 161.6 165.5 16
14:  156.4 155.5 155.0 156.4 159.4 161.3 162.9 162.7 162.7 166.9 16
15:  164.9 164.6 166.9 169.4 172.1 173.8 173.8 175.1 176.7 178.6 17
16:  174.8 174.4 174.9 175.9 177.2 181.7 193.8 192.5 188.4 190.4 19
17:  189.3 189.5 189.8 191.2 192.6 198.7 204.3 203.4
18: )
19: }
20:
21: transform{
22:   function = auto
23:   print = all
24: }
25:
26: automdl{print = all
27:   maxdiff = (2 1)}
28:
29: outlier{types = (ls ao)
30:   method = addone
31:   print = all
32: }
33:
34: estimate{
35:   print = all
36: }
37:
38: regression {
39:   variables = (tdlnolpyear)
40:   user = (diwali)
41:   aictest = (user tdlnolpyear)
42:   start = 2013.01
43:   data = (
44:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0405 -0.0405 0.0
45:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0
46:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.6595 0.6595 0.0
47:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0
48:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0
49:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.3595 0.3595 0.0
50:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0
51:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.6595 0.6595 0.0
52:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.0595 0.0595 0.0
53:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0
54:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -0.6595 0.6595 0.0
55:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0
56:     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.3405 -0.3405 0.0)
57:   print = all
58: }

```

- [Additive Outlier t-values](#)
- [Level Change Outlier t-values](#)
- [Final Additive Outlier t-values](#)
- [Final Level Change Outlier t-values](#)
- [Regression model](#)
- [Correlation matrix of regression parameter estimates](#)
- [ARIMA model](#)
- [Correlation matrix of ARMA parameter estimates](#)
- [Maximized log-likelihood and model selection criteria](#)
- [Roots of the AR and MA operators](#)
- [Matrix of regression variables](#)
- [Residuals from the estimated model](#)
- [QS Statistic for regARIMA Model Residuals](#)
- [Plot of Spectrum of the regARIMA model residuals](#)
- [Estimated regression effects \(X'beta\)](#)
- [Residuals from the estimated regression effects](#)
- [A 8 RegARIMA combined outlier component](#)
- [A 8.AO RegARIMA AO outlier component](#)
- [A 8.LS RegARIMA level change outlier component](#)
- [B 1 Original series \(prior adjusted\)](#)
- [QS diagnostic to detect seasonality](#)
- [Plot of Spectrum of the original series](#)
- [Peak probabilities for the Tukey spectrum](#)

- [Error Messages](#)
- [Log Entry](#)

[Previous Table](#) | [Index](#) | [Next Table](#)

### Likelihood statistics for model fit to untransformed series.

Likelihood Statistics	
Number of observations (nobs)	140
Effective number of observations (nefobs)	127
Number of parameters estimated (np)	5
Log likelihood (L)	-261.3694
<b>AIC</b>	532.7389
<b>AICC (F-corrected-AIC)</b>	533.2347
Hannan Quinn	538.5167
<b>BIC</b>	546.9598

### Likelihood statistics for model fit to log transformed series.

Likelihood Statistics

Number of observations (nobs)	140
Effective number of observations (nefobs)	127
Number of parameters estimated (np)	5
Log likelihood	381.5430
Transformation Adjustment	-635.4861
Adjusted Log likelihood (L)	-253.9431
AIC	517.8862
AICC (F-corrected-AIC)	518.3821
Hannan Quinn	523.6640
BIC	532.1071

nobs = number of observations  
nefobs = nobs - total order of differencing operators  
np = number of estimated regression and ARIMA model parameters including the variance  
V = Covariance matrix of z's  
z = Vector of data - regression values  
c = Vector of prior adjustment factors  
Log likelihood =  $-[\text{nefobs} \cdot \log(2 \cdot \pi) + \log|V| + z'(V^{-1})z]/2$   
Transformation Adjustment =  $\sum (\ln(|c_i \cdot (c_i \cdot y_i)^{(\lambda - 1)}|))$  where the sum is over the last nefobs observations  
 $AIC = -2 \cdot L + 2 \cdot np$   
 $AICC = -2 \cdot L + 2 \cdot np \cdot [\text{nefobs} / (\text{nefobs} - np - 1)]$   
Hannan Quinn =  $-2 \cdot L + 2 \cdot np \cdot \log[\log(\text{nefobs})]$   
 $BIC = -2 \cdot L + np \cdot \log(\text{nefobs})$

\*\*\*\*\* AICC (with aicdiff=-2.00) prefers log transformation \*\*\*\*\*

[Previous Table](#) | [Index](#) | [Next Table](#)

A 1 Time series data (for the span analyzed)

From 2013.Jan to 2024.Aug  
Observations 140

A 1 Time series data (for the span analyzed)													
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	TOTAL
2013	105.	106.	107.	108.	109.	112.	115.	117.	119.	121.	124.	119.	1363.
2014	116.	115.	116.	117.	119.	121.	125.	128.	126.	126.	125.	123.	1457.
2015	123.	123.	123.	123.	125.	127.	128.	130.	131.	132.	133.	131.	1530.
2016	131.	129.	129.	131.	134.	137.	139.	138.	137.	137.	136.	133.	1610.
2017	132.	132.	132.	132.	132.	134.	138.	140.	138.	139.	142.	140.	1631.
2018	138.	136.	136.	136.	137.	138.	140.	141.	139.	138.	138.	136.	1652.
2019	135.	135.	136.	137.	139.	141.	143.	145.	146.	149.	152.	155.	1713.
2020	153.	150.	148.	153.	152.	153.	157.	158.	162.	166.	166.	161.	1878.
2021	156.	156.	155.	156.	159.	161.	163.	163.	163.	167.	169.	167.	1935.
2022	165.	165.	167.	169.	172.	174.	174.	175.	177.	179.	177.	174.	2067.
2023	175.	174.	175.	176.	177.	182.	194.	193.	188.	190.	192.	191.	2207.
2024	189.	190.	190.	191.	193.	199.	204.	203.					1559.
AVGE	143.	142.	143.	144.	146.	148.	152.	152.	148.	149.	150.	148.	

Table Total- 20601.    Mean- 147.    Standard Deviation- 24.  
Minimum- 105.    Maximum- 204.

[Previous Table](#) | [Index](#) | [Next Table](#)

Transformed (prior-adjusted) data for regARIMA modeling

Data

Data												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	4.658	4.667	4.668	4.677	4.692	4.722	4.747	4.765	4.779	4.797	4.819	4.777
2014	4.750	4.743	4.751	4.766	4.777	4.792	4.832	4.848	4.839	4.835	4.831	4.815
2015	4.810	4.810	4.811	4.815	4.824	4.845	4.853	4.870	4.877	4.886	4.890	4.877
2016	4.876	4.861	4.861	4.877	4.896	4.920	4.933	4.927	4.916	4.919	4.910	4.891
2017	4.882	4.881	4.881	4.884	4.886	4.899	4.929	4.942	4.929	4.937	4.952	4.939
2018	4.928	4.913	4.909	4.911	4.916	4.927	4.942	4.945	4.934	4.929	4.926	4.913
2019	4.905	4.906	4.912	4.922	4.934	4.949	4.966	4.975	4.984	5.005	5.021	5.045
2020	5.033	5.009	4.996	5.033	5.023	5.033	5.054	5.061	5.085	5.109	5.112	5.079
2021	5.052	5.047	5.043	5.052	5.071	5.083	5.093	5.092	5.092	5.117	5.130	5.119
2022	5.105	5.104	5.117	5.132	5.148	5.158	5.158	5.165	5.174	5.185	5.176	5.160
2023	5.164	5.161	5.164	5.170	5.177	5.202	5.267	5.260	5.239	5.249	5.260	5.251
2024	5.243	5.244	5.246	5.253	5.261	5.292	5.320	5.315				

[Previous Table](#) | [Index](#) | [Next Table](#)

Automatic ARIMA Model Selection

Procedure based closely on TRAMO ,method of Gomez and Maravall (2000)  
"Automatic Modeling Methods for Univariate Series",  
A Course in Time Series (Edited by D. Pena, G. C. Tiao, R. S. Tsay),  
New York : J. Wiley and Sons

Maximum order for regular ARMA parameters : 2

Maximum order for seasonal ARMA parameters : 1

Maximum order for regular differencing : 2

Maximum order for seasonal differencing : 1

[Previous Table](#) | [Index](#) | [Next Table](#)

OUTLIER DETECTION

From 2013.Jan to 2024.Aug  
Observations 140

Types : AO and LS  
Method : add one  
Critical |t| for AO outliers : 3.88  
Critical |t| for LS outliers : 3.88

Estimation converged in 30 ARMA iterations, 91 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (0 1 1)(0 1 1)

Nonseasonal differences: 1  
Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal MA		
Lag 1	-0.33035	0.07440
Seasonal MA		
Lag 12	0.99871	0.06210

Model Innovation Variance	
Variance	0.11474E-03
Standard Error of Variance	0.14399E-04

[Previous Table](#) | [Index](#) | [Next Table](#)

Forward addition pass 1

Robust root mse : 6.75E-03  
Normal root mse : 1.07E-02

Outliers flagged as significant this pass		
	t(AO)	t(LS)
AO2013.Nov	5.33*	3.96
LS2013.Dec	-2.51	-4.73*
AO2019.Nov	-3.97*	-0.65
LS2019.Dec	4.60	5.82*
AO2020.Mar	-5.46*	-3.20
AO2020.Apr	6.21*	5.70
LS2020.May	-2.92	-4.42*
AO2023.Jun	-4.92*	-1.21
LS2023.Jul	6.10	6.81*

Add +LS2023.Jul ( 6.81 )

[Previous Table](#) | [Index](#) | [Next Table](#)

AO Outlier t-values

AO Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.2065	1.7104	-0.2908	-0.2340	-0.8652	1.3768	-1.0952	-0.4052	1.6644	-2.9485	5.3280	-2.5121
2014	-0.1569	-0.8525	0.6733	-0.5255	1.3642	-2.3799	0.9224	1.1162	0.4060	-0.4655	-0.5001	-0.2544
2015	-0.0511	0.2968	0.4912	-0.3258	-0.9503	2.6307	-3.1487	0.9702	0.6560	-0.2844	0.0990	-1.1985
2016	2.2445	-1.3605	0.0257	-0.3783	0.0339	1.2832	-0.2183	0.0807	-0.4485	0.9296	-1.1208	0.0259
2017	-0.3606	0.3460	0.6499	-0.4233	0.3050	-1.1324	-0.1536	2.0337	-1.2409	-0.8646	0.9804	-0.3436
2018	1.0289	-0.8554	0.7440	-0.7199	0.4104	0.0855	-0.5759	0.6952	0.3382	-0.4955	-0.7070	0.1849
2019	-0.2778	-0.0414	0.6730	-0.6753	0.5687	0.3499	-0.8085	-0.1764	-0.3720	1.2186	-3.9715	4.6008
2020	-0.3478	1.2819	-5.4565	6.2071	-2.9203	0.2230	0.4963	-2.4974	1.5238	0.4502	1.1924	-0.1006
2021	-1.4357	0.7000	0.0479	-1.4519	1.6443	0.1842	-0.4723	-0.0659	-1.4374	1.0298	-0.0908	0.6063
2022	-0.3686	-1.0558	1.3109	-0.8934	0.9647	1.5910	-2.3869	-0.0785	0.4135	1.3342	-0.9746	-1.4897
2023	1.5880	-0.7087	1.0547	-0.8811	0.9850	-4.9175	6.0956	-0.7091	-1.5030	0.0961	-0.2352	0.4810
2024	-0.5826	0.6019	0.0756	0.3013	-1.5411	0.7082	1.3962	-1.6608				

[Previous Table](#) | [Index](#) | [Next Table](#)

LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-0.607795	-0.131860	0.249780	1.661063	-0.584731	1.201702	1.868028	-0.846903	3.962553	-4.728263
2014	-0.630548	-0.372954	1.017382	-0.080961	0.776223	-1.449052	2.432940	0.929081	-0.890771	-1.553002	-0.793729	0.022043
2015	0.437071	0.518670	0.034428	-0.766772	-0.235359	1.314671	-2.976503	2.159194	0.582423	-0.487561	-0.023588	-0.185130
2016	1.769758	-1.889512	0.329883	0.287973	0.905011	0.849748	-1.243427	-0.887855	-1.022954	-0.291446	-1.807845	0.020290
2017	-0.021925	0.565006	0.000475	-1.059531	-0.369057	-0.866504	0.980697	1.231707	-2.085716	-0.061568	1.348782	-0.250455
2018	0.309998	-1.365554	0.029951	-1.183609	-0.009395	-0.678850	-0.818330	0.120866	-1.014284	-1.565889	-0.757690	0.395509
2019	0.093940	0.545711	0.613056	-0.484650	0.616881	-0.310753	-0.881573	0.437007	0.726707	1.333554	-0.654217	5.823981
2020	-1.680732	-1.108457	-3.198547	5.701808	-4.423000	0.340431	-0.023301	-0.833050	3.243223	0.757577	0.023223	-1.921857
2021	-1.757714	0.585209	-0.556532	-0.634627	1.733670	-0.948501	-1.248977	-0.478919	-0.373055	1.971640	0.291872	0.439915
2022	-0.549090	0.052768	1.774484	-0.363844	1.093514	-0.480059	-3.075274	0.817473	0.948651	0.274132	-1.902115	-0.312448
2023	2.117418	-0.474723	0.681222	-1.039094	0.398131	-1.208513	6.812796	-3.128903	-1.982252	0.469465	0.312758	0.696407
2024	-0.088175	0.860247	-0.121605	-0.244923	-0.736431	1.777482	0.629418					

Average absolute percentage error in within-sample forecasts:

- Last year: 0.92
- Last-1 year: 0.74
- Last-2 year: 2.84
- Last three years: 1.50

Estimation converged in 37 ARMA iterations, 130 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
LS2023.Jul	0.0450	0.00962	4.68

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (0 1 1)(0 1 1)

- Nonseasonal differences: 1
- Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal MA		
Lag 1	-0.36483	0.07355
Seasonal MA		
Lag 12	0.99863	0.06151

Model Innovation Variance	
Variance	0.97957E-04
Standard Error of Variance	0.12293E-04



[Previous Table](#) | [Index](#) | [Next Table](#)

## Forward addition pass 2

Robust root mse : 6.34E-03

Normal root mse : 9.90E-03

Outliers flagged as significant this pass

	t(AO)	t(LS)
AO2013.Nov	5.76*	4.40
LS2013.Dec	-2.86	-5.13*
AO2019.Nov	-4.35*	-0.96
LS2019.Dec	5.01	6.24*
AO2020.Mar	-5.98*	-3.63
AO2020.Apr	6.70*	6.25
LS2020.May	-3.13	-4.82*

Add +AO2020.Apr ( 6.70 )

[Previous Table](#) | [Index](#) | [Next Table](#)

## AO Outlier t-values

AO Outlier t-values

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.3388	1.8202	-0.3540	-0.2292	-0.7583	0.9318	-0.5859	-0.6995	1.9938	-3.3760	5.7631	-2.8580
2014	0.0609	-0.9948	0.8272	-0.7459	1.7464	-3.1129	1.5905	0.8553	0.5030	-0.5076	-0.4817	-0.2482
2015	-0.0558	0.2893	0.5058	-0.3395	-0.9094	2.3383	-2.8532	0.9173	0.6960	-0.3486	0.1990	-1.3493
2016	2.4252	-1.5218	0.1451	-0.4801	0.2098	0.7871	0.2690	-0.0700	-0.4424	0.9982	-1.2010	0.0968
2017	-0.3976	0.3400	0.7037	-0.5499	0.5666	-1.7134	0.3589	1.9157	-1.2251	-0.9045	1.0649	-0.4472
2018	1.1380	-0.9806	0.8762	-0.8748	0.6559	-0.4650	-0.0569	0.5134	0.4107	-0.5163	-0.7147	0.2167
2019	-0.2890	-0.0709	0.7493	-0.8084	0.7988	-0.1854	-0.3028	-0.3071	-0.4289	1.4641	-4.3548	5.0077
2020	-0.7420	1.7111	-5.9811	6.6990	-3.1268	-0.1590	1.0936	-2.8454	1.7486	0.3139	1.2450	-0.0947
2021	-1.4853	0.7347	0.1066	-1.6207	1.9402	-0.4121	0.0622	-0.2037	-1.4502	1.1016	-0.1602	0.6455
2022	-0.3543	-1.1393	1.4495	-1.0612	1.1671	1.1359	-1.9668	-0.1888	0.4282	1.3598	-0.9736	-1.5533
2023	1.6898	-0.7645	0.9326	-0.3003	-0.9094	1.1589	0.8803	1.2854	-2.2470	0.4159	-0.3861	0.5839
2024	-0.6681	0.6604	0.0376	0.3089	-1.4240	0.2008	1.9856	-1.9702				

[Previous Table](#) | [Index](#) | [Next Table](#)

## LS Outlier t-values

LS Outlier t-values

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-0.69610	-0.10876	0.26982	1.52282	-0.01310	0.95283	2.11273	-1.18136	4.39642	-5.12514
2014	-0.40332	-0.50224	1.14101	-0.22565	1.00669	-1.87868	3.26432	0.63197	-0.78061	-1.61156	-0.77288	0.02293
2015	0.43296	0.52344	0.04538	-0.79030	-0.22944	1.27318	-2.59061	2.12155	0.61117	-0.53877	0.03720	-0.29161
2016	1.93767	-2.06705	0.44737	0.20769	1.00090	0.65471	-0.64514	-1.08656	-0.97412	-0.24316	-1.89230	0.09193
2017	-0.06804	0.58768	0.02588	-1.13664	-0.22816	-1.16458	1.66573	1.06884	-2.09609	-0.07199	1.42237	-0.33699
2018	0.40184	-1.47544	0.14490	-1.30272	0.14257	-0.94124	-0.17536	-0.08108	-0.93041	-1.60885	-0.75582	0.42505
2019	0.06709	0.54318	0.66000	-0.57800	0.75754	-0.56206	-0.25740	0.24258	0.75115	1.45974	-0.95915	6.23573
2020	-2.03782	-0.80742	-3.63315	6.24840	-4.81937	0.34521	0.60953	-1.19530	3.50670	0.61761	0.09903	-1.95795
2021	-1.80149	0.65340	-0.56033	-0.73646	1.94115	-1.26394	-0.58682	-0.68752	-0.35277	2.04320	0.22324	0.48783
2022	-0.57870	0.00736	1.88903	-0.50586	1.24746	-0.68046	-2.56226	0.68839	1.00282	0.29529	-1.95130	-0.34282
2023	2.22341	-0.57000	0.69297	-0.84774	-0.35681	1.15894	0.00000	-0.88029	-2.87569	0.84083	0.15323	0.79105
2024	-0.17360	0.92826	-0.16299	-0.22510	-0.73556	1.61729	1.30251					

LS Outlier t-values have been set to zero for the following observations:

2023.Jul

Average absolute percentage error in within-sample forecasts:

- Last year: 0.94
- Last-1 year: 0.69
- Last-2 year: 3.14
- Last three years: 1.59

Estimation converged in 70 ARMA iterations, 239 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
AO2020.Apr	0.0247	0.00487	5.08
LS2023.Jul	0.0433	0.00837	5.17

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (0 1 1)(0 1 1)

- Nonseasonal differences: 1
- Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal MA		
Lag 1	-0.47280	0.07060
Seasonal MA		
Lag 12	0.95353	0.05510

Model Innovation Variance	
Variance	0.86086E-04
Standard Error of Variance	0.10803E-04

[Previous Table](#) | [Index](#) | [Next Table](#)

Forward addition pass 3

- Robust root mse : 7.02E-03
- Normal root mse : 9.28E-03

Outliers flagged as significant this pass		
	t(AO)	t(LS)
AO2013.Nov	5.60*	4.58
LS2013.Dec	-3.20	-5.04*

<b>AO2019.Nov</b>	-4.29*	-1.62
<b>LS2019.Dec</b>	4.64	5.74*

Add +LS2019.Dec ( 5.74 )

[Previous Table](#) | [Index](#) | [Next Table](#)

## AO Outlier t-values

AO Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.23889	1.90635	-0.96305	0.63260	-1.19001	1.03392	-0.43541	-0.94808	2.33892	-3.74414	5.60278	-3.19632
2014	0.74175	-1.12521	0.72826	-0.56008	1.57002	-2.81271	1.62220	0.35616	0.50190	-0.42747	-0.38097	-0.08777
2015	-0.18639	0.46380	-0.14088	0.55231	-1.55493	2.59453	-2.84379	1.10581	0.38613	-0.33862	0.38998	-1.39370
2016	2.24548	-1.40895	-0.00476	0.07119	-0.23754	0.82853	0.03136	0.08904	-0.55589	1.05709	-1.21600	0.36694
2017	-0.53988	0.48068	0.18219	-0.00526	0.22893	-1.25659	0.19663	1.58792	-1.02966	-0.73343	1.00693	-0.59971
2018	1.09994	-0.90834	0.58271	-0.37439	0.29534	-0.28238	-0.08201	0.39161	0.31647	-0.37014	-0.59676	0.28668
2019	-0.34632	0.08770	0.27989	-0.20693	0.31136	0.01971	-0.38596	0.02400	-0.77769	1.89355	-4.28722	4.63762
2020	-0.88820	0.51736	-1.90937	0.00000	2.07409	-2.08734	2.05821	-3.07742	2.01676	-0.23438	1.15380	-0.07657
2021	-1.29811	0.82060	-0.23202	-0.92551	1.36050	-0.34334	0.01477	-0.00188	-1.31206	1.08541	-0.34398	0.63728
2022	-0.32651	-0.85973	0.96683	-0.48457	0.52677	1.18859	-1.81529	0.10654	0.16972	1.12702	-0.71893	-1.33954
2023	1.52154	-0.71890	0.50589	0.26904	-1.16445	1.21699	0.46661	1.34263	-2.08990	0.68206	-0.60139	0.76952
2024	-0.88285	0.94239	-0.62756	1.04368	-1.66703	0.37212	1.56161	-1.69395				

AO Outlier t-values have been set to zero for the following observations:

2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

## LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-1.04346	0.60741	-0.47665	1.56306	-0.20866	0.53774	2.16697	-1.84716	4.57870	-5.03717
2014	0.44849	-0.82308	1.10761	-0.14075	0.81950	-1.87291	2.95294	0.16677	-0.44444	-1.30565	-0.57197	0.08187
2015	0.23251	0.55086	-0.24501	-0.00359	-0.95052	1.71621	-2.73533	2.14301	0.24904	-0.41362	0.16753	-0.50177
2016	1.89019	-1.96130	0.45650	0.46567	0.34367	0.75027	-0.67072	-0.72277	-0.87735	0.07675	-1.73744	0.34953
2017	-0.28022	0.64525	-0.17959	-0.49290	-0.48394	-0.87553	1.28032	0.94025	-1.78431	-0.01703	1.24169	-0.48645
2018	0.54280	-1.34273	0.21600	-0.78431	-0.14257	-0.64853	-0.16534	-0.02444	-0.69697	-1.23990	-0.60461	0.41958
2019	-0.07243	0.52095	0.37040	-0.10946	0.24528	-0.28878	-0.32347	0.33848	0.29810	1.63261	-1.61715	5.74151
2020	-2.20826	-0.70497	-1.66675	1.35098	1.35098	-1.87587	1.67262	-1.84784	3.42718	-0.03517	0.36705	-1.61316
2021	-1.48178	0.74608	-0.66201	-0.26508	1.32154	-1.01241	-0.42578	-0.44987	-0.44784	1.80378	-0.05909	0.53126
2022	-0.56246	-0.00156	1.47358	-0.18350	0.64721	-0.25670	-2.29931	0.81605	0.63533	0.34395	-1.59028	-0.35641
2023	1.94264	-0.66902	0.56433	-0.30210	-0.77258	1.21699	0.00000	-0.46661	-2.58678	1.00452	-0.16710	0.86457
2024	-0.45612	1.05747	-0.55963	0.51689	-1.27240	1.58560	0.97896					

LS Outlier t-values have been set to zero for the following observations:

2023.Jul

Average absolute percentage error in within-sample forecasts:

Last year: 1.08

Last-1 year: 0.82

Last-2 year: 3.32

Last three years: 1.74

Estimation converged in 97 ARMA iterations, 326 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
LS2019.Dec	0.0361	0.00759	4.75
AO2020.Apr	0.0235	0.00435	5.39
LS2023.Jul	0.0427	0.00755	5.65

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (0 1 1)(0 1 1)

Nonseasonal differences: 1  
Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal MA		
Lag 1	-0.50574	0.06811
Seasonal MA		
Lag 12	0.98166	0.05732

Model Innovation Variance	
Variance	0.71352E-04
Standard Error of Variance	0.89540E-05

[Previous Table](#) | [Index](#) | [Next Table](#)

Forward addition pass 4

Robust root mse : 7.55E-03  
Normal root mse : 8.45E-03

Outliers flagged as significant this pass		
	t(AO)	t(LS)
AO2013.Nov	4.83*	4.12
LS2013.Dec	-2.70	-4.26*

Add +AO2013.Nov ( 4.83 )

[Previous Table](#) | [Index](#) | [Next Table](#)

AO Outlier t-values

AO Outlier t-values											
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov
Dec											

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.06091	1.76710	-0.93617	0.63574	-1.10173	0.93813	-0.36262	-0.91920	2.21236	-3.42289	4.82968	-2.70231
2014	0.71086	-1.07514	0.77143	-0.66154	1.54623	-2.59819	1.51694	0.25448	0.39558	-0.19479	-0.75032	0.38254
2015	-0.42119	0.56455	-0.25434	0.62064	-1.53479	2.47443	-2.69121	1.15666	0.17273	-0.10072	-0.02132	-0.88585
2016	1.85202	-1.22545	0.02808	0.04958	-0.22718	0.76339	-0.06127	0.20109	-0.66762	1.22971	-1.60148	0.86294
2017	-0.75795	0.56115	0.09863	-0.00767	0.22206	-1.08403	0.12565	1.45640	-1.01572	-0.42720	0.47696	-0.15100
2018	0.81861	-0.77305	0.54566	-0.38232	0.29972	-0.25491	-0.09421	0.39295	0.15395	-0.07371	-1.00313	0.74794
2019	-0.54971	0.18385	0.22272	-0.22333	0.34888	-0.10940	-0.09208	-0.46557	0.32570	-0.45009	1.12247	-0.53472
2020	1.40802	-0.52216	-1.26612	0.00000	1.87259	-1.96813	1.98115	-2.83635	1.81997	-0.10973	0.59484	0.39215
2021	-1.37296	0.84298	-0.23372	-0.83407	1.22517	-0.32007	-0.01274	0.09530	-1.32620	1.27055	-0.85601	1.07604
2022	-0.52653	-0.65470	0.83396	-0.44102	0.43031	1.10223	-1.66761	0.22304	-0.04512	1.25791	-1.09939	-0.72456
2023	1.15257	-0.58477	0.43581	0.23571	-1.03561	1.10181	0.32652	1.32506	-2.06599	0.98131	-1.13104	1.26763
2024	-1.12164	1.04252	-0.70218	1.02002	-1.52441	0.37009	1.36890	-1.53748				

AO Outlier t-values have been set to zero for the following observations:

2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-1.0107	0.6111	-0.4902	1.4187	-0.2066	0.4219	2.0187	-1.8202	4.1197	-4.2565
2014	0.4371	-0.7949	1.0703	-0.2670	0.8798	-1.8013	2.7057	0.0717	-0.3699	-1.0563	-0.7192	0.5818
2015	-0.0825	0.6467	-0.3327	0.1082	-0.9677	1.6936	-2.5987	2.0688	0.0655	-0.2342	-0.0596	-0.0228
2016	1.5107	-1.7005	0.4257	0.3779	0.2920	0.6853	-0.6385	-0.5308	-0.8812	0.2773	-1.8570	0.9186
2017	-0.5786	0.7355	-0.2381	-0.4099	-0.3967	-0.7809	1.0994	0.8791	-1.6474	0.1151	0.8567	0.0312
2018	0.2925	-1.1255	0.2159	-0.7312	-0.0685	-0.5879	-0.1468	0.0167	-0.6655	-0.9326	-0.8055	0.9344
2019	-0.3637	0.5891	0.2700	-0.1162	0.2710	-0.3341	-0.1451	0.0146	0.8247	0.2757	1.1225	0.0000
2020	0.5347	-1.7952	-0.9819	1.1304	1.1304	-1.7545	1.6200	-1.8031	3.1103	-0.0489	0.1415	-0.8915
2021	-1.5669	0.8154	-0.6471	-0.2426	1.2033	-0.9218	-0.3686	-0.3456	-0.5118	1.7893	-0.4145	1.0713
2022	-0.7954	0.1190	1.2545	-0.1909	0.5736	-0.1730	-2.0875	0.8059	0.4206	0.4988	-1.6842	0.2196
2023	1.4731	-0.5271	0.4871	-0.2671	-0.6861	1.1018	0.0000	-0.3265	-2.4317	1.1544	-0.5481	1.4139
2024	-0.7857	1.1587	-0.6501	0.5677	-1.2005	1.4413	0.8337					

LS Outlier t-values have been set to zero for the following observations:

2019.Dec 2023.Jul

Average absolute percentage error in within-sample forecasts:

- Last year: 1.08
- Last-1 year: 0.70
- Last-2 year: 3.36
- Last three years: 1.71

Estimation converged in 31 ARMA iterations, 141 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
AO2013.Nov	0.0183	0.00348	5.25

LS2019.Dec	0.0316	0.00635	4.98
AO2020.Apr	0.0218	0.00349	6.25
LS2023.Jul	0.0410	0.00624	6.56

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (0 1 1)(0 1 1)

Nonseasonal differences: 1  
Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal MA		
Lag 1	-0.62768	0.06023
Seasonal MA		
Lag 12	0.99977	0.06449

Model Innovation Variance	
Variance	0.58785E-04
Standard Error of Variance	0.73770E-05

[Previous Table](#) | [Index](#) | [Next Table](#)

Forward addition pass 5

Robust root mse : 6.34E-03  
Normal root mse : 7.67E-03

Outliers flagged as significant this pass		
	t(AO)	t(LS)
AO2020.Aug	-3.89*	-2.88
LS2020.Sep	2.91	4.10*

Add +LS2020.Sep ( 4.10 )

[Previous Table](#) | [Index](#) | [Next Table](#)

AO Outlier t-values

AO Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.5196	2.2649	-1.5020	1.2603	-1.7413	1.6659	-1.2344	0.3051	0.2402	0.3650	0.0000	1.4961
2014	-1.1907	-0.2857	0.6564	-1.0033	2.1002	-3.1656	2.1452	-0.4420	0.9390	-0.7750	-0.1071	0.0561
2015	-0.4176	0.7647	-0.6799	1.2194	-2.2641	3.2334	-3.3161	1.5897	0.0771	-0.5272	0.8228	-1.7280
2016	2.4859	-1.7596	0.4645	-0.1716	-0.1783	0.7888	-0.2297	0.3755	-0.7969	1.2159	-1.4458	0.8545
2017	-0.8480	0.6290	0.0571	-0.0801	0.3387	-1.1032	0.1883	1.2344	-0.6781	-0.9614	1.3011	-0.9903
2018	1.4438	-1.3221	1.0358	-0.7985	0.6196	-0.4731	0.0928	0.1712	0.3794	-0.4292	-0.4736	0.4011
2019	-0.3798	0.0625	0.3475	-0.3948	0.5121	-0.2849	0.1237	-0.6828	0.6281	-0.8480	1.5540	-0.6624
2020	1.5164	-0.5580	-1.4261	0.0000	2.5688	-2.8639	3.0460	-3.8895	2.9101	-1.2943	1.5643	-0.2847

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2021	-0.9779	0.6325	-0.0253	-1.0325	1.3956	-0.4895	0.0536	0.1434	-1.2925	1.1376	-0.6124	0.8297
2022	-0.3755	-0.7488	0.9469	-0.5325	0.3443	1.2136	-1.7825	0.4038	-0.0801	0.9098	-0.4866	-1.2272
2023	1.5470	-0.9468	0.6718	0.1005	-1.0125	1.1809	0.0378	1.7005	-2.3839	1.2396	-1.2494	1.4987
2024	-1.5503	1.5332	-1.2267	1.4772	-1.8464	0.6249	1.2432	-1.5911				

AO Outlier t-values have been set to zero for the following observations:

2013.Nov 2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

## LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-1.52947	1.16170	-1.10189	2.02916	-0.96635	1.25376	0.75131	0.39964	-0.12327	-0.12327
2014	-1.96701	0.24248	0.73323	-0.44889	1.35503	-2.42839	3.27835	-0.59130	0.20514	-1.48897	-0.09488	0.09798
2015	-0.00332	0.74762	-0.63127	0.59450	-1.60327	2.47790	-3.35141	2.62618	-0.23671	-0.37604	0.57454	-0.90767
2016	2.20197	-2.27788	0.89536	0.05867	0.36795	0.68888	-0.73299	-0.31806	-0.99616	0.44055	-1.75366	0.84840
2017	-0.69179	0.83608	-0.29831	-0.40186	-0.25760	-0.86761	1.12119	0.78010	-1.44538	-0.22356	1.51179	-0.83013
2018	0.95382	-1.64647	0.73787	-1.12995	0.30900	-0.80765	0.04422	-0.12298	-0.43207	-1.11703	-0.34608	0.50703
2019	-0.21666	0.46731	0.35418	-0.27241	0.43923	-0.48392	0.02820	-0.19500	1.04028	-0.05591	1.55403	0.00000
2020	0.66244	-1.89472	-1.02551	1.48860	1.48860	-2.48706	2.55663	-2.88469	4.09829	-1.15108	1.18119	-1.63631
2021	-1.11442	0.64931	-0.49137	-0.44639	1.41478	-1.10123	-0.22043	-0.31651	-0.57589	1.75534	-0.29223	0.81139
2022	-0.68401	-0.00571	1.34366	-0.36273	0.59708	-0.02398	-2.21397	1.00026	0.27360	0.41840	-1.22327	-0.34988
2023	1.85736	-0.93031	0.77414	-0.42942	-0.63680	1.18089	0.00000	-0.03775	-2.78571	1.50488	-0.72777	1.51802
2024	-1.18314	1.60964	-1.15482	1.05589	-1.60537	1.71304	0.66167					

LS Outlier t-values have been set to zero for the following observations:

2019.Dec 2023.Jul

Average absolute percentage error in within-sample forecasts:

Last year: 1.21  
 Last-1 year: 0.77  
 Last-2 year: 3.61  
 Last three years: 1.86

Estimation converged in 34 ARMA iterations, 149 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

## Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
AO2013.Nov	0.0186	0.00303	6.13
LS2019.Dec	0.0299	0.00568	5.27
AO2020.Apr	0.0231	0.00313	7.38
LS2020.Sep	0.0212	0.00567	3.73
LS2023.Jul	0.0407	0.00553	7.35

[Previous Table](#) | [Index](#) | [Next Table](#)

## ARIMA Model

ARIMA Model: (0 1 1)(0 1 1)

Nonseasonal differences: 1

Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
<b>Nonseasonal MA</b>		
Lag 1	-0.69023	0.05578
<b>Seasonal MA</b>		
Lag 12	0.99930	0.06242

Model Innovation Variance	
Variance	0.53182E-04
Standard Error of Variance	0.66739E-05

[Previous Table](#) | [Index](#) | [Next Table](#)

## Forward addition pass 6

Robust root mse : 6.92E-03

Normal root mse : 7.29E-03

[Previous Table](#) | [Index](#) | [Next Table](#)

## AO Outlier t-values

AO Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.3681	2.1785	-1.5708	1.3752	-1.7182	1.5760	-1.1000	0.1616	0.3672	0.2098	0.0000	1.2716
2014	-0.9192	-0.3990	0.8141	-1.2170	2.2121	-3.1460	2.3973	-1.1030	1.4002	-1.0591	0.1873	-0.0519
2015	-0.3862	0.7859	-0.8370	1.3566	-2.2259	2.9586	-2.8818	1.2618	0.2473	-0.6783	0.9944	-1.7557
2016	2.3510	-1.7334	0.6284	-0.3067	-0.0258	0.5213	-0.0625	0.1097	-0.4862	0.9585	-1.2163	0.7921
2017	-0.7859	0.5747	0.0180	-0.1077	0.3978	-1.0822	0.4525	0.5350	-0.0399	-1.2250	1.5010	-1.2510
2018	1.5842	-1.4668	1.2065	-0.9873	0.8345	-0.7283	0.4671	-0.3458	0.7578	-0.6594	-0.1475	0.1699
2019	-0.1934	-0.0557	0.3941	-0.4626	0.6022	-0.4873	0.4467	-1.0193	0.9875	-1.0426	1.5525	-0.6315
2020	1.3696	-0.5694	-1.0729	0.0000	1.6620	-1.4602	0.8854	-0.4636	-1.6211	1.1291	-0.0283	0.6590
2021	-1.3912	0.8983	-0.2377	-0.7413	1.1376	-0.5007	0.2112	-0.1261	-0.8126	0.8135	-0.4565	0.6449
2022	-0.2775	-0.6348	0.8031	-0.4667	0.2973	0.9232	-1.3136	0.0975	0.2036	0.5014	-0.1673	-1.1918
2023	1.4436	-0.9419	0.6604	0.0324	-0.8416	1.0228	0.0184	1.4051	-2.0122	1.2219	-1.3114	1.5777
2024	-1.6674	1.6695	-1.4290	1.6034	-1.8347	0.7577	0.8922	-1.2884				

AO Outlier t-values have been set to zero for the following observations:

2013.Nov 2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

## LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec



	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-1.57036	1.28481	-1.22558	1.91598	-0.96507	1.04472	0.80775	0.22828	-0.07566	-0.07566
2014	-1.65522	0.10490	0.80321	-0.68654	1.53916	-2.51789	3.25834	-1.14432	0.87984	-1.69062	0.25263	-0.09079
2015	0.00475	0.71219	-0.73124	0.80604	-1.68505	2.40267	-3.03048	2.25949	-0.05052	-0.50410	0.74128	-1.08347
2016	2.13614	-2.17988	1.00450	-0.14922	0.41395	0.46122	-0.49597	-0.38132	-0.58492	0.30783	-1.45276	0.77796
2017	-0.67639	0.76614	-0.28973	-0.32323	-0.12549	-0.85581	1.13163	0.30103	-0.68161	-0.60724	1.64253	-1.11060
2018	1.18542	-1.72179	0.97277	-1.24344	0.56924	-0.96322	0.37320	-0.48430	0.14945	-1.24072	-0.03155	0.23928
2019	-0.07314	0.28149	0.38353	-0.34050	0.50905	-0.59676	0.29577	-0.52388	1.35047	-0.40602	1.55248	0.00000
2020	0.63149	-1.69559	-0.79053	1.10617	1.10617	-1.54232	1.04942	-0.46357	0.00000	1.62106	-0.48745	-0.40573
2021	-1.57362	0.97976	-0.66972	-0.23244	1.12839	-0.96123	-0.04297	-0.43086	-0.20095	1.29037	-0.20157	0.63648
2022	-0.54766	-0.03682	1.12842	-0.34613	0.51089	-0.03529	-1.73262	0.67934	0.50320	0.12898	-0.79255	-0.48776
2023	1.69768	-0.94950	0.77534	-0.42530	-0.51745	1.02277	0.00000	-0.01842	-2.28944	1.38835	-0.84990	1.54712
2024	-1.34799	1.71135	-1.35418	1.26803	-1.67239	1.67875	0.38019					

LS Outlier t-values have been set to zero for the following observations:

2019.Dec 2020.Sep 2023.Jul

No more outliers identified

[Previous Table](#) | [Index](#) | [Next Table](#)

## Backward deletion pass 1

Backward deletion results			
	Parameter Estimate	Standard Error	t-value
-LS2020.Sep	0.0212	0.00567	3.73

Estimation converged in 25 ARMA iterations, 116 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

## Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
AO2013.Nov	0.0183	0.00348	5.25
LS2019.Dec	0.0316	0.00635	4.98
AO2020.Apr	0.0218	0.00349	6.25
LS2023.Jul	0.0410	0.00624	6.56

[Previous Table](#) | [Index](#) | [Next Table](#)

## ARIMA Model

ARIMA Model: (0 1 1)(0 1 1)

Nonseasonal differences: 1

Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal MA		

<b>Lag 1</b>	-0.62767	0.06024
<b>Seasonal MA</b>		
<b>Lag 12</b>	0.99964	0.06448

**Model Innovation Variance**

<b>Variance</b>	0.58792E-04
<b>Standard Error of Variance</b>	0.73779E-05

[Previous Table](#) | [Index](#) | [Next Table](#)

**Final AO Outlier t-values****Final AO Outlier t-values**

	<u>Jan</u>	<u>Feb</u>	<u>Mar</u>	<u>Apr</u>	<u>May</u>	<u>Jun</u>	<u>Jul</u>	<u>Aug</u>	<u>Sep</u>	<u>Oct</u>	<u>Nov</u>	<u>Dec</u>
<b>2013</b>	-2.3681	2.1785	-1.5708	1.3752	-1.7182	1.5760	-1.1000	0.1616	0.3672	0.2098	0.0000	1.2716
<b>2014</b>	-0.9192	-0.3990	0.8141	-1.2170	2.2121	-3.1460	2.3973	-1.1030	1.4002	-1.0591	0.1873	-0.0519
<b>2015</b>	-0.3862	0.7859	-0.8370	1.3566	-2.2259	2.9586	-2.8818	1.2618	0.2473	-0.6783	0.9944	-1.7557
<b>2016</b>	2.3510	-1.7334	0.6284	-0.3067	-0.0258	0.5213	-0.0625	0.1097	-0.4862	0.9585	-1.2163	0.7921
<b>2017</b>	-0.7859	0.5747	0.0180	-0.1077	0.3978	-1.0822	0.4525	0.5350	-0.0399	-1.2250	1.5010	-1.2510
<b>2018</b>	1.5842	-1.4668	1.2065	-0.9873	0.8345	-0.7283	0.4671	-0.3458	0.7578	-0.6594	-0.1475	0.1699
<b>2019</b>	-0.1934	-0.0557	0.3941	-0.4626	0.6022	-0.4873	0.4467	-1.0193	0.9875	-1.0426	1.5525	-0.6315
<b>2020</b>	1.3696	-0.5694	-1.0729	0.0000	1.6620	-1.4602	0.8854	-0.4636	-1.6211	1.1291	-0.0283	0.6590
<b>2021</b>	-1.3912	0.8983	-0.2377	-0.7413	1.1376	-0.5007	0.2112	-0.1261	-0.8126	0.8135	-0.4565	0.6449
<b>2022</b>	-0.2775	-0.6348	0.8031	-0.4667	0.2973	0.9232	-1.3136	0.0975	0.2036	0.5014	-0.1673	-1.1918
<b>2023</b>	1.4436	-0.9419	0.6604	0.0324	-0.8416	1.0228	0.0184	1.4051	-2.0122	1.2219	-1.3114	1.5777
<b>2024</b>	-1.6674	1.6695	-1.4290	1.6034	-1.8347	0.7577	0.8922	-1.2884				

AO Outlier t-values have been set to zero for the following observations:

2013.Nov 2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

**Final LS Outlier t-values****Final LS Outlier t-values**

	<u>Jan</u>	<u>Feb</u>	<u>Mar</u>	<u>Apr</u>	<u>May</u>	<u>Jun</u>	<u>Jul</u>	<u>Aug</u>	<u>Sep</u>	<u>Oct</u>	<u>Nov</u>	<u>Dec</u>
<b>2013</b>			-1.57036	1.28481	-1.22558	1.91598	-0.96507	1.04472	0.80775	0.22828	-0.07566	-0.07566
<b>2014</b>	-1.65522	0.10490	0.80321	-0.68654	1.53916	-2.51789	3.25834	-1.14432	0.87984	-1.69062	0.25263	-0.09079
<b>2015</b>	0.00475	0.71219	-0.73124	0.80604	-1.68505	2.40267	-3.03048	2.25949	-0.05052	-0.50410	0.74128	-1.08347
<b>2016</b>	2.13614	-2.17988	1.00450	-0.14922	0.41395	0.46122	-0.49597	-0.38132	-0.58492	0.30783	-1.45276	0.77796
<b>2017</b>	-0.67639	0.76614	-0.28973	-0.32323	-0.12549	-0.85581	1.13163	0.30103	-0.68161	-0.60724	1.64253	-1.11060
<b>2018</b>	1.18542	-1.72179	0.97277	-1.24344	0.56924	-0.96322	0.37320	-0.48430	0.14945	-1.24072	-0.03155	0.23928
<b>2019</b>	-0.07314	0.28149	0.38353	-0.34050	0.50905	-0.59676	0.29577	-0.52388	1.35047	-0.40602	1.55248	0.00000
<b>2020</b>	0.63149	-1.69559	-0.79053	1.10617	1.10617	-1.54232	1.04942	-0.46357	3.73203	1.62106	-0.48745	-0.40573
<b>2021</b>	-1.57362	0.97976	-0.66972	-0.23244	1.12839	-0.96123	-0.04297	-0.43086	-0.20095	1.29037	-0.20157	0.63648
<b>2022</b>	-0.54766	-0.03682	1.12842	-0.34613	0.51089	-0.03529	-1.73262	0.67934	0.50320	0.12898	-0.79255	-0.48776
<b>2023</b>	1.69768	-0.94950	0.77534	-0.42530	-0.51745	1.02277	0.00000	-0.01842	-2.28944	1.38835	-0.84990	1.54712
<b>2024</b>	-1.34799	1.71135	-1.35418	1.26803	-1.67239	1.67875	0.38019					

LS Outlier t-values have been set to zero for the following observations:

2019.Dec 2023.Jul

**ARIMA Estimates (H-R) for Unit Root Identification : Model No. 1**

Model Estimated : ( 2 0 0 ) ( 1 0 0 )

Regular  $\hat{A}R$  : 1.3937 -0.4091

Seasonal  $\hat{A}R$  : 0.4728

### ARIMA Estimates (H-R) for Unit Root Identification : Model No. 2

Model Estimated : ( 1 1 1 ) ( 1 0 1 )

Regular  $\hat{A}R$  : 0.0905

Seasonal  $\hat{A}R$  : 0.8439

Regular  $\hat{M}A$  : -0.5307

Seasonal  $\hat{M}A$  : 0.6423

### ARIMA Estimates (H-R) for Unit Root Identification : Model No. 3

Model Estimated : ( 1 1 1 ) ( 1 1 1 )

Regular  $\hat{A}R$  : 0.0558

Seasonal  $\hat{A}R$  : -0.4146

Regular  $\hat{M}A$  : -0.7034

Seasonal  $\hat{M}A$  : 0.2763

Results of Unit Root Test for identifying orders of differencing:

Regular difference order : 1 Seasonal difference order : 1

Mean is not significant.

### ARIMA Estimates and Likelihood Values for $\hat{A}RMA$ Order Identification

Model Estimated : ( 3 1 0 ) ( 0 1 0 )

Regular  $\hat{A}R$  : 0.5119 -0.4035 0.0375

$\hat{B}IC$  : 501.0285

$\hat{B}IC2$  : -6.0625

Model Estimated : ( 3 1 0 ) ( 0 1 1 )

Regular  $\hat{A}R$  : 0.5925 -0.4415 0.0449

Seasonal  $\hat{M}A$  : 0.9921

$\hat{B}IC$  : 442.7914

$\hat{B}IC2$  : -6.5211

Model Estimated : ( 3 1 0 ) ( 1 1 0 )

Regular  $\hat{A}R$  : 0.5105 -0.4266 0.0521

Seasonal  $\hat{A}R$  : -0.5384

$\hat{B}IC$  : 465.2397

$\hat{B}IC2$  : -6.3444

Model Estimated : ( 3 1 0 ) ( 1 1 1 )

Regular  $\hat{A}R$  : 0.6015 -0.4464 0.0450

Seasonal  $\hat{A}R$  : -0.0640

Seasonal  $\hat{M}A$  : 0.9873

$\hat{B}IC$  : 447.2201

$\hat{B}IC2$  : -6.4862

Model Estimated : ( 0 1 0 ) ( 0 1 1 )

Seasonal  $\hat{M}A$  : 0.9716

$\hat{B}IC$  : 473.1318

$\hat{B}IC2$  : -6.2822

Model Estimated : ( 0 1 1 ) ( 0 1 1 )

Regular  $\hat{MA}$  : -0.6276  
Seasonal  $\hat{MA}$  : 0.9938

$\hat{BIC}$  : 438.5922  
 $\hat{BIC}_2$  : -6.5542

Model Estimated : ( 0 1 2 ) ( 0 1 1 )

Regular  $\hat{MA}$  : -0.6086 0.0323  
Seasonal  $\hat{MA}$  : 0.9918

$\hat{BIC}$  : 443.3736  
 $\hat{BIC}_2$  : -6.5165

Model Estimated : ( 1 1 0 ) ( 0 1 1 )

Regular  $\hat{AR}$  : 0.4001  
Seasonal  $\hat{MA}$  : 0.9887

$\hat{BIC}$  : 456.4050  
 $\hat{BIC}_2$  : -6.4139

Model Estimated : ( 1 1 1 ) ( 0 1 1 )

Regular  $\hat{AR}$  : -0.0288  
Regular  $\hat{MA}$  : -0.6460  
Seasonal  $\hat{MA}$  : 0.9932

$\hat{BIC}$  : 443.4008  
 $\hat{BIC}_2$  : -6.5163

Model Estimated : ( 1 1 2 ) ( 0 1 1 )

Regular  $\hat{AR}$  : 0.6867  
Regular  $\hat{MA}$  : 0.1390 0.5367  
Seasonal  $\hat{MA}$  : 0.9929

$\hat{BIC}$  : 445.5434  
 $\hat{BIC}_2$  : -6.4994

Model Estimated : ( 2 1 0 ) ( 0 1 1 )

Regular  $\hat{AR}$  : 0.5746 -0.4161  
Seasonal  $\hat{MA}$  : 0.9929

$\hat{BIC}$  : 438.1908  
 $\hat{BIC}_2$  : -6.5573

Model Estimated : ( 2 1 1 ) ( 0 1 1 )

Regular  $\hat{AR}$  : 0.4845 -0.3795  
Regular  $\hat{MA}$  : -0.1081  
Seasonal  $\hat{MA}$  : 0.9918

$\hat{BIC}$  : 442.7914  
 $\hat{BIC}_2$  : -6.5211

Model Estimated : ( 2 1 2 ) ( 0 1 1 )

Regular  $\hat{AR}$  : 0.4962 -0.3732  
Regular  $\hat{MA}$  : -0.0967 0.0157  
Seasonal  $\hat{MA}$  : 0.9946

$\hat{BIC}$  : 447.6345  
 $\hat{BIC}_2$  : -6.4830

Model Estimated : ( 2 1 0 ) ( 0 1 0 )

Regular AR : 0.4971 -0.3845

BIC : 496.3625  
BIC2 : -6.0993

Best Five ARIMA Models

- Model # 1 : (2 1 0)(0 1 1) (BIC2 = -6.557)
- Model # 2 : (0 1 1)(0 1 1) (BIC2 = -6.554)
- Model # 3 : (2 1 1)(0 1 1) (BIC2 = -6.521)
- Model # 4 : (0 1 2)(0 1 1) (BIC2 = -6.517)
- Model # 5 : (1 1 1)(0 1 1) (BIC2 = -6.516)

Preliminary model choice : (2 1 0)(0 1 1)

[Previous Table](#) | [Index](#) | [Next Table](#)

OUTLIER DETECTION

From 2013.Jan to 2024.Aug  
Observations 140

Types : AO and LS  
Method : add one  
Critical |t| for AO outliers : 3.88  
Critical |t| for LS outliers : 3.88

Estimation converged in 48 ARMA iterations, 193 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (2 1 0)(0 1 1)

Nonseasonal differences: 1  
Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal AR		
Lag 1	0.27238	0.07653
Lag 2	-0.27955	0.07644
Seasonal MA		
Lag 12	0.99754	0.06069

Model Innovation Variance	
Variance	0.10940E-03
Standard Error of Variance	0.13729E-04

[Previous Table](#) | [Index](#) | [Next Table](#)

Forward addition pass 1

Robust root mse : 7.55E-03  
Normal root mse : 1.05E-02

Outliers flagged as significant this pass		
	t(AO)	t(LS)

<b>AO2013.Nov</b>	4.70*	3.52
<b>LS2013.Dec</b>	-2.40	-4.07*
<b>LS2019.Dec</b>	3.94	5.07*
<b>AO2020.Mar</b>	-4.69*	-3.15
<b>AO2020.Apr</b>	5.43*	4.42
<b>LS2020.May</b>	-3.20	-4.34*
<b>AO2023.Jun</b>	-4.37*	-1.39
<b>LS2023.Jul</b>	5.12	5.66*

Add +LS2023.Jul ( 5.66 )

[Previous Table](#) | [Index](#) | [Next Table](#)

## AO Outlier t-values

AO Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
<b>2013</b>	-1.9381	1.3694	-0.4436	0.0228	-0.8590	1.1362	-1.0654	-0.2533	1.9593	-2.8425	4.7024	-2.4032
<b>2014</b>	0.3490	-0.7315	0.3326	-0.8330	1.3501	-1.8590	0.9979	0.7088	0.3059	-0.2349	-0.3599	-0.2204
<b>2015</b>	-0.1125	0.0939	0.4802	-0.0944	-1.0683	2.3476	-2.7613	1.0400	0.2725	-0.3371	0.4355	-1.1551
<b>2016</b>	1.8661	-1.3573	0.2940	-0.1959	-0.0374	1.0975	-0.1185	0.3068	-0.4461	0.7040	-1.0946	0.1638
<b>2017</b>	-0.3497	0.2125	0.5024	-0.5011	0.4470	-0.8177	-0.3354	1.5389	-1.0280	-0.4765	0.8686	-0.4915
<b>2018</b>	1.0370	-0.7311	0.8043	-0.6806	0.3506	0.1501	-0.4414	0.5230	0.1503	-0.4237	-0.6356	0.1019
<b>2019</b>	-0.3522	-0.1023	0.6701	-0.5713	0.4651	0.1841	-0.6567	-0.2761	-0.6713	1.4631	-3.3125	3.9362
<b>2020</b>	-0.9013	2.0637	-4.6941	5.4301	-3.1960	0.3418	0.6006	-2.2179	1.5904	0.0858	0.9934	0.2320
<b>2021</b>	-1.1648	0.5136	0.0322	-1.2299	1.4178	-0.1168	-0.4237	0.1175	-1.2825	0.8545	-0.3806	0.5311
<b>2022</b>	-0.1573	-0.8902	1.3614	-0.7934	0.6752	1.3492	-1.8328	0.2696	-0.0739	0.8500	-0.6547	-1.1005
<b>2023</b>	1.3650	-0.8667	0.7842	-1.0121	1.5999	-4.3652	5.1184	-1.0990	-0.7765	0.3572	-0.5620	0.4056
<b>2024</b>	-0.4575	0.5158	-0.1295	0.4589	-1.1450	0.5594	0.9827	-1.1375				

[Previous Table](#) | [Index](#) | [Next Table](#)

## LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
<b>2013</b>			-0.26822	0.44757	0.41085	1.79718	-0.03640	1.68315	2.09867	-1.06354	3.52415	-4.06534
<b>2014</b>	-0.18671	-0.74807	0.43249	-0.10432	1.24014	-0.93891	2.06173	0.45171	-0.69220	-1.18590	-0.80681	-0.22599
<b>2015</b>	0.12972	0.31035	0.15878	-0.61628	-0.46389	1.26035	-2.52898	1.92709	0.25324	-0.18657	0.35745	-0.34543
<b>2016</b>	1.51884	-1.49174	0.69884	0.22417	0.54042	0.60082	-1.17059	-0.97959	-1.47898	-0.75898	-1.89522	-0.12856
<b>2017</b>	-0.39289	0.17174	-0.17113	-0.98188	-0.17314	-0.89454	0.42513	0.96653	-1.51720	0.14203	0.91107	-0.49083
<b>2018</b>	0.30239	-1.36863	-0.18854	-1.48639	-0.38790	-0.95383	-1.19643	-0.48428	-1.33136	-1.57396	-0.89012	0.13567
<b>2019</b>	-0.02885	0.53849	0.70337	-0.37826	0.54376	-0.20688	-0.50408	0.55570	1.00400	2.08740	-0.27408	5.07225
<b>2020</b>	-1.28067	0.17538	-3.15456	4.42169	-4.34237	0.81590	0.26433	-0.70490	2.87708	0.31027	0.17179	-1.43156
<b>2021</b>	-1.80593	0.07629	-0.75247	-0.80422	1.18073	-1.10750	-0.91929	-0.23567	-0.42649	1.64343	0.26434	0.87864
<b>2022</b>	0.02153	0.27482	1.71109	-0.48644	0.79413	-0.29559	-2.47363	0.48379	0.04974	0.16908	-1.20281	-0.14612
<b>2023</b>	1.63001	-0.57399	0.82470	-0.44106	1.19246	-1.38976	5.65636	-2.60328	-0.83649	0.41673	-0.15975	0.74724
<b>2024</b>	0.09256	0.82892	-0.00352	0.20541	-0.53519	1.31279	0.43683					

Average absolute percentage error in within-sample forecasts:

Last year: 1.00  
 Last-1 year: 0.71  
 Last-2 year: 2.47  
 Last three years: 1.40

Estimation converged in 53 ARIMA iterations, 231 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
LS2023.Jul	0.0422	0.00933	4.52

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (2 1 0)(0 1 1)

Nonseasonal differences: 1  
Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal AR		
Lag 1	0.33159	0.07701
Lag 2	-0.26931	0.07702
Seasonal MA		
Lag 12	0.99731	0.06048

Model Innovation Variance	
Variance	0.94636E-04
Standard Error of Variance	0.11876E-04

[Previous Table](#) | [Index](#) | [Next Table](#)

Forward addition pass 2

Robust root mse : 6.68E-03  
Normal root mse : 9.73E-03

Outliers flagged as significant this pass		
	t(AO)	t(LS)
AO2013.Nov	5.42*	4.24
LS2013.Dec	-2.84	-4.69*
AO2019.Nov	-3.96*	-0.78
LS2019.Dec	4.49	5.74*
AO2020.Mar	-5.50*	-3.62
AO2020.Apr	6.32*	5.44
LS2020.May	-3.53	-4.97*

Add +AO2020.Apr ( 6.32 )

[Previous Table](#) | [Index](#) | [Next Table](#)

AO Outlier t-values

AO Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.19783	1.58223	-0.49204	-0.06792	-0.75444	0.88820	-0.73901	-0.50219	2.31309	-3.46464	5.42009	-2.84003
2014	0.42324	-0.74094	0.42622	-1.02502	1.82494	-2.69367	1.61775	0.53110	0.36891	-0.27333	-0.34813	-0.19184
2015	-0.13054	0.09426	0.54544	-0.23886	-1.02236	2.31083	-2.73174	0.99799	0.43294	-0.40534	0.48699	-1.35817
2016	2.18501	-1.57297	0.36599	-0.30035	0.11685	0.71307	0.28895	0.08109	-0.42015	0.87151	-1.22062	0.24412
2017	-0.39799	0.22577	0.54988	-0.66528	0.78472	-1.44403	0.10781	1.59255	-1.12289	-0.54875	1.03087	-0.61340
2018	1.14477	-0.88089	0.93246	-0.87573	0.63114	-0.32545	-0.05240	0.36629	0.28138	-0.45470	-0.66368	0.19826
2019	-0.39485	-0.11984	0.76173	-0.79502	0.74263	-0.25552	-0.22856	-0.48663	-0.56337	1.75104	-3.96297	4.49008
2020	-1.11204	2.31183	-5.50435	6.32314	-3.53366	-0.06670	1.34421	-2.78545	1.90080	-0.01216	1.05205	0.23129
2021	-1.34509	0.69858	0.07917	-1.55443	1.84562	-0.65665	0.02916	-0.00893	-1.34323	1.01773	-0.46320	0.61044
2022	-0.18491	-1.01753	1.58192	-1.11179	0.92893	1.09032	-1.63402	0.10900	0.03137	1.01146	-0.75417	-1.25069
2023	1.61743	-1.04444	0.88852	-0.16320	-0.45409	0.46869	0.90723	1.03476	-1.89675	0.50715	-0.57718	0.47993
2024	-0.57418	0.57398	-0.14048	0.47288	-1.13957	0.17479	1.59186	-1.54092				

[Previous Table](#) | [Index](#) | [Next Table](#)

LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-0.41480	0.39919	0.51114	1.75370	0.29571	1.50901	2.34169	-1.46768	4.23779	-4.68788
2014	-0.01098	-0.70638	0.51368	-0.18828	1.49989	-1.50493	2.93178	0.26410	-0.61050	-1.21780	-0.76770	-0.19441
2015	0.12152	0.33559	0.18028	-0.71791	-0.32466	1.35884	-2.44704	2.04943	0.41097	-0.30203	0.36548	-0.43648
2016	1.80012	-1.79644	0.79379	0.19093	0.68565	0.49343	-0.68069	-1.15323	-1.29062	-0.59846	-2.03364	-0.02355
2017	-0.42555	0.22989	-0.14187	-1.04728	0.04822	-1.24402	1.13270	0.95186	-1.67022	0.17918	1.08284	-0.61476
2018	0.39537	-1.48703	-0.03626	-1.57167	-0.12968	-1.16908	-0.63692	-0.54875	-1.15432	-1.61735	-0.86856	0.22438
2019	-0.10211	0.54704	0.74412	-0.51034	0.79894	-0.42377	-0.00401	0.37155	1.17491	2.10227	-0.78130	5.74482
2020	-1.64933	0.18369	-3.62228	5.44222	-4.97082	0.84681	0.96014	-1.25165	3.33582	0.20507	0.22510	-1.50740
2021	-1.88829	0.32850	-0.82169	-0.95184	1.60809	-1.43074	-0.35337	-0.40013	-0.38670	1.82520	0.14922	0.91201
2022	-0.09325	0.21091	1.88596	-0.71934	1.11161	-0.41779	-2.21770	0.47437	0.29623	0.24449	-1.42116	-0.17921
2023	1.88040	-0.78385	0.93589	-0.52740	-0.26829	0.46869	0.00000	-0.90723	-2.51214	0.64710	-0.18806	0.76243
2024	-0.02790	0.91566	-0.02962	0.20172	-0.57708	1.29937	1.05160					

LS Outlier t-values have been set to zero for the following observations:

2023.Jul

Average absolute percentage error in within-sample forecasts:

- Last year: 0.96
- Last-1 year: 0.65
- Last-2 year: 2.76
- Last three years: 1.46

Estimation converged in 36 ARMA iterations, 168 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
AO2020.Apr	0.0248	0.00469	5.29



LS2023.Jul	0.0418	0.00810	5.17
------------	--------	---------	------

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (2 1 0)(0 1 1)

Nonseasonal differences: 1  
Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal AR		
Lag 1	0.46011	0.07866
Lag 2	-0.33278	0.07882
Seasonal MA		
Lag 12	0.90090	0.05423

Model Innovation Variance	
Variance	0.85627E-04
Standard Error of Variance	0.10745E-04

[Previous Table](#) | [Index](#) | [Next Table](#)

Forward addition pass 3

Robust root mse : 7.08E-03  
Normal root mse : 9.25E-03

Outliers flagged as significant this pass		
	t(AO)	t(LS)
AO2013.Oct	-4.02*	-2.12
AO2013.Nov	5.57*	4.76
LS2013.Dec	-3.14	-4.78*
AO2019.Nov	-4.34*	-1.69
LS2019.Dec	4.58	5.74*

Add +LS2019.Dec ( 5.74 )

[Previous Table](#) | [Index](#) | [Next Table](#)

AO Outlier t-values

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.24805	1.78978	-1.03876	0.66962	-1.23486	1.22372	-0.77869	-0.60654	2.52198	-4.01890	5.57429	-3.14340
2014	0.68673	-0.45381	0.05105	-0.61140	1.71121	-2.69783	1.76522	0.19864	0.15275	-0.07383	-0.39038	0.05870
2015	-0.25610	0.22545	0.09121	0.40612	-1.63915	2.82349	-2.94101	1.11432	0.29624	-0.38408	0.52430	-1.45178
2016	2.24721	-1.53492	0.10587	0.36433	-0.43392	0.79827	0.17926	0.01926	-0.50976	1.06998	-1.32262	0.49618
2017	-0.51813	0.39085	0.02812	-0.12959	0.55708	-1.23627	0.02028	1.56223	-1.20037	-0.43569	1.12162	-0.81991
2018	1.10831	-0.79810	0.59029	-0.32928	0.23929	-0.11654	-0.12811	0.26986	0.27335	-0.40532	-0.52445	0.30129

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2019	-0.47696	0.06119	0.32779	-0.28414	0.29322	-0.04770	-0.20650	-0.42243	-0.53839	2.12585	-4.33979	4.58351
2020	-1.09935	0.55361	-1.25943	0.00000	1.61271	-2.29923	2.32497	-2.93988	2.11088	-0.47143	1.01137	0.20085
2021	-1.39624	1.10037	-0.31983	-1.06843	1.46665	-0.64540	0.00531	0.21412	-1.27570	1.02604	-0.54872	0.55670
2022	-0.16537	-0.82846	1.27409	-0.75594	0.34200	1.36539	-1.72192	0.30647	-0.02432	0.87636	-0.56413	-1.29412
2023	1.74888	-1.09481	0.44178	0.49766	-0.89787	0.67318	0.53469	1.12400	-1.81814	0.66044	-0.48848	0.46905
2024	-0.73835	0.81350	-0.73633	1.26574	-1.59612	0.37977	1.35128	-1.45089				

AO Outlier t-values have been set to zero for the following observations:

2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-0.8346	0.9523	-0.1924	1.9183	-0.1730	1.1578	2.1979	-2.1173	4.7589	-4.7785
2014	0.5998	-0.5743	0.2021	0.1150	1.1603	-1.7658	2.8495	-0.1719	-0.5122	-0.7734	-0.6471	0.0209
2015	-0.0796	0.3576	-0.0281	-0.1843	-0.8786	1.9238	-2.9063	2.1242	0.2206	-0.2863	0.3708	-0.5262
2016	1.9577	-1.8849	0.7410	0.5609	-0.0618	0.6798	-0.6855	-0.9904	-1.0252	-0.1528	-1.9835	0.2795
2017	-0.5695	0.3170	-0.3517	-0.4004	-0.1790	-1.1305	0.9836	0.9468	-1.7255	0.3286	1.0741	-0.8450
2018	0.5578	-1.3364	0.0289	-0.9817	-0.4191	-0.8271	-0.6297	-0.4094	-0.8722	-1.3396	-0.6461	0.2512
2019	-0.2643	0.5510	0.4464	-0.1138	0.3718	-0.1298	-0.0485	0.3043	1.0282	1.9489	-1.6883	5.7369
2020	-2.1053	-0.2317	-1.2360	0.6105	0.6105	-1.6800	2.2101	-1.7749	3.2534	-0.3590	0.4476	-1.2828
2021	-1.6265	0.7626	-1.1199	-0.5752	1.2505	-1.2570	-0.1564	-0.1649	-0.5321	1.6505	-0.1050	0.8338
2022	-0.1187	0.1641	1.5817	-0.5961	0.6957	0.1103	-2.2292	0.7171	0.1951	0.2366	-1.2629	-0.2976
2023	1.9166	-1.0756	0.7973	0.0428	-0.8250	0.6732	0.0000	-0.5347	-2.3005	0.8499	-0.2801	0.5557
2024	-0.2468	1.0153	-0.3763	0.8836	-1.2796	1.4492	0.8509					

LS Outlier t-values have been set to zero for the following observations:

2023.Jul

Average absolute percentage error in within-sample forecasts:

- Last year: 1.04
- Last-1 year: 0.74
- Last-2 year: 3.05
- Last three years: 1.61

Estimation converged in 27 ARMA iterations, 137 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
LS2019.Dec	0.0351	0.00730	4.81
AO2020.Apr	0.0248	0.00421	5.89
LS2023.Jul	0.0418	0.00731	5.72

[Previous Table](#) | [Index](#) | [Next Table](#)

## ARIMA Model

ARIMA Model: (2 1 0)(0 1 1)

Nonseasonal differences: 1

Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal AR		
Lag 1	0.49526	0.07776
Lag 2	-0.35426	0.07767
Seasonal MA		
Lag 12	0.92077	0.05592

Model Innovation Variance	
Variance	0.71499E-04
Standard Error of Variance	0.89725E-05

[Previous Table](#) | [Index](#) | [Next Table](#)

## Forward addition pass 4

Robust root mse : 7.75E-03

Normal root mse : 8.46E-03

Outliers flagged as significant this pass		
	t(AO)	t(LS)
AO2013.Nov	4.76*	4.26
LS2013.Dec	-2.57	-3.96*

Add +AO2013.Nov ( 4.76 )

[Previous Table](#) | [Index](#) | [Next Table](#)

## AO Outlier t-values

AO Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.042472	1.632971	-0.979500	0.648015	-1.141465	1.148111	-0.729858	-0.548035	2.310980	-3.630801	4.763231	-2.565750
2014	0.490144	-0.300230	0.037966	-0.586514	1.602535	-2.466496	1.611319	0.127629	0.065802	0.116438	-0.701342	0.470837
2015	-0.449313	0.271865	0.062947	0.388087	-1.538943	2.641699	-2.727090	1.061873	0.194099	-0.180057	0.113131	-0.954351
2016	1.859945	-1.348805	0.126732	0.350759	-0.427137	0.709454	0.127914	0.028412	-0.541492	1.205378	-1.640163	0.926955
2017	-0.696338	0.425322	0.000294	-0.130804	0.551826	-1.116658	-0.002757	1.424545	-1.179473	-0.151193	0.589477	-0.339414
2018	0.787816	-0.669112	0.554850	-0.312731	0.233050	-0.108227	-0.122703	0.226850	0.159942	-0.110823	-0.943263	0.792722
2019	-0.681614	0.136791	0.296564	-0.280825	0.273714	-0.051693	-0.162063	-0.380012	0.291857	-0.255001	0.922257	-0.609469
2020	1.279988	-0.300640	-1.110486	0.000000	1.521077	-2.156223	2.225532	-2.720169	1.860162	-0.246938	0.377776	0.705404
2021	-1.521170	1.128904	-0.279647	-0.995043	1.349332	-0.631613	0.030055	0.224256	-1.255238	1.224687	-1.051935	1.009642
2022	-0.380090	-0.668981	1.187304	-0.735355	0.273148	1.256243	-1.541952	0.306753	-0.163816	1.059348	-1.000090	-0.675174
2023	1.380832	-0.965323	0.401921	0.472808	-0.820944	0.607025	0.449554	1.019108	-1.756547	0.932722	-0.958588	0.927833
2024	-0.931764	0.824789	-0.697610	1.198860	-1.461058	0.359651	1.205240	-1.354545				

AO Outlier t-values have been set to zero for the following observations:

2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

LS Outlier t-values

LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-0.8014	0.8999	-0.2174	1.7506	-0.2284	1.0298	1.9778	-2.0110	4.2551	-3.9604
2014	0.4713	-0.3742	0.1440	0.0787	1.0901	-1.6739	2.5823	-0.1996	-0.4203	-0.5339	-0.7354	0.4739
2015	-0.3385	0.4361	-0.0331	-0.1419	-0.8112	1.8428	-2.7156	1.9893	0.1596	-0.1754	0.1353	-0.0597
2016	1.5837	-1.6238	0.7040	0.4864	-0.1184	0.6180	-0.6058	-0.8250	-0.8756	0.0590	-2.0218	0.8050
2017	-0.7939	0.4079	-0.3261	-0.3272	-0.1018	-1.0527	0.8734	0.8762	-1.5816	0.4542	0.7156	-0.3003
2018	0.2851	-1.0722	0.0827	-0.8754	-0.3364	-0.7374	-0.5524	-0.3397	-0.7321	-1.0083	-0.8179	0.8089
2019	-0.5588	0.6167	0.3805	-0.1308	0.3533	-0.1190	-0.0300	0.2492	0.9059	0.4220	0.9223	0.0000
2020	0.6095	-1.5175	-1.0715	0.5930	0.5930	-1.5903	2.0920	-1.7558	2.9369	-0.2739	0.1522	-0.5001
2021	-1.7120	0.9131	-1.0350	-0.5545	1.1609	-1.1663	-0.0791	-0.1306	-0.5182	1.6484	-0.4643	1.3518
2022	-0.3915	0.2643	1.4186	-0.6288	0.6389	0.1672	-2.0032	0.6579	0.1302	0.4129	-1.4155	0.3072
2023	1.4681	-0.9146	0.7514	0.0589	-0.7733	0.6070	0.0000	-0.4496	-2.0508	1.0158	-0.5935	1.0608
2024	-0.5405	1.0659	-0.3576	0.8465	-1.2204	1.2994	0.7262					

LS Outlier t-values have been set to zero for the following observations:

2019.Dec 2023.Jul

Average absolute percentage error in within-sample forecasts:

Last year: 0.97  
Last-1 year: 0.77  
Last-2 year: 2.59  
Last three years: 1.44

Estimation converged in 21 ARMA iterations, 118 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
AO2013.Nov	0.0184	0.00365	5.05
LS2019.Dec	0.0317	0.00642	4.94
AO2020.Apr	0.0248	0.00362	6.85
LS2023.Jul	0.0417	0.00637	6.55

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (2 1 0)(0 1 1)

Nonseasonal differences: 1  
Seasonal differences: 1

ARIMA Model

	Estimate	Standard Error
<b>Nonseasonal AR</b>		
<b>Lag 1</b>	0.58028	0.07350
<b>Lag 2</b>	-0.41506	0.07320
<b>Seasonal MA</b>		
<b>Lag 12</b>	0.99927	0.06455

Model Innovation Variance

<b>Variance</b>	0.56106E-04
<b>Standard Error of Variance</b>	0.70408E-05

[Previous Table](#) | [Index](#) | [Next Table](#)

## Forward addition pass 5

Robust root mse : 6.53E-03

Normal root mse : 7.49E-03

[Previous Table](#) | [Index](#) | [Next Table](#)

## AO Outlier t-values

AO Outlier t-values

	<u>Jan</u>	<u>Feb</u>	<u>Mar</u>	<u>Apr</u>	<u>May</u>	<u>Jun</u>	<u>Jul</u>	<u>Aug</u>	<u>Sep</u>	<u>Oct</u>	<u>Nov</u>	<u>Dec</u>
<b>2013</b>	-2.35229	1.89681	-1.22775	0.85897	-1.37783	1.43074	-0.90143	-0.12683	0.93044	-0.39006	0.00000	1.76515
<b>2014</b>	-1.36669	0.26046	0.02152	-0.76958	1.99460	-2.90778	1.90705	-0.05323	0.27866	-0.37368	-0.01850	0.06624
<b>2015</b>	-0.32858	0.24729	0.02193	0.50433	-1.92013	3.27213	-3.30964	1.30261	0.42277	-0.77249	0.91802	-1.74619
<b>2016</b>	2.45308	-1.69681	0.22843	0.45901	-0.57966	0.78701	0.06700	0.00317	-0.43001	0.98963	-1.28669	0.69192
<b>2017</b>	-0.61883	0.42840	-0.06590	-0.18774	0.76026	-1.27833	-0.05534	1.60763	-1.18696	-0.58020	1.37571	-1.00932
<b>2018</b>	1.14912	-0.88876	0.69440	-0.39958	0.30893	-0.12861	-0.15360	0.16041	0.38101	-0.53342	-0.44171	0.53465
<b>2019</b>	-0.61091	0.10345	0.34240	-0.38320	0.33301	-0.08139	-0.12057	-0.48400	0.49472	-0.47693	1.14037	-0.80115
<b>2020</b>	1.36941	-0.28443	-1.19904	0.00000	1.90363	-2.68973	2.88609	-3.33975	2.45289	-0.90636	0.92414	0.42596
<b>2021</b>	-1.58320	1.37444	-0.29928	-1.21856	1.60922	-0.85608	0.10030	0.29216	-1.28344	1.11254	-0.80551	0.76163
<b>2022</b>	-0.21511	-0.82527	1.45842	-0.97872	0.21944	1.49680	-1.73566	0.38744	-0.13357	0.85058	-0.58860	-1.17536
<b>2023</b>	1.87487	-1.30915	0.46808	0.60237	-0.96364	0.68619	0.40733	1.13110	-1.88375	0.88891	-0.62604	0.69161
<b>2024</b>	-0.95280	0.92444	-0.89164	1.52000	-1.72277	0.46188	1.30470	-1.61776				

AO Outlier t-values have been set to zero for the following observations:

2013.Nov 2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

## LS Outlier t-values

LS Outlier t-values

	<u>Jan</u>	<u>Feb</u>	<u>Mar</u>	<u>Apr</u>	<u>May</u>	<u>Jun</u>	<u>Jul</u>	<u>Aug</u>	<u>Sep</u>	<u>Oct</u>	<u>Nov</u>	<u>Dec</u>
<b>2013</b>			-1.0328	1.1446	-0.3634	2.0555	-0.4555	1.1269	1.3681	-0.2013	0.5515	0.5515
<b>2014</b>	-1.9589	0.5201	0.0578	0.0200	1.3715	-2.1314	2.9774	-0.3747	-0.2818	-0.7722	-0.1178	-0.0858
<b>2015</b>	-0.2010	0.3756	-0.0591	-0.0978	-0.9834	2.3882	-3.3608	2.4528	0.1677	-0.5756	0.7815	-0.8287
<b>2016</b>	2.2314	-2.0771	0.9052	0.5052	-0.3007	0.7171	-0.6652	-0.7814	-0.7885	-0.0332	-1.7750	0.4792
<b>2017</b>	-0.7345	0.3533	-0.3996	-0.2844	0.0452	-1.2893	0.9557	1.0507	-1.7734	0.3120	1.3344	-1.0779
<b>2018</b>	0.6950	-1.3215	0.2407	-0.9799	-0.2785	-0.8204	-0.5964	-0.3256	-0.6083	-1.2790	-0.3456	0.4292
<b>2019</b>	-0.5089	0.5638	0.3817	-0.2197	0.4532	-0.1319	0.0109	0.2224	1.0737	0.2234	1.1404	0.0000
<b>2020</b>	0.8012	-1.5225	-1.1108	0.7828	0.7828	-2.0113	2.6728	-2.4089	3.4576	-0.8521	0.7394	-0.8817

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2021	-1.6207	1.1609	-1.2544	-0.7304	1.4092	-1.4173	0.0847	-0.0915	-0.6053	1.6510	-0.2999	1.1145
2022	-0.2255	0.1524	1.6021	-0.9595	0.7589	0.3727	-2.2591	0.7908	0.1112	0.3462	-1.1493	-0.1194
2023	1.9373	-1.3568	0.9439	0.1226	-0.9590	0.6862	0.0000	-0.4073	-2.1871	1.1660	-0.3934	0.7052
2024	-0.5095	1.1622	-0.4626	1.1045	-1.5644	1.4616	0.7089					

LS Outlier t-values have been set to zero for the following observations:

2019.Dec 2023.Jul

No more outliers identified

[Previous Table](#) | [Index](#) | [Next Table](#)

## Final AO Outlier t-values

Final AO Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	-2.35229	1.89681	-1.22775	0.85897	-1.37783	1.43074	-0.90143	-0.12683	0.93044	-0.39006	0.00000	1.76515
2014	-1.36669	0.26046	0.02152	-0.76958	1.99460	-2.90778	1.90705	-0.05323	0.27866	-0.37368	-0.01850	0.06624
2015	-0.32858	0.24729	0.02193	0.50433	-1.92013	3.27213	-3.30964	1.30261	0.42277	-0.77249	0.91802	-1.74619
2016	2.45308	-1.69681	0.22843	0.45901	-0.57966	0.78701	0.06700	0.00317	-0.43001	0.98963	-1.28669	0.69192
2017	-0.61883	0.42840	-0.06590	-0.18774	0.76026	-1.27833	-0.05534	1.60763	-1.18696	-0.58020	1.37571	-1.00932
2018	1.14912	-0.88876	0.69440	-0.39958	0.30893	-0.12861	-0.15360	0.16041	0.38101	-0.53342	-0.44171	0.53465
2019	-0.61091	0.10345	0.34240	-0.38320	0.33301	-0.08139	-0.12057	-0.48400	0.49472	-0.47693	1.14037	-0.80115
2020	1.36941	-0.28443	-1.19904	0.00000	1.90363	-2.68973	2.88609	-3.33975	2.45289	-0.90636	0.92414	0.42596
2021	-1.58320	1.37444	-0.29928	-1.21856	1.60922	-0.85608	0.10030	0.29216	-1.28344	1.11254	-0.80551	0.76163
2022	-0.21511	-0.82527	1.45842	-0.97872	0.21944	1.49680	-1.73566	0.38744	-0.13357	0.85058	-0.58860	-1.17536
2023	1.87487	-1.30915	0.46808	0.60237	-0.96364	0.68619	0.40733	1.13110	-1.88375	0.88891	-0.62604	0.69161
2024	-0.95280	0.92444	-0.89164	1.52000	-1.72277	0.46188	1.30470	-1.61776				

AO Outlier t-values have been set to zero for the following observations:

2013.Nov 2020.Apr

[Previous Table](#) | [Index](#) | [Next Table](#)

## Final LS Outlier t-values

Final LS Outlier t-values												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013			-1.0328	1.1446	-0.3634	2.0555	-0.4555	1.1269	1.3681	-0.2013	0.5515	0.5515
2014	-1.9589	0.5201	0.0578	0.0200	1.3715	-2.1314	2.9774	-0.3747	-0.2818	-0.7722	-0.1178	-0.0858
2015	-0.2010	0.3756	-0.0591	-0.0978	-0.9834	2.3882	-3.3608	2.4528	0.1677	-0.5756	0.7815	-0.8287
2016	2.2314	-2.0771	0.9052	0.5052	-0.3007	0.7171	-0.6652	-0.7814	-0.7885	-0.0332	-1.7750	0.4792
2017	-0.7345	0.3533	-0.3996	-0.2844	0.0452	-1.2893	0.9557	1.0507	-1.7734	0.3120	1.3344	-1.0779
2018	0.6950	-1.3215	0.2407	-0.9799	-0.2785	-0.8204	-0.5964	-0.3256	-0.6083	-1.2790	-0.3456	0.4292
2019	-0.5089	0.5638	0.3817	-0.2197	0.4532	-0.1319	0.0109	0.2224	1.0737	0.2234	1.1404	0.0000
2020	0.8012	-1.5225	-1.1108	0.7828	0.7828	-2.0113	2.6728	-2.4089	3.4576	-0.8521	0.7394	-0.8817
2021	-1.6207	1.1609	-1.2544	-0.7304	1.4092	-1.4173	0.0847	-0.0915	-0.6053	1.6510	-0.2999	1.1145
2022	-0.2255	0.1524	1.6021	-0.9595	0.7589	0.3727	-2.2591	0.7908	0.1112	0.3462	-1.1493	-0.1194
2023	1.9373	-1.3568	0.9439	0.1226	-0.9590	0.6862	0.0000	-0.4073	-2.1871	1.1660	-0.3934	0.7052
2024	-0.5095	1.1622	-0.4626	1.1045	-1.5644	1.4616	0.7089					

LS Outlier t-values have been set to zero for the following observations:

Final Checks for Identified Model

Checking for Unit Roots.

No unit root found.

Nonseasonal ~~MA~~ not within 0.001 of 1.0 - model passes test.

Checking for insignificant ~~ARMA~~ coefficients.

**Final automatic model choice : (2 1 0)(0 1 1)**

End of automatic model selection procedure.

Average absolute percentage error in within-sample forecasts:

Last year: 0.97  
Last-1 year: 0.77  
Last-2 year: 2.59  
Last three years: 1.44

Estimation converged in 21 ~~ARMA~~ iterations, 118 function evaluations.

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Model

Regression Model			
	Parameter Estimate	Standard Error	t-value
Automatically Identified Outliers			
AO2013.Nov	0.0184	0.00365	5.05
LS2019.Dec	0.0317	0.00642	4.94
AO2020.Apr	0.0248	0.00362	6.85
LS2023.Jul	0.0417	0.00637	6.55

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Parameter Correlation Matrix

Regression Parameter Correlation Matrix				
	AO2013.Nov	LS2019.Dec	AO2020.Apr	LS2023.Jul
AO2013.Nov	1.0000			
LS2019.Dec	-0.0879	1.0000		
AO2020.Apr	0.0000	0.0000	1.0000	
LS2023.Jul	-0.0000	-0.0000	-0.0142	1.0000

[Previous Table](#) | [Index](#) | [Next Table](#)

ARIMA Model

ARIMA Model: (2 1 0)(0 1 1)

Nonseasonal differences: 1  
Seasonal differences: 1

ARIMA Model		
	Estimate	Standard Error
Nonseasonal AR		
Lag 1	0.58028	0.07350
Lag 2	-0.41506	0.07320
Seasonal MA		
Lag 12	0.99927	0.06455

Model Innovation Variance	
Variance	0.56106E-04
Standard Error of Variance	0.70408E-05

[Previous Table](#) | [Index](#) | [Next Table](#)

ARMA Parameter Correlation Matrix

ARMA Parameter Correlation Matrix			
	Nonseasonal AR Lag 1	Nonseasonal AR Lag 2	Seasonal MA Lag 12
Nonseasonal AR Lag 1	1.0000		
Nonseasonal AR Lag 2	-0.4195	1.0000	
Seasonal MA Lag 12	-0.0558	0.0662	1.0000

[Previous Table](#) | [Index](#) | [Next Table](#)

Likelihood Statistics	
Number of observations (nobs)	140
Effective number of observations (nefobs)	127
Number of parameters estimated (np)	8
Log likelihood	426.4309
Transformation Adjustment	-635.4861
Adjusted Log likelihood (L)	-209.0552
AIC	434.1104
AICC (F-corrected-AIC)	435.3308
Hannan Quinn	443.3549
BIC	456.8639

nobs = number of observations  
nefobs = nobs - total order of differencing operators  
np = number of estimated regression and ARIMA model parameters including the variance  
V = Covariance matrix of z's  
z = Vector of data - regression values  
c = Vector of prior adjustment factors  
Log likelihood =  $-\text{[nefobs} \cdot \log(2 \cdot \pi) + \log|V| + z'(V\text{-inverse})z]/2$   
Transformation Adjustment =  $\text{sum}(\ln(|(ci \cdot (ci \cdot yi)^{(\text{lam}-1))}|))$  where the sum is over the last nefobs observations  
 $AIC = -2 \cdot L + 2 \cdot np$   
 $AICC = -2 \cdot L + 2 \cdot np \cdot [\text{nefobs}/(\text{nefobs}-np-1)]$



$$\text{Hannan Quinn} = -2 * L + 2 * np * \log[\log(\text{nefobs})]$$
$$\text{BIC} = -2 * L + np * \log(\text{nefobs})$$

[Previous Table](#) | [Index](#) | [Next Table](#)

Roots of ARIMA Model				
	Real	Imaginary	Modulus	Frequency
Nonseasonal AR				
Root 1	0.699036	1.38588	1.55219	0.175649
Root 2	0.699036	-1.38588	1.55219	-0.175649
Seasonal MA				
Root 1	1.00073	0.00000	1.00073	0.00000

[Previous Table](#) | [Index](#) | [Next Table](#)

Regression Matrix

From 2013.Jan to 2024.Aug  
Observations 140

Regression Matrix				
	AO2013.Nov	LS2019.Dec	AO2020.Apr	LS2023.Jul
2013.Jan	0.00000	-1.00000	0.00000	-1.00000
2013.Feb	0.00000	-1.00000	0.00000	-1.00000
2013.Mar	0.00000	-1.00000	0.00000	-1.00000
2013.Apr	0.00000	-1.00000	0.00000	-1.00000
2013.May	0.00000	-1.00000	0.00000	-1.00000
2013.Jun	0.00000	-1.00000	0.00000	-1.00000
2013.Jul	0.00000	-1.00000	0.00000	-1.00000
2013.Aug	0.00000	-1.00000	0.00000	-1.00000
2013.Sep	0.00000	-1.00000	0.00000	-1.00000
2013.Oct	0.00000	-1.00000	0.00000	-1.00000
2013.Nov	1.00000	-1.00000	0.00000	-1.00000
2013.Dec	0.00000	-1.00000	0.00000	-1.00000
2014.Jan	0.00000	-1.00000	0.00000	-1.00000
2014.Feb	0.00000	-1.00000	0.00000	-1.00000
2014.Mar	0.00000	-1.00000	0.00000	-1.00000
2014.Apr	0.00000	-1.00000	0.00000	-1.00000
2014.May	0.00000	-1.00000	0.00000	-1.00000
2014.Jun	0.00000	-1.00000	0.00000	-1.00000
2014.Jul	0.00000	-1.00000	0.00000	-1.00000
2014.Aug	0.00000	-1.00000	0.00000	-1.00000
2014.Sep	0.00000	-1.00000	0.00000	-1.00000
2014.Oct	0.00000	-1.00000	0.00000	-1.00000
2014.Nov	0.00000	-1.00000	0.00000	-1.00000
2014.Dec	0.00000	-1.00000	0.00000	-1.00000
2015.Jan	0.00000	-1.00000	0.00000	-1.00000
2015.Feb	0.00000	-1.00000	0.00000	-1.00000
2015.Mar	0.00000	-1.00000	0.00000	-1.00000
2015.Apr	0.00000	-1.00000	0.00000	-1.00000
2015.May	0.00000	-1.00000	0.00000	-1.00000
2015.Jun	0.00000	-1.00000	0.00000	-1.00000
2015.Jul	0.00000	-1.00000	0.00000	-1.00000
2015.Aug	0.00000	-1.00000	0.00000	-1.00000
2015.Sep	0.00000	-1.00000	0.00000	-1.00000
2015.Oct	0.00000	-1.00000	0.00000	-1.00000
2015.Nov	0.00000	-1.00000	0.00000	-1.00000

2015.Dec	0.00000	-1.00000	0.00000	-1.00000
2016.Jan	0.00000	-1.00000	0.00000	-1.00000
2016.Feb	0.00000	-1.00000	0.00000	-1.00000
2016.Mar	0.00000	-1.00000	0.00000	-1.00000
2016.Apr	0.00000	-1.00000	0.00000	-1.00000
2016.May	0.00000	-1.00000	0.00000	-1.00000
2016.Jun	0.00000	-1.00000	0.00000	-1.00000
2016.Jul	0.00000	-1.00000	0.00000	-1.00000
2016.Aug	0.00000	-1.00000	0.00000	-1.00000
2016.Sep	0.00000	-1.00000	0.00000	-1.00000
2016.Oct	0.00000	-1.00000	0.00000	-1.00000
2016.Nov	0.00000	-1.00000	0.00000	-1.00000
2016.Dec	0.00000	-1.00000	0.00000	-1.00000
2017.Jan	0.00000	-1.00000	0.00000	-1.00000
2017.Feb	0.00000	-1.00000	0.00000	-1.00000
2017.Mar	0.00000	-1.00000	0.00000	-1.00000
2017.Apr	0.00000	-1.00000	0.00000	-1.00000
2017.May	0.00000	-1.00000	0.00000	-1.00000
2017.Jun	0.00000	-1.00000	0.00000	-1.00000
2017.Jul	0.00000	-1.00000	0.00000	-1.00000
2017.Aug	0.00000	-1.00000	0.00000	-1.00000
2017.Sep	0.00000	-1.00000	0.00000	-1.00000
2017.Oct	0.00000	-1.00000	0.00000	-1.00000
2017.Nov	0.00000	-1.00000	0.00000	-1.00000
2017.Dec	0.00000	-1.00000	0.00000	-1.00000
2018.Jan	0.00000	-1.00000	0.00000	-1.00000
2018.Feb	0.00000	-1.00000	0.00000	-1.00000
2018.Mar	0.00000	-1.00000	0.00000	-1.00000
2018.Apr	0.00000	-1.00000	0.00000	-1.00000
2018.May	0.00000	-1.00000	0.00000	-1.00000
2018.Jun	0.00000	-1.00000	0.00000	-1.00000
2018.Jul	0.00000	-1.00000	0.00000	-1.00000
2018.Aug	0.00000	-1.00000	0.00000	-1.00000
2018.Sep	0.00000	-1.00000	0.00000	-1.00000
2018.Oct	0.00000	-1.00000	0.00000	-1.00000
2018.Nov	0.00000	-1.00000	0.00000	-1.00000
2018.Dec	0.00000	-1.00000	0.00000	-1.00000
2019.Jan	0.00000	-1.00000	0.00000	-1.00000
2019.Feb	0.00000	-1.00000	0.00000	-1.00000
2019.Mar	0.00000	-1.00000	0.00000	-1.00000
2019.Apr	0.00000	-1.00000	0.00000	-1.00000
2019.May	0.00000	-1.00000	0.00000	-1.00000
2019.Jun	0.00000	-1.00000	0.00000	-1.00000
2019.Jul	0.00000	-1.00000	0.00000	-1.00000
2019.Aug	0.00000	-1.00000	0.00000	-1.00000
2019.Sep	0.00000	-1.00000	0.00000	-1.00000
2019.Oct	0.00000	-1.00000	0.00000	-1.00000
2019.Nov	0.00000	-1.00000	0.00000	-1.00000
2019.Dec	0.00000	0.00000	0.00000	-1.00000
2020.Jan	0.00000	0.00000	0.00000	-1.00000
2020.Feb	0.00000	0.00000	0.00000	-1.00000
2020.Mar	0.00000	0.00000	0.00000	-1.00000
2020.Apr	0.00000	0.00000	1.00000	-1.00000
2020.May	0.00000	0.00000	0.00000	-1.00000
2020.Jun	0.00000	0.00000	0.00000	-1.00000
2020.Jul	0.00000	0.00000	0.00000	-1.00000

2020.Aug	0.00000	0.00000	0.00000	-1.00000
2020.Sep	0.00000	0.00000	0.00000	-1.00000
2020.Oct	0.00000	0.00000	0.00000	-1.00000
2020.Nov	0.00000	0.00000	0.00000	-1.00000
2020.Dec	0.00000	0.00000	0.00000	-1.00000
2021.Jan	0.00000	0.00000	0.00000	-1.00000
2021.Feb	0.00000	0.00000	0.00000	-1.00000
2021.Mar	0.00000	0.00000	0.00000	-1.00000
2021.Apr	0.00000	0.00000	0.00000	-1.00000
2021.May	0.00000	0.00000	0.00000	-1.00000
2021.Jun	0.00000	0.00000	0.00000	-1.00000
2021.Jul	0.00000	0.00000	0.00000	-1.00000
2021.Aug	0.00000	0.00000	0.00000	-1.00000
2021.Sep	0.00000	0.00000	0.00000	-1.00000
2021.Oct	0.00000	0.00000	0.00000	-1.00000
2021.Nov	0.00000	0.00000	0.00000	-1.00000
2021.Dec	0.00000	0.00000	0.00000	-1.00000
2022.Jan	0.00000	0.00000	0.00000	-1.00000
2022.Feb	0.00000	0.00000	0.00000	-1.00000
2022.Mar	0.00000	0.00000	0.00000	-1.00000
2022.Apr	0.00000	0.00000	0.00000	-1.00000
2022.May	0.00000	0.00000	0.00000	-1.00000
2022.Jun	0.00000	0.00000	0.00000	-1.00000
2022.Jul	0.00000	0.00000	0.00000	-1.00000
2022.Aug	0.00000	0.00000	0.00000	-1.00000
2022.Sep	0.00000	0.00000	0.00000	-1.00000
2022.Oct	0.00000	0.00000	0.00000	-1.00000
2022.Nov	0.00000	0.00000	0.00000	-1.00000
2022.Dec	0.00000	0.00000	0.00000	-1.00000
2023.Jan	0.00000	0.00000	0.00000	-1.00000
2023.Feb	0.00000	0.00000	0.00000	-1.00000
2023.Mar	0.00000	0.00000	0.00000	-1.00000
2023.Apr	0.00000	0.00000	0.00000	-1.00000
2023.May	0.00000	0.00000	0.00000	-1.00000
2023.Jun	0.00000	0.00000	0.00000	-1.00000
2023.Jul	0.00000	0.00000	0.00000	0.00000
2023.Aug	0.00000	0.00000	0.00000	0.00000
2023.Sep	0.00000	0.00000	0.00000	0.00000
2023.Oct	0.00000	0.00000	0.00000	0.00000
2023.Nov	0.00000	0.00000	0.00000	0.00000
2023.Dec	0.00000	0.00000	0.00000	0.00000
2024.Jan	0.00000	0.00000	0.00000	0.00000
2024.Feb	0.00000	0.00000	0.00000	0.00000
2024.Mar	0.00000	0.00000	0.00000	0.00000
2024.Apr	0.00000	0.00000	0.00000	0.00000
2024.May	0.00000	0.00000	0.00000	0.00000
2024.Jun	0.00000	0.00000	0.00000	0.00000
2024.Jul	0.00000	0.00000	0.00000	0.00000
2024.Aug	0.00000	0.00000	0.00000	0.00000

[Previous Table](#) | [Index](#) | [Next Table](#)

## Model Residuals

From 2013.Feb to 2024.Aug  
Observations 139

## Model

Model												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013		-0.01261667	0.00572145	0.00636211	0.00305530	0.01005357	-0.00000507	0.01371999	0.01080782	0.00283398	0.00280707	-0.006184
2014	-0.01056367	0.00497611	0.00048046	0.00080979	-0.00005604	-0.00167524	0.02308319	-0.00306938	-0.00507717	-0.00664668	-0.00226823	-0.001359
2015	0.00140304	0.00366869	-0.00192147	-0.00196187	-0.00030323	0.00248416	-0.01394101	0.01861635	-0.00229821	-0.00283075	0.00465264	0.002748
2016	0.00694575	-0.01203457	0.00746512	0.00370205	0.00281800	0.00432256	-0.00629095	-0.00600148	-0.00547053	-0.00766472	-0.01207288	0.000998
2017	-0.00204323	0.00348900	-0.00369795	-0.00440571	-0.00574035	-0.00267706	0.01092443	-0.00212100	-0.01166761	0.00813899	0.00682464	-0.004307
2018	0.00198945	-0.00650439	-0.00111051	-0.00764080	-0.00466104	-0.00619704	-0.00256640	-0.00382712	-0.01014009	-0.01102610	-0.00216882	0.000022
2019	-0.00108121	0.00628352	0.00203573	0.00077460	0.00212700	-0.00294793	-0.00088399	0.00341816	0.00712466	0.00577121	0.01070179	0.005048
2020	-0.00111984	-0.01371732	-0.00409168	0.00328847	-0.00482380	-0.00775573	0.00769212	-0.00345390	0.02527195	-0.00085706	0.00160071	-0.011341
2021	-0.00618371	0.00260327	-0.01124478	0.00210656	0.00576559	-0.01049224	-0.00258359	-0.00448576	0.00149249	0.01117819	0.00085025	0.004671
2022	-0.00103550	0.00765278	0.00906656	-0.00030033	0.00626080	-0.00821595	-0.01260303	0.00908742	0.00145357	-0.00529689	-0.00888608	0.006728
2023	0.00974274	-0.00476025	0.00548532	-0.00327086	-0.00144537	0.00813687	-0.00218063	-0.01198497	-0.01149229	0.00645590	-0.00206370	0.003026
2024	0.00199058	0.00836740	-0.00246564	0.00051187	-0.00300011	0.01495699	-0.00072374	-0.01004348				

[Previous Table](#) | [Index](#) | [Next Table](#)

## QS Statistic for regARIMA Model Residuals

QS Statistic for regARIMA Model Residuals (full series): 0.00 (P-Value = 1.0000)

QS Statistic for regARIMA Model Residuals (starting 2016.Sep): 0.00 (P-Value = 1.0000)

[Previous Table](#) | [Index](#) | [Next Table](#)

## 10\*LOG(SPECTRUM) of the regARIMA model residuals

Spectrum estimated from 2016.Sep to 2024.Aug.

```

+++++I+++++I
-38.35I      *      I      -38.35
      I      *      I
      I      *      I
      I      **     I
-39.29I      **     I      -39.29
      I      **     I
      I      **     I
      I      **     I
-40.23I      *      *      *      *      *      *      I      -40.23
      I      **   *      *      *      *      *      I
      I      **   *      *      *      *      *      I
      I      **   *      *      *      *      *      I
-41.17I      **   *      *      *      *      *      I      -41.17
      I      *   **   *      *      *      *      I
      I      *   **   **   S*   **   **   **   **   I
      I      *   **   **   S**  ***   **   **   **   I
-42.10I      *   **   **   S**  ***   **   **   **   I      -42.10
      I      *   **   **   S** **  ***   **   **   **   I
      I      *   **   **   S*****   **   **   **   I
      I      *   **   **   S*****   *   **   ***   I
-43.04I      *****   **   S*****   *   **   ***   I      -43.04
      I      *****   **   S*****   **   **   ****   I
      I*      *****   **   S*****   **   ***   ****   I
      I*      *****S   ***   S*****   **   ***   ****   *   I
-43.98I*      *****S   ***   S*****   *****   ****   *   SI      -43.98
      I*      *****S   ***   S*****   *****   ****   **   *SI
      I*      *****S   ***   S*****   *****   ****   ****   *SI
      I*      *****S   ***   S*****   *****   ****   ****   *SI
-44.92I*      *****S*****   S*****   *****   ****   ****   *SI      -44.92
      I*      *****S*****   S*****   *****   ****   ****   *SI

```

[Previous Table](#) | [Index](#) | [Next Table](#)

From 2013.Jan to 2024.Aug  
Observations 140

35/42

2015.Aug	0.00000	-0.734234E-01	-0.734234E-01	4.94326
2015.Sep	0.00000	-0.734234E-01	-0.734234E-01	4.95091
2015.Oct	0.00000	-0.734234E-01	-0.734234E-01	4.95925
2015.Nov	0.00000	-0.734234E-01	-0.734234E-01	4.96302
2015.Dec	0.00000	-0.734234E-01	-0.734234E-01	4.95091
2016.Jan	0.00000	-0.734234E-01	-0.734234E-01	4.94938
2016.Feb	0.00000	-0.734234E-01	-0.734234E-01	4.93479
2016.Mar	0.00000	-0.734234E-01	-0.734234E-01	4.93479
2016.Apr	0.00000	-0.734234E-01	-0.734234E-01	4.95091
2016.May	0.00000	-0.734234E-01	-0.734234E-01	4.96977
2016.Jun	0.00000	-0.734234E-01	-0.734234E-01	4.99340
2016.Jul	0.00000	-0.734234E-01	-0.734234E-01	5.00646
2016.Aug	0.00000	-0.734234E-01	-0.734234E-01	5.00068
2016.Sep	0.00000	-0.734234E-01	-0.734234E-01	4.98975
2016.Oct	0.00000	-0.734234E-01	-0.734234E-01	4.99194
2016.Nov	0.00000	-0.734234E-01	-0.734234E-01	4.98313
2016.Dec	0.00000	-0.734234E-01	-0.734234E-01	4.96452
2017.Jan	0.00000	-0.734234E-01	-0.734234E-01	4.95547
2017.Feb	0.00000	-0.734234E-01	-0.734234E-01	4.95471
2017.Mar	0.00000	-0.734234E-01	-0.734234E-01	4.95471
2017.Apr	0.00000	-0.734234E-01	-0.734234E-01	4.95698
2017.May	0.00000	-0.734234E-01	-0.734234E-01	4.95925
2017.Jun	0.00000	-0.734234E-01	-0.734234E-01	4.97201
2017.Jul	0.00000	-0.734234E-01	-0.734234E-01	5.00285
2017.Aug	0.00000	-0.734234E-01	-0.734234E-01	5.01578
2017.Sep	0.00000	-0.734234E-01	-0.734234E-01	5.00213
2017.Oct	0.00000	-0.734234E-01	-0.734234E-01	5.01077
2017.Nov	0.00000	-0.734234E-01	-0.734234E-01	5.02572
2017.Dec	0.00000	-0.734234E-01	-0.734234E-01	5.01292
2018.Jan	0.00000	-0.734234E-01	-0.734234E-01	5.00140
2018.Feb	0.00000	-0.734234E-01	-0.734234E-01	4.98681
2018.Mar	0.00000	-0.734234E-01	-0.734234E-01	4.98240
2018.Apr	0.00000	-0.734234E-01	-0.734234E-01	4.98461
2018.May	0.00000	-0.734234E-01	-0.734234E-01	4.98975
2018.Jun	0.00000	-0.734234E-01	-0.734234E-01	5.00068
2018.Jul	0.00000	-0.734234E-01	-0.734234E-01	5.01578
2018.Aug	0.00000	-0.734234E-01	-0.734234E-01	5.01863
2018.Sep	0.00000	-0.734234E-01	-0.734234E-01	5.00718
2018.Oct	0.00000	-0.734234E-01	-0.734234E-01	5.00213
2018.Nov	0.00000	-0.734234E-01	-0.734234E-01	4.99923
2018.Dec	0.00000	-0.734234E-01	-0.734234E-01	4.98608
2019.Jan	0.00000	-0.734234E-01	-0.734234E-01	4.97870
2019.Feb	0.00000	-0.734234E-01	-0.734234E-01	4.97944
2019.Mar	0.00000	-0.734234E-01	-0.734234E-01	4.98534
2019.Apr	0.00000	-0.734234E-01	-0.734234E-01	4.99559
2019.May	0.00000	-0.734234E-01	-0.734234E-01	5.00790
2019.Jun	0.00000	-0.734234E-01	-0.734234E-01	5.02289
2019.Jul	0.00000	-0.734234E-01	-0.734234E-01	5.03906
2019.Aug	0.00000	-0.734234E-01	-0.734234E-01	5.04809
2019.Sep	0.00000	-0.734234E-01	-0.734234E-01	5.05703
2019.Oct	0.00000	-0.734234E-01	-0.734234E-01	5.07804
2019.Nov	0.00000	-0.734234E-01	-0.734234E-01	5.09467
2019.Dec	0.00000	-0.417011E-01	-0.417011E-01	5.08706
2020.Jan	0.00000	-0.417011E-01	-0.417011E-01	5.07475
2020.Feb	0.00000	-0.417011E-01	-0.417011E-01	5.05033
2020.Mar	0.00000	-0.417011E-01	-0.417011E-01	5.03756

2020.Apr	0.248223E-01	-0.417011E-01	-0.168788E-01	5.04993
2020.May	0.00000	-0.417011E-01	-0.417011E-01	5.06426
2020.Jun	0.00000	-0.417011E-01	-0.417011E-01	5.07475
2020.Jul	0.00000	-0.417011E-01	-0.417011E-01	5.09603
2020.Aug	0.00000	-0.417011E-01	-0.417011E-01	5.10303
2020.Sep	0.00000	-0.417011E-01	-0.417011E-01	5.12683
2020.Oct	0.00000	-0.417011E-01	-0.417011E-01	5.15067
2020.Nov	0.00000	-0.417011E-01	-0.417011E-01	5.15369
2020.Dec	0.00000	-0.417011E-01	-0.417011E-01	5.12062
2021.Jan	0.00000	-0.417011E-01	-0.417011E-01	5.09412
2021.Feb	0.00000	-0.417011E-01	-0.417011E-01	5.08835
2021.Mar	0.00000	-0.417011E-01	-0.417011E-01	5.08513
2021.Apr	0.00000	-0.417011E-01	-0.417011E-01	5.09412
2021.May	0.00000	-0.417011E-01	-0.417011E-01	5.11312
2021.Jun	0.00000	-0.417011E-01	-0.417011E-01	5.12497
2021.Jul	0.00000	-0.417011E-01	-0.417011E-01	5.13484
2021.Aug	0.00000	-0.417011E-01	-0.417011E-01	5.13361
2021.Sep	0.00000	-0.417011E-01	-0.417011E-01	5.13361
2021.Oct	0.00000	-0.417011E-01	-0.417011E-01	5.15910
2021.Nov	0.00000	-0.417011E-01	-0.417011E-01	5.17219
2021.Dec	0.00000	-0.417011E-01	-0.417011E-01	5.16029
2022.Jan	0.00000	-0.417011E-01	-0.417011E-01	5.14704
2022.Feb	0.00000	-0.417011E-01	-0.417011E-01	5.14522
2022.Mar	0.00000	-0.417011E-01	-0.417011E-01	5.15910
2022.Apr	0.00000	-0.417011E-01	-0.417011E-01	5.17396
2022.May	0.00000	-0.417011E-01	-0.417011E-01	5.18978
2022.Jun	0.00000	-0.417011E-01	-0.417011E-01	5.19961
2022.Jul	0.00000	-0.417011E-01	-0.417011E-01	5.19961
2022.Aug	0.00000	-0.417011E-01	-0.417011E-01	5.20706
2022.Sep	0.00000	-0.417011E-01	-0.417011E-01	5.21615
2022.Oct	0.00000	-0.417011E-01	-0.417011E-01	5.22685
2022.Nov	0.00000	-0.417011E-01	-0.417011E-01	5.21785
2022.Dec	0.00000	-0.417011E-01	-0.417011E-01	5.20133
2023.Jan	0.00000	-0.417011E-01	-0.417011E-01	5.20534
2023.Feb	0.00000	-0.417011E-01	-0.417011E-01	5.20305
2023.Mar	0.00000	-0.417011E-01	-0.417011E-01	5.20592
2023.Apr	0.00000	-0.417011E-01	-0.417011E-01	5.21162
2023.May	0.00000	-0.417011E-01	-0.417011E-01	5.21898
2023.Jun	0.00000	-0.417011E-01	-0.417011E-01	5.24406
2023.Jul	0.00000	0.00000	0.00000	5.26683
2023.Aug	0.00000	0.00000	0.00000	5.26010
2023.Sep	0.00000	0.00000	0.00000	5.23857
2023.Oct	0.00000	0.00000	0.00000	5.24913
2023.Nov	0.00000	0.00000	0.00000	5.25958
2023.Dec	0.00000	0.00000	0.00000	5.25070
2024.Jan	0.00000	0.00000	0.00000	5.24333
2024.Feb	0.00000	0.00000	0.00000	5.24439
2024.Mar	0.00000	0.00000	0.00000	5.24597
2024.Apr	0.00000	0.00000	0.00000	5.25332
2024.May	0.00000	0.00000	0.00000	5.26062
2024.Jun	0.00000	0.00000	0.00000	5.29180
2024.Jul	0.00000	0.00000	0.00000	5.31959
2024.Aug	0.00000	0.00000	0.00000	5.31517

[Previous Table](#) | [Index](#) | [Next Table](#)

Residuals from the Estimated Regression Effects

From 2013.Jan to 2024.Aug  
Observations 140

Data

Data												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2013	4.731	4.741	4.742	4.751	4.766	4.795	4.820	4.838	4.853	4.870	4.874	4.850
2014	4.824	4.817	4.824	4.839	4.851	4.865	4.905	4.922	4.913	4.908	4.904	4.889
2015	4.883	4.883	4.884	4.889	4.898	4.918	4.926	4.943	4.951	4.959	4.963	4.951
2016	4.949	4.935	4.935	4.951	4.970	4.993	5.006	5.001	4.990	4.992	4.983	4.965
2017	4.955	4.955	4.955	4.957	4.959	4.972	5.003	5.016	5.002	5.011	5.026	5.013
2018	5.001	4.987	4.982	4.985	4.990	5.001	5.016	5.019	5.007	5.002	4.999	4.986
2019	4.979	4.979	4.985	4.996	5.008	5.023	5.039	5.048	5.057	5.078	5.095	5.087
2020	5.075	5.050	5.038	5.050	5.064	5.075	5.096	5.103	5.127	5.151	5.154	5.121
2021	5.094	5.088	5.085	5.094	5.113	5.125	5.135	5.134	5.134	5.159	5.172	5.160
2022	5.147	5.145	5.159	5.174	5.190	5.200	5.200	5.207	5.216	5.227	5.218	5.201
2023	5.205	5.203	5.206	5.212	5.219	5.244	5.267	5.260	5.239	5.249	5.260	5.251
2024	5.243	5.244	5.246	5.253	5.261	5.292	5.320	5.315				

[Previous Table](#) | [Index](#) | [Next Table](#)

A 8 RegARIMA combined outlier component  
(AO, LS outliers included)

From 2013.Jan to 2024.Aug  
Observations 140

A 8 RegARIMA combined outlier component													
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	AVGE
2013	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	94.6	92.9	93.1
2014	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2015	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2016	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2017	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2018	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2019	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	95.9	93.2
2020	95.9	95.9	95.9	98.3	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	96.1
2021	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9
2022	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9
2023	95.9	95.9	95.9	95.9	95.9	95.9	100.0	100.0	100.0	100.0	100.0	100.0	98.0
2024	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0					100.0
AVGE	94.5	94.5	94.5	94.7	94.5	94.5	94.8	94.8	94.4	94.4	94.5	94.7	

Table Total- 13241.    Mean- 95.    Standard Deviation- 6.  
Minimum- 93.    Maximum- 100.

A 8.A RegARIMA outlier component

From 2024.Sep to 2025.Aug  
Observations 12

A 8.A RegARIMA outlier component												
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2024									100.0	100.0	100.0	100.0



2025	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0					100.0
------	-------	-------	-------	-------	-------	-------	-------	-------	--	--	--	--	-------

[Previous Table](#) | [Index](#) | [Next Table](#)

A 8.AO RegARIMA AO outlier component

From 2013.Jan to 2024.Aug  
Observations 140

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	AVGE
2013	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	101.9	100.0	100.2
2014	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2015	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2016	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2017	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2018	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2019	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2020	100.0	100.0	100.0	102.5	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.2
2021	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2022	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2023	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
2024	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0					100.0
AVGE	100.0	100.0	100.0	100.2	100.0	100.0	100.0	100.0	100.0	100.0	100.2	100.0	

Table Total- 14004.    Mean- 100.    Standard Deviation- 0.  
Minimum- 100.    Maximum- 103.

[Previous Table](#) | [Index](#) | [Next Table](#)

A 8.LS RegARIMA level change outlier component

From 2013.Jan to 2024.Aug  
Observations 140

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	AVGE
2013	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2014	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2015	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2016	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2017	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2018	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9
2019	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	92.9	95.9	93.2
2020	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9
2021	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9
2022	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9	95.9
2023	95.9	95.9	95.9	95.9	95.9	95.9	100.0	100.0	100.0	100.0	100.0	100.0	98.0
2024	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0					100.0
AVGE	94.5	94.5	94.5	94.5	94.5	94.5	94.8	94.8	94.4	94.4	94.4	94.7	

Table Total- 13237.    Mean- 95.    Standard Deviation- 6.  
Minimum- 93.    Maximum- 100.

[Previous Table](#) | [Index](#) | [Next Table](#)



```

I      S      I
I      S      I
I      S      I
-25.34I      S      I      -25.34
I      S      I
I      S      I
I      S      I
-27.55I      S      I      -27.55
I      S      I
I      S      I
I      S      I
-29.76I      S      I      -29.76
I      S      I
I      S      I
I      S      I
-31.97I      S      S      I      -31.97
I      S      S      I
I      S      S      I
I      S      S      I
-34.17I      S*      S*      S      I      -34.17
I      S*      S*      S
I      *S*      S*      S
I      *S*      S*      S      S
-36.38I      *S*      *S*      *S      S      I      -36.38
I      *S*      *S*      *S      S
I      *S***      *S**      **S*      S*
I      *      *S****      *S**      **S*      S*
-38.59I      *      *S****      *S**      ***S*      S*      -38.59
I      ****S****      **S*****S*      S*
I      ****S****      **S*****S*      S*
I      ****S****      **S*****S*      S*
-40.80I      ****S****      **S*****S*      S*      -40.80
I      ****S****      **S*****S*      S*
I*      ****S****      ***S*****S**      *S*      *
I**      ****S****      ****S*****S**      ***S*      *
-43.01I*****S*****S*****S*****S*      *      I      -43.01
I*****S*****S*****S*****S*      **
I*****S*****S*****S*****S*T      **
I*****S*****S*****S*****S*T      **
-45.21I*****S*****S*****S*****S*T      **      I      -45.21
I*****S*****S*****S*****S*T      **
I*****S*****S*****S*****S*T      ***
I*****S*****S*****S*****S*T*      ****
-47.42I*****S*****S*****S*****S*T*      ****      I      -47.42
I*****S*****S*****S*****S*T*****
I*****S*****S*****S*****S*T*****
I*****S*****S*****S*****S*T*****
-49.63I*****S*****S*****S*****S*T*****      *      I
I*****S*****S*****S*****S*T*****      **      *SI      -49.63
I*****S*****S*****S*****S*T*****      ***      **SI
I*****S*****S*****S*****S*T*****S      ***      **SI
-51.84I*****S*****S*****S*****S*T*****S*T*****SI      -51.84
+++++++I+++++++I
S=SEASONAL FREQUENCIES, T=TRADING DAY FREQUENCIES

```

[Previous Table](#) | [Index](#) | [Next Table](#)

## Peak probabilities for Tukey spectrum estimator

Tukey spectrum estimated from 2016.Sep to 2024.Aug.

Peak probabilities for Tukey spectrum estimator							
	S1	S2	S3	S4	S5	S6	TD
Model Residuals	0.798	0.107	0.592	0.000	0.336	0.772	0.107
Prior Adjusted Series (Table B1)	0.969 *	0.780	0.980 *	0.857	0.062	0.830	0.057

-----  
\*\* - Peak Probability > 0.99,

\* - 0.90 < Peak Probability < 0.99



## A.2 R Implementation

# Implementation of regARIMA Procedure Using R

Rohit Jangid

2024-11-01

## Contents

<b>Introduction</b>	<b>1</b>
<b>Loading Required Libraries</b>	<b>2</b>
<b>Helper Functions</b>	<b>2</b>
Creating Regressor for Trading Days . . . . .	2
Selection Criteria for Model Evaluation . . . . .	2
Plotting Model Output . . . . .	3
Generating Constant Term . . . . .	5
Outlier Detection Functions . . . . .	5
Calculating Outlier T-Value . . . . .	6
Forward and Backward Pass for Outlier Detection . . . . .	6
<b>Data Preparation</b>	<b>8</b>
Reading Diwali Dates . . . . .	8
Calculating Diwali Regressor . . . . .	8
Creating Trading Day Regressor . . . . .	9
Preparing Time Series Objects . . . . .	9
<b>Model Fitting and Transformation</b>	<b>11</b>
<b>Default Model Estimation</b>	<b>13</b>
<b>Trading Day Regressor Selection</b>	<b>15</b>
<b>Outlier Detection in Default Model</b>	<b>19</b>
<b>Next Steps</b>	<b>21</b>
<b>Conclusion</b>	<b>21</b>
<b>References</b>	<b>21</b>

## Introduction

This document presents an implementation of the **regARIMA** procedure from the X13-ARIMA-SEATS software using R. The **regARIMA** procedure is utilized for seasonal adjustment and modeling of time series data. The methodology involves several steps, including data preparation, model fitting, outlier detection, and model selection based on information criteria.

## Loading Required Libraries

We begin by loading the necessary R libraries for data manipulation, visualization, and time series analysis.

```
library(ggplot2)
library(gridExtra)
library(forecast)
library(tseries)
library(seasonal)
```

## Helper Functions

The implementation relies on several helper functions to create regressors, evaluate model selection criteria, plot model outputs, and detect outliers. Below, we define these functions with detailed explanations.

### Creating Regressor for Trading Days

The `td1nolpyear_regressor` function generates a regressor based on the number of trading days and weekends for each month in the time series.

```
td1nolpyear_regressor <- function(start = c(2023, 1), n) {
  # Initialize an empty vector to store the regressor values
  regressor <- numeric(n)

  # Loop through each month from the start date for n months
  for (i in 0:(n - 1)) {
    # Calculate the year and month for the current iteration
    year <- start[1] + (start[2] + i - 1) %/% 12
    month <- (start[2] + i - 1) %% 12 + 1

    # Start and end dates for the current month
    start_date <- as.Date(paste(year, sprintf("%02d", month), "01", sep = "-"))
    end_date <- seq(start_date, by = "month", length.out = 2)[2] - 1

    # Get all dates in the current month
    dates <- seq(start_date, end_date, by = "day")

    # Count weekdays and weekends (Saturday and Sunday)
    weekdays_count <- sum(!weekdays(dates) %in% c("Saturday", "Sunday"))
    weekends_count <- sum(weekdays(dates) %in% c("Saturday", "Sunday"))

    # Calculate the regressor for the month
    regressor[i + 1] <- weekdays_count - (5 / 2) * weekends_count
  }

  return(regressor)
}
```

### Selection Criteria for Model Evaluation

The `selection_criteria` function calculates various information criteria (AIC, BIC, AICc, HQ) to evaluate and compare different ARIMA models.

```
selection_criteria <- function(model, adjust_for_log_transform = FALSE) {
  # Extract log-likelihood from the model
```

```

loglik <- as.numeric(logLik(model))

# Extract original series and frequency from model
original_series <- model$x
freq <- frequency(model$x)

# Extract non-seasonal and seasonal orders from model$call
arima_order <- eval(model$call$order)
seasonal_order <- eval(model$call$seasonal)

# Determine the number of regressors
number_of_regressors <- dim(eval(model$call$xreg))[2]
if (is.null(number_of_regressors)) {
  number_of_regressors <- 1
}

# Non-seasonal and seasonal differencing
order_diff <- arima_order[2]
seasonal_diff <- seasonal_order[2]

# Calculate the effective number of observations
effective_n <- length(original_series) - order_diff - freq * seasonal_diff

if (adjust_for_log_transform) {
  loglik <- loglik - sum(original_series[(length(original_series) -
                                          effective_n + 1):length(original_series)])
}

# Calculate the number of parameters (including the variance term)
n_p <- number_of_regressors + arima_order[1] + arima_order[3] +
  seasonal_order[1] + seasonal_order[3] + 1

# Calculate AIC
aic <- -2 * loglik + 2 * n_p

# Calculate BIC
bic <- -2 * loglik + n_p * log(effective_n)

# Calculate AICc (corrected AIC)
aicc <- aic + (2 * n_p * (n_p + 1)) / (effective_n - n_p - 1)

# Calculate Hannan-Quinn Criterion
hq <- -2 * loglik + 2 * n_p * log(log(effective_n))

# Return the results as a list
return(list(AIC = aic, BIC = bic, AICc = aicc, HQ = hq, npar = n_p))
}

```

## Plotting Model Output

The `plot_model_output` function visualizes the original time series, fitted values, residuals, and provides a summary of the model fit, including information criteria and Ljung-Box test results.



```

plot_model_output <- function(model, series_name = "Series",
                              adj_inf_criteria_for_log = FALSE) {
  # Extract the original series, fitted values, and residuals from the model
  original_series <- model$x
  fitted_values <- fitted(model)
  residuals <- residuals(model)

  # Perform Ljung-Box test for residuals independence
  ljung_box_test <- Box.test(residuals, lag = 20, type = "Ljung-Box")
  lb_pvalue <- round(ljung_box_test$p.value, 4)

  # Extract model coefficients and standard errors
  coef_vals <- round(coef(model), 4)
  coef_errors <- round(sqrt(diag(vcov(model))), 4)

  # Create the model equation as a string
  model_eq <- paste0("ARIMA", substr(paste(model$call)[3], 2, 10),
                    " x ",
                    substr(paste(model$call)[4], 2, 10),
                    " Method = ", paste(model$call$method))

  # Use the selection_criteria function to get AIC, BIC, AICc, and HQ
  criteria <- selection_criteria(model,
                                adjust_for_log_transform = adj_inf_criteria_for_log)

  n_p <- criteria$npar

  # Create a dataframe to store the original series, fitted values, and residuals
  df <- data.frame(
    Time = as.numeric(time(original_series)),
    Original = as.numeric(original_series),
    Fitted = as.numeric(fitted_values),
    Residuals = as.numeric(residuals)
  )

  # Plot the original series and fitted values
  p1 <- ggplot(df, aes(x = Time)) +
    geom_line(aes(y = Original), color = "blue", linewidth = 1,
              show.legend = TRUE) +
    geom_line(aes(y = Fitted), color = "red",
              linetype = "dashed", linewidth = 1, show.legend = TRUE) +
    ggtitle(paste(series_name, ": Original vs Fitted")) +
    xlab("Time") + ylab("Values") +
    theme_minimal() +
    labs(caption = paste("Model Summary: AIC =", round(criteria$AIC, 2),
                        ", npar=", n_p,
                        ", BIC =", round(criteria$BIC, 2),
                        ", AICc =", round(criteria$AICc, 2),
                        ", HQ =", round(criteria$HQ, 2),
                        ", Log-Likelihood =", round(logLik(model), 2),
                        "\nLjung-Box Test p-value:", lb_pvalue,
                        "\nModel Equation:", model_eq,
                        "\nCoefficients (Estimate ± Std. Error):\n",

```

```

        paste(names(coef_vals), "=", coef_vals, "±", coef_errors,
              collapse = ", "))

# Plot the residuals
p2 <- ggplot(df, aes(x = Time, y = Residuals)) +
  geom_line(color = "purple", linewidth = 1) +
  geom_hline(yintercept = 0, linetype = "dashed", color = "black") +
  ggtitle(paste(series_name, ": Residuals")) +
  xlab("Time") + ylab("Residuals") +
  theme_minimal()

# Print the model summary and Ljung-Box test result on the console
print(summary(model))
print(ljung_box_test)

# Combine the plots
grid.arrange(p1, p2, nrow = 2)
}

```

## Generating Constant Term

The `const_term` function creates a constant term for the model based on differencing orders.

```

const_term <- function(y, d, D){
  return(diffinv(diffinv(rep(1, length(y)), lag = 1, differences =d), lag = 12,
    differences = D)[-1:(d+D*frequency(y))])
}

```

## Outlier Detection Functions

Functions AO (Additive Outlier) and LS (Level Shift) are defined to detect different types of outliers in the time series.

```

AO <- function(y, t) {
  time_point = start(y)
  time_point[2] = time_point[2] + (t-1)
  time_point[1] = time_point[1] + ((time_point[2]-1) %/% 12)
  time_point[2] = ifelse( time_point[2] %/% 12 == 0 , 12, time_point[2]%12)
  new_var_name = paste("AO", time_point[1], ".", sprintf("%02d", time_point[2]), sep = "")

  x = as.integer(seq_along(y) == t)
  x = matrix(x, ncol = 1)
  colnames(x)[1] <- new_var_name
  return(x)
}

LS <- function(y, t) {
  time_point = start(y)
  time_point[2] = time_point[2] + (t-1)
  time_point[1] = time_point[1] + ((time_point[2]-1) %/% 12)
  time_point[2] = ifelse( time_point[2] %/% 12 == 0 , 12, time_point[2]%12)
  new_var_name = paste("LS", time_point[1], ".", sprintf("%02d", time_point[2]), sep = "")

  x = as.integer(seq_along(y) >= t) - 1
}

```

```

x = matrix(x, ncol = 1)
colnames(x)[1] <- new_var_name
return(x)
}

```

## Calculating Outlier T-Value

The `outlier_t_value` function computes the t-value for potential outliers to assess their significance in the model.

```

outlier_t_value <- function(model, t, xreg = NULL, type = c("AO", "LS")){
  y = model$x
  X = xreg
  nreg = ifelse(is.null(xreg), 0, dim(xreg)[2])
  ncoeff = sum(c(eval(model$call$order)[-2], eval(model$call$seasonal)[-2]))
  model_order = eval(model$call$order)
  model_seasonal = eval(model$call$seasonal)
  new_variable = NULL
  if(type == 'AO'){
    new_variable = AO(y, t)
  }
  else if(type == "LS"){
    new_variable = LS(y, t)
  }
  X = cbind(X, new_variable)
  if(abs(det(t(X) %*% X)) < 1e-10){
    return(list(tvalue = 0, time_point = colnames(X)[dim(X)[2]] ))
  }
  fit = Arima(y, order = model_order,
              seasonal = model_seasonal,
              xreg = X,
              include.mean = FALSE, method = 'ML',
              fixed = c(coef(model)[1:ncoeff], rep(NA, nreg + 1)))
  ind = length(fit$coef)
  ind2 = dim(fit$var.coef)[1]
  return(list(tvalue = abs((fit$coef)[ind]) / sqrt((fit$var.coef)[ind2, ind2]),
             time_point = colnames(X)[dim(X)[2]]))
}

```

## Forward and Backward Pass for Outlier Detection

The `forward_pass` and `backward_pass` functions iteratively identify and remove outliers based on t-values exceeding a critical threshold.

```

forward_pass <- function(model, xreg=NULL, types = c("AO", "LS"), tcritical=3.88){
  y = model$x
  n = length(y)

  max_tvalue = 0
  ind_max_tvalue = NULL
  type_max_tvalue = NULL
  for(outliertype in types){
    for(i in 1:n){
      foo = outlier_t_value(model, i, xreg, outliertype)
    }
  }
}

```

```

        tvalue = foo$tvalue
        time_point = foo$time_point
        if(tvalue > tcritical){
            print(paste(time_point, tvalue))
            if(tvalue > max_tvalue){
                max_tvalue = tvalue
                ind_max_tvalue = i
                type_max_tvalue = outliertype
            }
        }
    }
}
if(is.null(type_max_tvalue)){
    return(NULL)
}
if(type_max_tvalue == "AO"){
    return(AO(y, ind_max_tvalue))
}
if(type_max_tvalue == "LS"){
    return(LS(y, ind_max_tvalue))
}
return(NULL)
}

backward_pass <- function(model, xreg=NULL, tcritical=3.88){
    y = model$x
    X = xreg
    nreg = ifelse(is.null(xreg), 0, dim(xreg)[2])
    ncoeff = sum(c(eval(model$call$order)[-2], eval(model$call$seasonal)[-2]))
    model_order = eval(model$call$order)
    model_seasonal = eval(model$call$seasonal)

    fit = Arima(y, order = model_order,
                seasonal = model_seasonal,
                xreg = X,
                include.mean = FALSE, method = 'ML',
                fixed = c(coef(model)[1:ncoeff], rep(NA, nreg)))
    betas = abs(fit$coef[-c(1:ncoeff)])
    se_betas = sqrt(diag(fit$var.coef))
    tvalues = betas / se_betas

    min_tvalue = 1e8
    min_ind = NULL
    for(i in 1:length(tvalues)){
        if(grepl("(A0|LS)(1[6-9][0-9]|20[0-9][0-9])\\. (0[1-9]|1[0-2])$",
                colnames(X)[i])){
            if(tvalues[i] < tcritical ){
                print(paste(colnames(X)[i], tvalues[i]))
                if(tvalues[i] < min_tvalue ){
                    min_tvalue = tvalues[i]
                    min_ind = i
                }
            }
        }
    }
}

```

```

    }
  }

  return(min_ind)
}

```

## Data Preparation

### Reading Diwali Dates

We start by reading the Diwali dates from a CSV file. This file was generated using the data vendor : Calendarific The `ddates` dataframe is expected to contain the columns: `year`, `month`, and `day` for each Diwali date.

```

setwd("~/Documents/Projects/X-13-ARIMA-SEATS-Model-for-Seasonal-Adjustment")
ddates <- read.csv('data/diwali.csv')

```

### Calculating Diwali Regressor

The `diwali_w` function calculates the proportion of days falling in September, October, and November within a window of `w` days before Diwali.

```

diwali_w <- function(year, w) {
  # Find Diwali month and day for the given year
  m <- ddates$month[which(ddates$year == year)]
  d <- ddates$day[which(ddates$year == year)]

  # Create a Date object for Diwali
  diwali_date <- as.Date(paste(year, m, d, sep = "-"))

  # Calculate the date 'w' days before Diwali
  start_date <- diwali_date - w

  # Initialize counts for each month
  sep_count <- 0
  oct_count <- 0
  nov_count <- 0

  # Iterate over each day in the range
  for (i in 1:w) {
    current_date <- diwali_date - i + 1
    current_month <- format(current_date, "%m")

    # Count days in each month
    if (current_month == "09") {
      sep_count <- sep_count + 1
    } else if (current_month == "10") {
      oct_count <- oct_count + 1
    } else if (current_month == "11") {
      nov_count <- nov_count + 1
    }
  }

  total_days <- w
}

```

```

    return(c("sep" = sep_count / total_days,
            "oct" = oct_count / total_days, "nov" = nov_count / total_days))
}

```

We then compute the average proportions over a 200-year period.

```

P_diwali <- matrix(0, ncol = 3, nrow = 200)
for(i in 1:200){
  P_diwali[i,] <- diwali_w(1899 + i, 20)
}
colMeans(P_diwali)

```

```
## [1] 0.01075 0.79800 0.19125
```

## Creating Trading Day Regressor

We generate the trading day regressor using the `td1nolpyear_regressor` function.

```

# Assuming 'data' length will be defined later
# td1nolpyear_regressor function defined earlier

```

## Preparing Time Series Objects

We create time series objects for the main data, Diwali indicator, and trading day regressor.

```

# Load helper functions if any
# source('code/helper_functions.r') # Uncomment if helper functions are in a separate file

# Loading data
data <- c(105.4, 106.4, 106.5, 107.5, 109.1, 112.4, 115.2, 117.3, 119.0, 121.1, 123.9, 118.7,
          115.6, 114.8, 115.7, 117.4, 118.8, 120.5, 125.4, 127.5, 126.4, 125.8, 125.3, 123.4,
          122.7, 122.7, 122.8, 123.4, 124.5, 127.1, 128.1, 130.3, 131.3, 132.4, 132.9, 131.3,
          131.1, 129.2, 129.2, 131.3, 133.8, 137.0, 138.8, 138.0, 136.5, 136.8, 135.6, 133.1,
          131.9, 131.8, 131.8, 132.1, 132.4, 134.1, 138.3, 140.1, 138.2, 139.4, 141.5, 139.7,
          138.1, 136.1, 135.5, 135.8, 136.5, 138.0, 140.1, 140.5, 138.9, 138.2, 137.8, 136.0,
          135.0, 135.1, 135.9, 137.3, 139.0, 141.1, 143.4, 144.7, 146.0, 149.1, 151.6, 155.3,
          153.4, 149.7, 147.8, 153.4, 151.8, 153.4, 156.7, 157.8, 161.6, 165.5, 166.0, 160.6,
          156.4, 155.5, 155.0, 156.4, 159.4, 161.3, 162.9, 162.7, 162.7, 166.9, 169.1, 167.1,
          164.9, 164.6, 166.9, 169.4, 172.1, 173.8, 173.8, 175.1, 176.7, 178.6, 177.0, 174.1,
          174.8, 174.4, 174.9, 175.9, 177.2, 181.7, 193.8, 192.5, 188.4, 190.4, 192.4, 190.7,
          189.3, 189.5, 189.8, 191.2, 192.6, 198.7, 204.3, 203.4)

diwali_ind <- c(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0405, -0.0405, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3405, -0.3405, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.6595, 0.6595, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3405, -0.3405, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3405, -0.3405, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.3595, 0.3595, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3405, -0.3405, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.6595, 0.6595, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.0595, 0.0595, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3405, -0.3405, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -0.6595, 0.6595, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3405, -0.3405, 0.0,
               0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3405, -0.3405, 0.0)

```

```

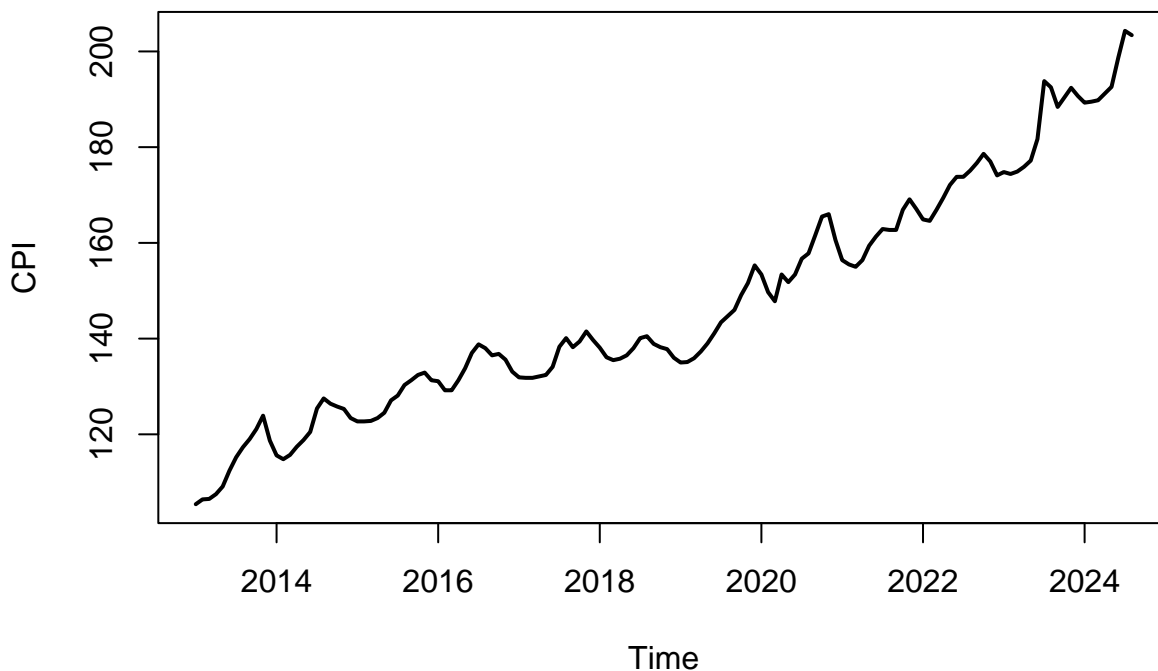
# Create trading day regressor
tdinolpyear <- tdinolpyear_regressor(start = c(2013, 1), n = length(data))

# It matches exactly with the regressor being used in the program
# Here is the image of the regressor of the program:
# ![Regressor Image](path_to_image.png) # Replace with actual image path

# Create time series objects
y <- ts(data, start = c(2013, 1), frequency = 12)
plot(y, main = "Monthly CPI of India", lwd = 2, ylab = "CPI")

```

## Monthly CPI of India



```

diwali <- ts(diwali_ind[1:length(y)], start = c(2013, 1), frequency = 12)
tdinolpyear <- ts(tdinolpyear, start = c(2013, 1), frequency = 12)

print(tdinolpyear)

```

```

##      Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
## 2013  3.0  0.0 -4.0  2.0  3.0 -5.0  3.0 -0.5 -1.5  3.0 -1.5 -0.5
## 2014  3.0  0.0 -4.0  2.0 -0.5 -1.5  3.0 -4.0  2.0  3.0 -5.0  3.0
## 2015 -0.5  0.0 -0.5  2.0 -4.0  2.0  3.0 -4.0  2.0 -0.5 -1.5  3.0
## 2016 -4.0  1.0  3.0 -1.5 -0.5  2.0 -4.0  3.0  2.0 -4.0  2.0 -0.5
## 2017 -0.5  0.0  3.0 -5.0  3.0  2.0 -4.0  3.0 -1.5 -0.5  2.0 -4.0
## 2018  3.0  0.0 -0.5 -1.5  3.0 -1.5 -0.5  3.0 -5.0  3.0  2.0 -4.0
## 2019  3.0  0.0 -4.0  2.0  3.0 -5.0  3.0 -0.5 -1.5  3.0 -1.5 -0.5
## 2020  3.0 -2.5 -0.5  2.0 -4.0  2.0  3.0 -4.0  2.0 -0.5 -1.5  3.0
## 2021 -4.0  0.0  3.0  2.0 -4.0  2.0 -0.5 -0.5  2.0 -4.0  2.0  3.0
## 2022 -4.0  0.0  3.0 -1.5 -0.5  2.0 -4.0  3.0  2.0 -4.0  2.0 -0.5
## 2023 -0.5  0.0  3.0 -5.0  3.0  2.0 -4.0  3.0 -1.5 -0.5  2.0 -4.0
## 2024  3.0  1.0 -4.0  2.0  3.0 -5.0  3.0 -0.5

```

```
print(y)
```

```
##           Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 2013 105.4 106.4 106.5 107.5 109.1 112.4 115.2 117.3 119.0 121.1 123.9 118.7
## 2014 115.6 114.8 115.7 117.4 118.8 120.5 125.4 127.5 126.4 125.8 125.3 123.4
## 2015 122.7 122.7 122.8 123.4 124.5 127.1 128.1 130.3 131.3 132.4 132.9 131.3
## 2016 131.1 129.2 129.2 131.3 133.8 137.0 138.8 138.0 136.5 136.8 135.6 133.1
## 2017 131.9 131.8 131.8 132.1 132.4 134.1 138.3 140.1 138.2 139.4 141.5 139.7
## 2018 138.1 136.1 135.5 135.8 136.5 138.0 140.1 140.5 138.9 138.2 137.8 136.0
## 2019 135.0 135.1 135.9 137.3 139.0 141.1 143.4 144.7 146.0 149.1 151.6 155.3
## 2020 153.4 149.7 147.8 153.4 151.8 153.4 156.7 157.8 161.6 165.5 166.0 160.6
## 2021 156.4 155.5 155.0 156.4 159.4 161.3 162.9 162.7 162.7 166.9 169.1 167.1
## 2022 164.9 164.6 166.9 169.4 172.1 173.8 173.8 175.1 176.7 178.6 177.0 174.1
## 2023 174.8 174.4 174.9 175.9 177.2 181.7 193.8 192.5 188.4 190.4 192.4 190.7
## 2024 189.3 189.5 189.8 191.2 192.6 198.7 204.3 203.4
```

## Model Fitting and Transformation

We fit ARIMA models to the time series data `y` and its logarithm `log(y)`, then compare them using the Akaike Information Criterion corrected (AICc) to decide on the appropriate transformation.

```
# Regression matrix
Xreg <- cbind(tdlnolpyear, diwali)

# Fit ARIMA models without transformation
no_transformation_test_fit <- Arima(y, xreg = Xreg,
                                   order = c(0,1,1), seasonal = c(0,1,1),
                                   include.mean = FALSE, method = 'ML', lambda = NULL)

# Fit ARIMA models with log transformation
log_transformation_test_fit <- Arima(log(y), xreg = Xreg,
                                   order = c(0,1,1), seasonal = c(0,1,1),
                                   include.mean = FALSE, method = 'ML', lambda = NULL)

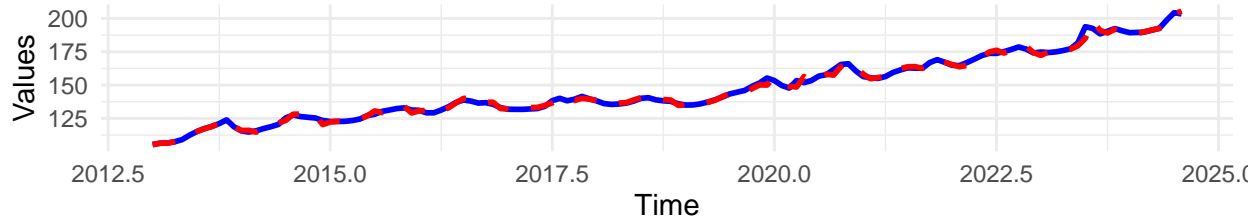
# Plot model outputs
plot_model_output(no_transformation_test_fit, series_name = "No Transformation")

## Series: y
## Regression with ARIMA(0,1,1)(0,1,1)[12] errors
##
## Coefficients:
##          ma1          sma1  tdlnolpyear  diwali
##          0.3482  -0.9124         -0.0074  0.2917
## s.e.    0.0952   0.1868          0.0253  0.3759
##
## sigma^2 = 3.167: log likelihood = -261.37
## AIC=532.74  AICc=533.23  BIC=546.96
##
## Training set error measures:
##              ME      RMSE      MAE      MPE      MAPE      MASE
## Training set 0.03178592 1.668093 1.097779 0.001223092 0.7192208 0.1414778
##              ACF1
## Training set -0.06194926
##
```



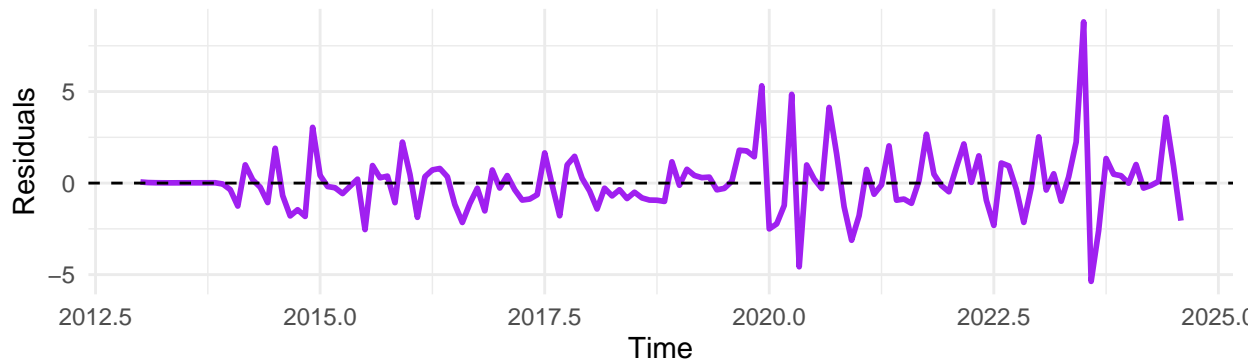
```
## Box-Ljung test
##
## data: residuals
## X-squared = 14.742, df = 20, p-value = 0.791
```

### No Transformation : Original vs Fitted



Model Summary: AIC = 532.74 , npar= 5 , BIC = 546.96 , AICc = 533.23 , HQ = 538.52 , Log-Likelihood = -261.37  
 Ljung-Box Test p-value: 0.791  
 Model Equation: ARIMA(0, 1, 1) x (0, 1, 1) Method = ML  
 Coefficients (Estimate  $\pm$  Std. Error):  
 ma1 = 0.3482  $\pm$  0.0952, sma1 = -0.9124  $\pm$  0.1868, td1nolpyear = -0.0074  $\pm$  0.0253, diwali = 0.2917  $\pm$  0.3759

### No Transformation : Residuals

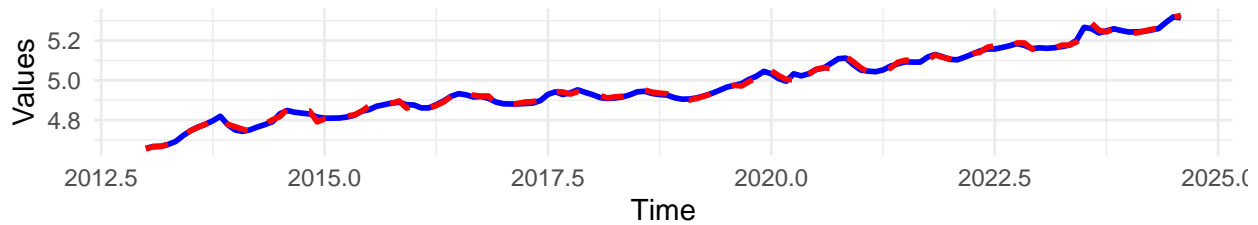


```
plot_model_output(log_transformation_test_fit,
  series_name = "Log Transformed Series", adj_inf_criteria_for_log = TRUE)
```

```
## Series: log(y)
## Regression with ARIMA(0,1,1)(0,1,1)[12] errors
##
## Coefficients:
##      ma1      sma1  td1nolpyear  diwali
##      0.3325 -1.0000      -1e-04  0.0015
## s.e.  0.0916  0.1821      2e-04  0.0024
##
## sigma^2 = 0.00012: log likelihood = 381.54
## AIC=-753.09  AICc=-752.59  BIC=-738.87
##
## Training set error measures:
##              ME      RMSE      MAE      MPE      MAPE
## Training set -0.0005356371 0.01026985 0.007298123 -0.01142566 0.1458139
##              MASE      ACF1
## Training set 0.1402815 -0.03762701
##
## Box-Ljung test
##
## data: residuals
```

```
## X-squared = 12.787, df = 20, p-value = 0.8863
```

### Log Transformed Series : Original vs Fitted



Model Summary: AIC = 517.89 , npar= 5 , BIC = 532.11 , AICc = 518.38 ,HQ = 523.66 , Log-Likelihood = 381.54

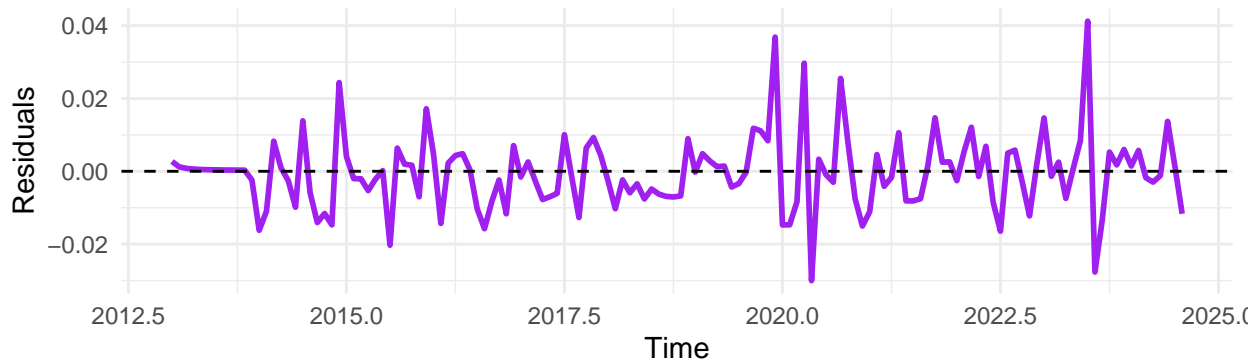
Ljung-Box Test p-value: 0.8863

Model Equation: ARIMA(0, 1, 1) x (0, 1, 1) Method = ML

Coefficients (Estimate ± Std. Error):

ma1 = 0.3325 ± 0.0916, sma1 = -1 ± 0.1821, td1nolyear = -1e-04 ± 2e-04, diwali = 0.0015 ± 0.0024

### Log Transformed Series : Residuals



### Interpretation of AICc

The AICc values from both models are compared to determine whether a log transformation is preferable. According to the comments, if  $AICC_{nolog} - AICC_{log} < -2$ , we prefer no log transform. In this case:

$AICC_{nolog} - AICC_{log} = 529.65 - 514.66 = 14.99$

Since 14.99 is not less than -2, we prefer the log-transformed model.

```
Z <- log(y)
```

## Default Model Estimation

We proceed with fitting the default ARIMA model to the log-transformed series Z. This includes initial outlier identification and tests for trading day effects.

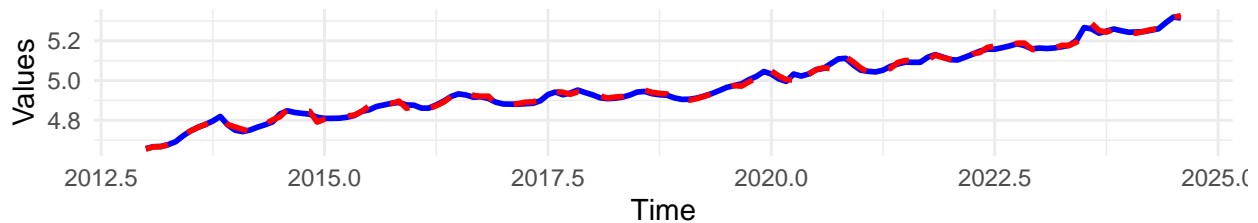
```
# Create constant term
const <- const_term(y, 1, 1)

# Fit default ARIMA model without constant term
default_model_fit1 <- Arima(Z, xreg = Xreg, order = c(0,1,1),
                           seasonal = c(0,1,1), include.mean = FALSE,
                           method = 'ML', lambda = NULL)

# Plot model output
plot_model_output(default_model_fit1, adj_inf_criteria_for_log = TRUE)
```

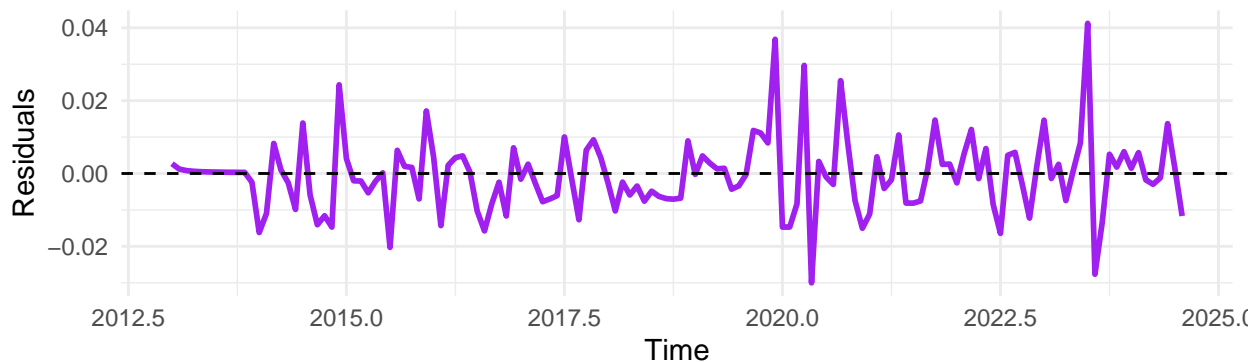
```
## Series: Z
## Regression with ARIMA(0,1,1)(0,1,1)[12] errors
##
## Coefficients:
##      ma1      sma1  td1nolpyear  diwali
##      0.3325 -1.0000      -1e-04  0.0015
## s.e.  0.0916   0.1821      2e-04  0.0024
##
## sigma^2 = 0.00012: log likelihood = 381.54
## AIC=-753.09  AICc=-752.59  BIC=-738.87
##
## Training set error measures:
##              ME      RMSE      MAE      MPE      MAPE
## Training set -0.0005356371 0.01026985 0.007298123 -0.01142566 0.1458139
##              MASE      ACF1
## Training set 0.1402815 -0.03762701
##
## Box-Ljung test
##
## data: residuals
## X-squared = 12.787, df = 20, p-value = 0.8863
```

Series : Original vs Fitted



Model Summary: AIC = 517.89 , npar= 5 , BIC = 532.11 , AICc = 518.38 , HQ = 523.66 , Log-Likelihood = 381.54  
 Ljung-Box Test p-value: 0.8863  
 Model Equation: ARIMA(0, 1, 1) x (0, 1, 1) Method = ML  
 Coefficients (Estimate ± Std. Error):  
 ma1 = 0.3325 ± 0.0916, sma1 = -1 ± 0.1821, td1nolpyear = -1e-04 ± 2e-04, diwali = 0.0015 ± 0.0024

Series : Residuals



```
# Perform t-test on residuals to test the presence of a constant term
t.test(residuals(default_model_fit1))
```

```
##
## One Sample t-test
##
```

```
## data: residuals(default_model_fit1)
## t = -0.61575, df = 139, p-value = 0.5391
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
## -0.002255568 0.001184294
## sample estimates:
## mean of x
## -0.0005356371
```

### Interpretation:

The p-value from the t-test (0.5391) indicates that the constant term is not significant. Therefore, we will not include the constant term in the model.

## Trading Day Regressor Selection

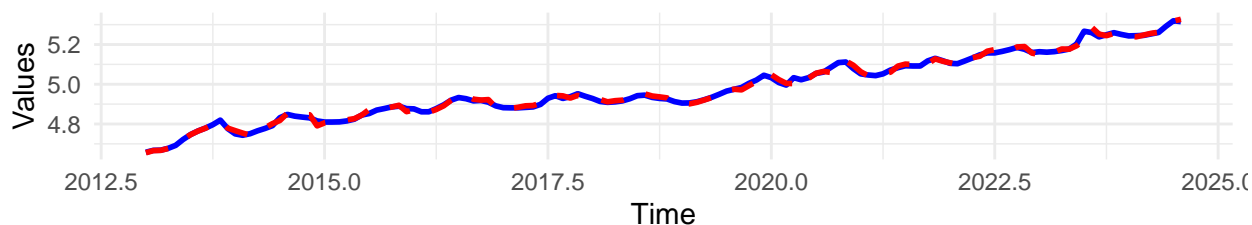
We evaluate whether to include the trading day regressor (`tdinolpyear`) and the Diwali regressor (`diwali`) based on AICc differences.

```
# AIC test for tdinolpyear and diwali

# Model with trading day regressor
model_with_td <- Arima(Z, xreg = tdinolpyear,
                      order = c(0,1,1), seasonal = c(0,1,1),
                      include.mean = FALSE, method = 'ML', lambda = NULL)
plot_model_output(model_with_td,
                  series_name = "Model with Trading Day Regressor",
                  adj_inf_criteria_for_log = TRUE)

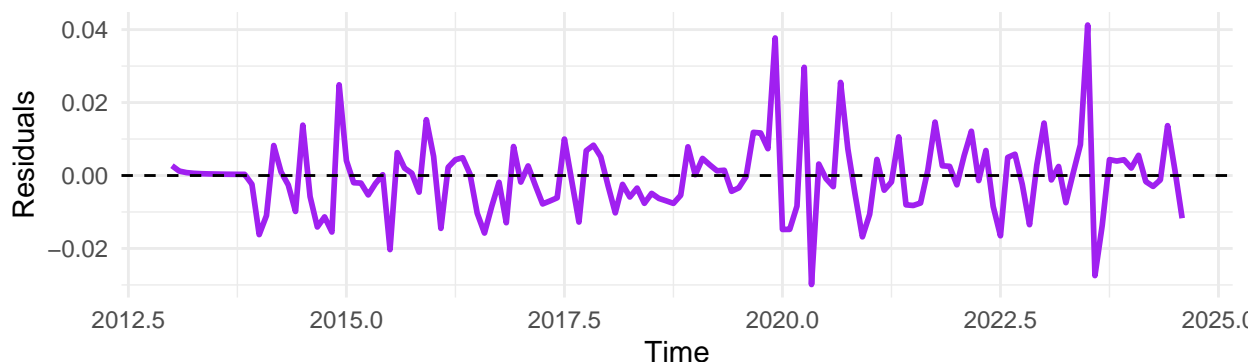
## Series: Z
## Regression with ARIMA(0,1,1)(0,1,1)[12] errors
##
## Coefficients:
##          ma1          sma1          xreg
##          0.3274      -0.9998      -1e-04
## s.e.  0.0913      0.1641      2e-04
##
## sigma^2 = 0.0001195: log likelihood = 381.35
## AIC=-754.71  AICc=-754.38  BIC=-743.33
##
## Training set error measures:
##              ME          RMSE          MAE          MPE          MAPE
## Training set -0.0005362574 0.01028626 0.007273905 -0.01144279 0.1453235
##              MASE          ACF1
## Training set 0.139816 -0.03619314
##
## Box-Ljung test
##
## data: residuals
## X-squared = 13.127, df = 20, p-value = 0.8719
```

## Model with Trading Day Regressor : Original vs Fitted



Model Summary: AIC = 516.27 , npar= 4 , BIC = 527.64 , AICc = 516.59 , HQ = 520.89 , Log-Likelihood = 381.35  
 Ljung-Box Test p-value: 0.8719  
 Model Equation: ARIMA(0, 1, 1) x (0, 1, 1) Method = ML  
 Coefficients (Estimate ± Std. Error):  
 ma1 = 0.3274 ± 0.0913, sma1 = -0.9998 ± 0.1641, xreg = -1e-04 ± 2e-04

## Model with Trading Day Regressor : Residuals

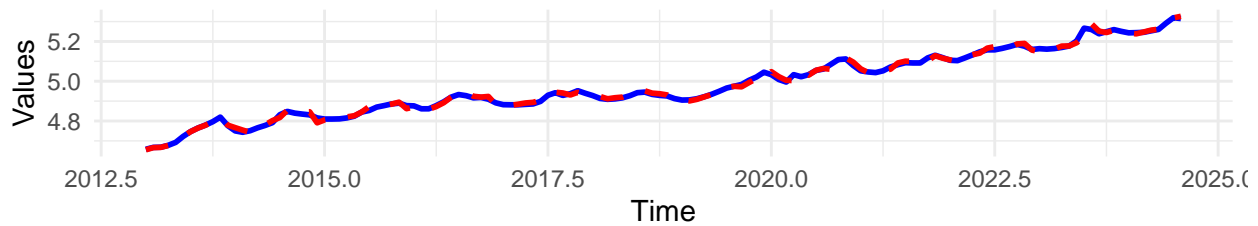


```
# Model without trading day regressor
model_without_td <- Arima(Z, order = c(0,1,1),
                          seasonal = c(0,1,1), include.mean = FALSE,
                          method = 'ML', lambda = NULL)
plot_model_output(model_without_td,
                  series_name = "Model without Trading Day Regressor",
                  adj_inf_criteria_for_log = TRUE)
```

```
## Series: Z
## ARIMA(0,1,1)(0,1,1)[12]
##
## Coefficients:
##      ma1      sma1
##      0.3304 -1.0000
## s.e.  0.0907  0.1826
##
## sigma^2 = 0.0001187: log likelihood = 381.25
## AIC=-756.51 AICc=-756.31 BIC=-747.97
##
## Training set error measures:
##              ME      RMSE      MAE      MPE      MAPE
## Training set -0.0005364757 0.01029314 0.007268806 -0.01144638 0.1452537
##              MASE      ACF1
## Training set 0.139718 -0.03733072
##
## Box-Ljung test
##
```

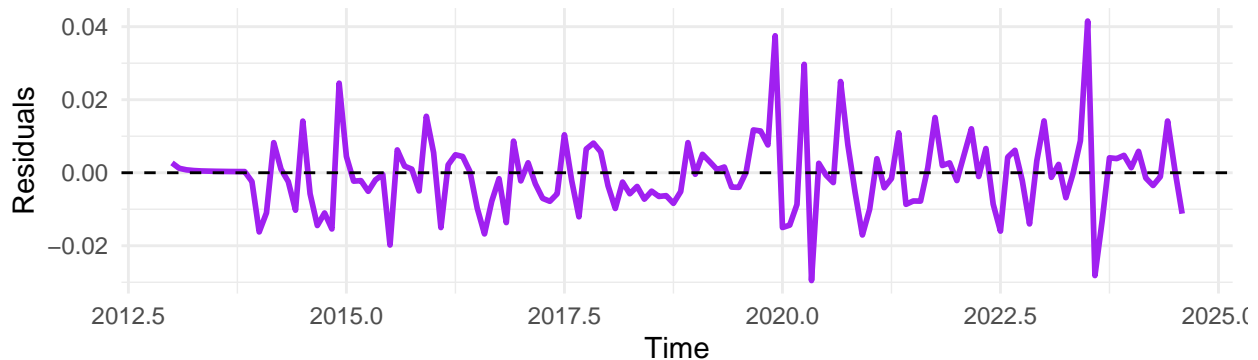
```
## data: residuals
## X-squared = 13.555, df = 20, p-value = 0.8523
```

### Model without Trading Day Regressor : Original vs Fitted



Model Summary: AIC = 516.47 , npar= 4 , BIC = 527.84 , AICc = 516.79 ,HQ = 521.09 , Log-Likelihood = 381.25  
 Ljung-Box Test p-value: 0.8523  
 Model Equation: ARIMA(0, 1, 1) x (0, 1, 1) Method = ML  
 Coefficients (Estimate ± Std. Error):  
 ma1 = 0.3304 ± 0.0907, sma1 = -1 ± 0.1826

### Model without Trading Day Regressor : Residuals



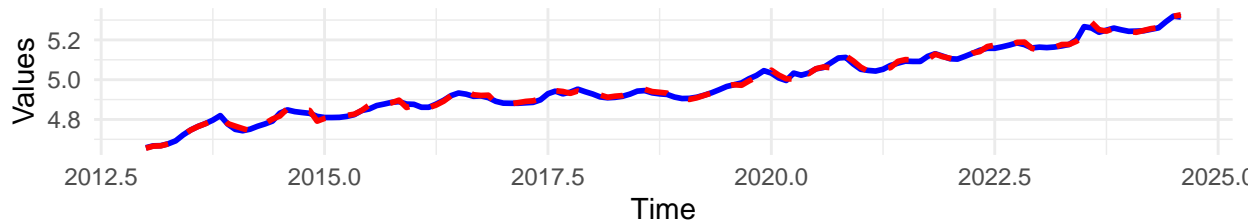
```
# AICC comparison
# AICC_with - AICC_without = -754.38 + 754.18 = 0.2 > 0
# Trading day regressor will not be included in the model
```

```
# Model with Diwali regressor
model_with_diwali <- Arima(Z, xreg = diwali, order = c(0,1,1),
  seasonal = c(0,1,1), include.mean = FALSE,
  method = 'ML', lambda = NULL)
plot_model_output(model_with_diwali,
  series_name = "Model with Diwali Regressor",
  adj_inf_criteria_for_log = TRUE)
```

```
## Series: Z
## Regression with ARIMA(0,1,1)(0,1,1)[12] errors
##
## Coefficients:
##          ma1      sma1      xreg
##          0.3356 -1.0000  0.0015
## s.e.  0.0910   0.2093  0.0024
##
## sigma^2 = 0.0001192: log likelihood = 381.45
## AIC=-754.9  AICc=-754.57  BIC=-743.53
##
## Training set error measures:
##              ME      RMSE      MAE      MPE      MAPE      MASE
```

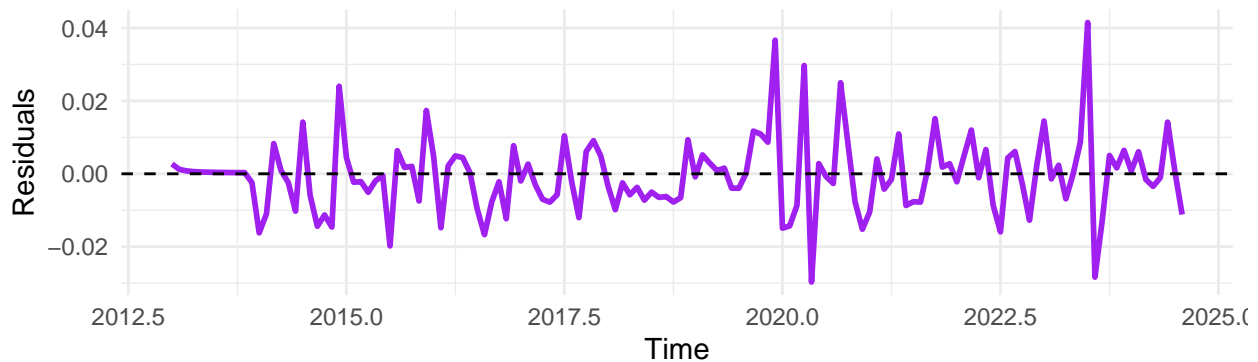
```
## Training set -0.000535844 0.01027714 0.007297058 -0.01142898 0.1458251 0.140261
## ACF1
## Training set -0.03880191
##
## Box-Ljung test
##
## data: residuals
## X-squared = 13.168, df = 20, p-value = 0.8701
```

### Model with Diwali Regressor : Original vs Fitted



Model Summary: AIC = 516.07 , npar= 4 , BIC = 527.45 , AICc = 516.4 , HQ = 520.69 , Log-Likelihood = 381.45  
 Ljung-Box Test p-value: 0.8701  
 Model Equation: ARIMA(0, 1, 1) x (0, 1, 1) Method = ML  
 Coefficients (Estimate ± Std. Error):  
 ma1 = 0.3356 ± 0.091, sma1 = -1 ± 0.2093, xreg = 0.0015 ± 0.0024

### Model with Diwali Regressor : Residuals

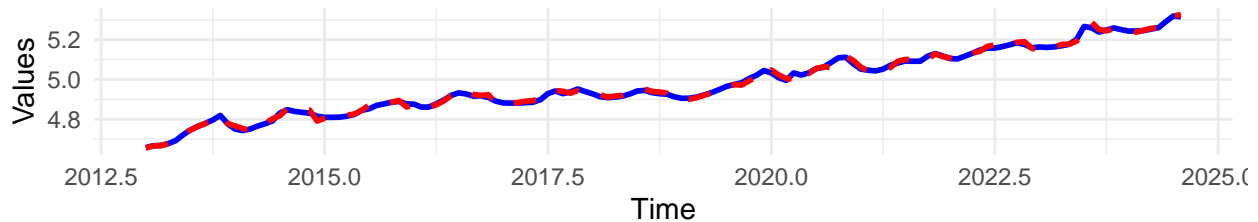


```
# Model without Diwali regressor
model_without_diwali <- Arima(Z, order = c(0,1,1),
                             seasonal = c(0,1,1), include.mean = FALSE,
                             method = 'ML', lambda = NULL)
plot_model_output(model_without_diwali,
                  series_name = "Model without Diwali Regressor",
                  adj_inf_criteria_for_log = TRUE)
```

```
## Series: Z
## ARIMA(0,1,1)(0,1,1)[12]
##
## Coefficients:
##      ma1      sma1
## 0.3304 -1.0000
## s.e. 0.0907 0.1826
##
## sigma^2 = 0.0001187: log likelihood = 381.25
## AIC=-756.51 AICc=-756.31 BIC=-747.97
##
```

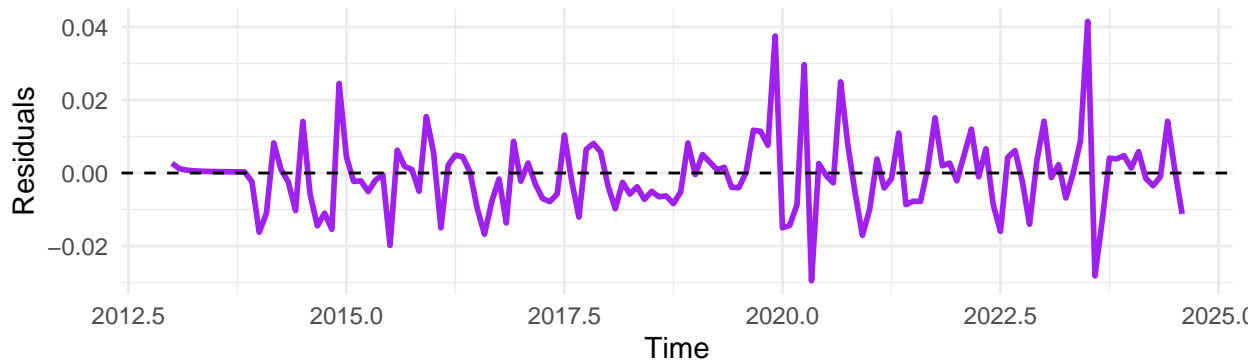
```
## Training set error measures:
##           ME           RMSE           MAE           MPE           MAPE
## Training set -0.0005364757 0.01029314 0.007268806 -0.01144638 0.1452537
##           MASE           ACF1
## Training set 0.139718 -0.03733072
##
## Box-Ljung test
##
## data: residuals
## X-squared = 13.555, df = 20, p-value = 0.8523
```

Model without Diwali Regressor : Original vs Fitted



Model Summary: AIC = 516.47 , npar= 4 , BIC = 527.84 , AICc = 516.79 , HQ = 521.09 , Log-Likelihood = 381.25  
 Ljung-Box Test p-value: 0.8523  
 Model Equation: ARIMA(0, 1, 1) x (0, 1, 1) Method = ML  
 Coefficients (Estimate ± Std. Error):  
 ma1 = 0.3304 ± 0.0907, sma1 = -1 ± 0.1826

Model without Diwali Regressor : Residuals



```
# AICC comparison
# AICC_with - AICC_without = -754.57 + 754.18 = 0.39 > 0
# Diwali regressor will not be included in the model
```

## Outlier Detection in Default Model

We fit the default ARIMA model and perform forward and backward passes to identify significant outliers.

```
# Fit default ARIMA model without regressors
Xreg <- NULL
default_model <- Arima(Z, order = c(0,1,1), seasonal = c(0,1,1), xreg = Xreg,
  include.mean = FALSE, method = 'ML', lambda = NULL)

default_model
```

```
## Series: Z
## ARIMA(0,1,1)(0,1,1)[12]
```



```

##
## Coefficients:
##      ma1      sma1
##      0.3304 -1.0000
## s.e.  0.0907  0.1826
##
## sigma^2 = 0.0001187: log likelihood = 381.25
## AIC=-756.51 AICc=-756.31 BIC=-747.97
# Note: The coefficients are negative compared to X13-ARIMA-SEATS
# due to different parameterizations.

# Forward Pass for Outlier Detection
print("Forward pass 1")

## [1] "Forward pass 1"

curr_outlier <- forward_pass(default_model, xreg = Xreg,
                             types = c("AO", "LS"), tcritical = 3.88)

## [1] "A02020.04 4.17002561699784"
## [1] "A02023.07 4.08495930589087"
## [1] "LS2019.12 3.88044849190372"
## [1] "LS2023.07 4.64218785557147"

# default value taken from the manual See X-13ARIMA-SEATS Manual
k <- 2
while(!is.null(curr_outlier)){
  Xreg <- cbind(Xreg, curr_outlier)
  print(paste("Forward pass", k))
  k <- k + 1
  curr_outlier <- forward_pass(default_model,
                              xreg = Xreg,
                              types = c("AO", "LS"), tcritical = 3.88)
}

## [1] "Forward pass 2"
## [1] "A02020.03 3.92397472587285"
## [1] "A02020.04 4.51624029586687"
## [1] "LS2019.12 4.23721133059193"
## [1] "LS2020.04 4.12090040971873"
## [1] "Forward pass 3"
## [1] "A02013.11 4.17134927338926"
## [1] "LS2019.12 4.56295072308321"
## [1] "Forward pass 4"
## [1] "A02013.11 4.10983186897022"
## [1] "Forward pass 5"

# Backward Pass for Outlier Detection
k <- 1
while(TRUE){
  print(paste("Backward pass", k))
  k <- k + 1
  ind <- backward_pass(default_model, Xreg, tcritical = 3.88)
  if(is.null(ind)){
    print("Outlier detection is done")
    break
  }
}

```

```

    }
    else{
      Xreg <- Xreg[,-ind]
    }
  }
}

```

```

## [1] "Backward_pass 1"
## [1] "Outlier detection is done"

```

```

# Outliers identified from the default model
print(colnames(Xreg))

```

```

## [1] "LS2023.07" "A02020.04" "LS2019.12" "A02013.11"

```

### Interpretation:

The identified outliers (`colnames(Xreg)`) match those obtained from the X13-ARIMA-SEATS program, confirming the accuracy of our implementation.

## Next Steps

Our next goal is to understand and implement the Iterative Generalized Least Squares (IGLS) algorithm to align the parameter estimates with those from the X13-ARIMA-SEATS program.

## Conclusion

This document detailed the implementation of the `regARIMA` procedure using X13-ARIMA-SEATS in R. We covered data preparation, model fitting, transformation decisions based on AICc, AIC tests for regressors and outlier detection.

## References

- X-13ARIMA-SEATS Manual
- R Documentation for `Arima`