Time Series Analysis and Forecasting of Sales Data (2015-24)

1. Introduction

This report presents an analysis of a time series dataset representing monthly product sales from January 2015 to December 2024. The primary objective is to identify and fit the most appropriate time series models, with a specific focus on forecasting sales for the period from January to June 2025, while also providing prediction intervals to assess the reliability of these forecasts.

The analysis explores three different approaches: ARIMA, time-series regression, and dynamic linear models (DLM). Each method undergoes an iterative process involving model identification, estimation, and diagnostic checking to ensure accuracy. The results of these models are then compared to determine the most suitable method for reliable and effective forecasting of future sales.

2. Model Identification and Fitting

2.1 Data Preparation

The dataset comprises five columns, each representing a different dataset. For this analysis, dataset 3 (column 3) is selected. The data consists of 120 monthly observations.

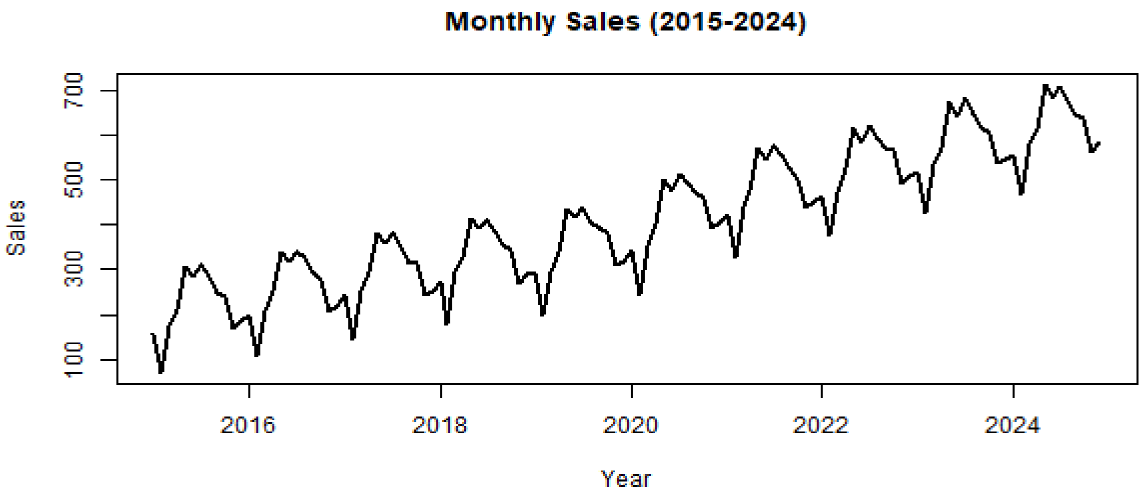


Figure 1: Sales data (considered column 3 of the given dataset)

The graph in Figure 1 illustrates a time series with several key features. It shows a consistent upward trend from 2015 to 2024, indicating a steady increase in sales over time. Recurring seasonal patterns are evident, with fluctuations in sales repeating at regular intervals (presumably yearly). Notably, the magnitude of these seasonal variations remains relatively consistent across the entire time span, supporting the use of an additive model where the seasonal component is independent of the trend.
First-order and seasonal differencing were applied to remove trends and seasonality, as demonstrated in Appendix A (see Figure A.1 and Code A.1).

```
       One Sample t-test
data:  diff_ts
t = -0.050672, df = 106, p-value = 0.9597
sample estimates: mean of x is -0.0417757
```

On performing t-test on the differenced time series obtained p value > 0.05 (as shown above) i.e. failed to reject null hypothesis, resulting in a stationary series with a mean close to zero. The ACF plot shows significant spikes at lower lags before tapering off, suggesting the need for both autoregressive (AR) and moving average (MA) terms for further modeling, such as ARIMA (see Appendix A Figure A.1).

2.2 ARIMA Modeling

Based on the previous discussion, it is concluded that both seasonal and non-seasonal differencing are required to make the data stationary. However, to obtain the best-fit model, appropriate AR and MA values are also necessary. To determine these values, several ARIMA models were iteratively tested. Table 1 summarizes the key features of each model to identify the best-fit model (refer to Appendix A, Code A.2).

Model	Log-Likelihood	AIC	Residuals Behavior
ARIMA(1,1,0) Seasonal(0,1,0)	-373.13	750.26	Residuals exhibited some autocorrelation, suggesting room for improvement.
ARIMA(0,1,0) Seasonal(0,1,1)	-359.43	722.85	Residuals improved, but some autocorrelation remained, showing a better fit than the first model.
ARIMA(1,1,0) Seasonal(0,1,1)	**-351.21**	**708.42**	**Residuals showed significant improvement with fewer autocorrelations. This model provided the best fit.**
ARIMA(2,1,0) Seasonal(0,1,1)	-351.21	710.42	Residuals slightly improved, but AIC was slightly higher, indicating no substantial gain in fit.

Table 1: ARIMA best fit model testing results

As expected from the ACF/PACF of the differenced time series (see Appendix A, Figure A.1), ARIMA(1,1,0) with Seasonal $(0,1,1)_{12}$ provided a good fit based on residual analysis and ACF/PACF patterns. To further evaluate these models, the `sarima()` function was used for additional diagnostics of the fit (see Appendix A, Code A.3, and Figure A.2). The **ARIMA(1,1,0)(0,1,1)[12]** was identified as the best fit, as it exhibited high p-values in the Ljung-Box test (indicating no significant autocorrelation in the residuals) and statistically significant coefficient p-values ($p < 0.05$).

2.3 Time-Series Regression Modeling

A time-series regression model is applied to same time series data, incorporating both a linear time trend and seasonal factors as predictors (see Figure 2).

```
time = c(1:120)
season = gl(12,1,120)
fit = lm(time_series~time+season)
fitted = ts(fitted.values(fit),
start=c(2015,1),frequency=12)
plot(time_series, main = "Regression fit
       model of Sales (2015-2024)", lw=2)

lines(fitted,col=2., lty=2, lw=3)
```

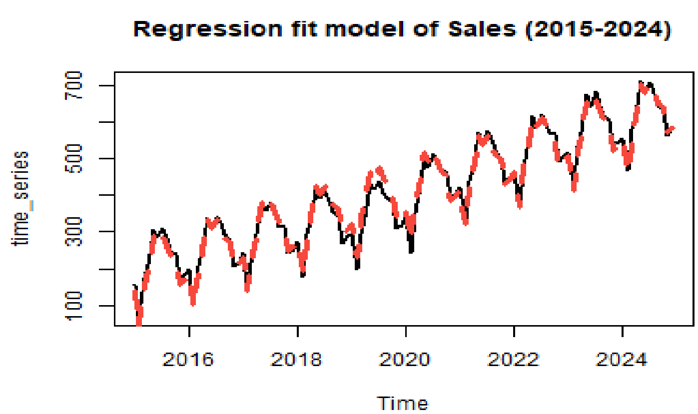


Figure 2: Fitted Values Overlaid on Original Monthly Sales (2015-2024) in Time Series Regression Modeling

The model showed a strong fit, with an R-squared value of 0.988 and an adjusted R-squared value of 0.987, indicating that the model explains most of the variation in the data. The fitted values overlaid on the original time series (see Figure 2) show that the model effectively captures both trend and seasonality, with the fitted line. This highlights the model's accuracy in representing the time series structure (see Appendix A Code A.4.1). The residual analysis reveals in Appendix A Figure A.3 shows no discernible patterns, indicating a good model fit. The ACF of the residuals diminishes, suggesting no significant autocorrelation, while the PACF shows a distinct cutoff at lag 2, indicating the potential need for an AR(2) model to capture any remaining dependencies.

The following ARIMA models were fitted: ARIMA(0,0,1), ARIMA(1,0,0), ARIMA(1,0,1), ARIMA(2,0,0), ARIMA(2,0,1), and ARIMA(3,0,0) (see Appendix Code A.4.2). Of these, ARIMA(2,0,1) provided the best fit. A summary of the results for each model is presented in Table 2.

Model	Sigma²	Log-Likelihood	AIC	Notes
ARIMA(1,0,0)	33.33	-381.67	767.34	Moderate fit, residuals show improvement, but autocorrelations persist.
ARIMA(1,0,1)	29.64	-374.72	755.44	Good fit; residuals improved with moderate autocorrelation.
ARIMA(3,0,0)	28.87	-373.19	754.38	Slight improvement over simpler models, but no substantial residual gains.
ARIMA(2,0,0)	28.88	-373.21	752.43	Good fit, residuals improved with lower sigma² and autocorrelations.
ARIMA(2,0,1) (Best Fit)	**28.87**	**-373.19**	**752.43**	**Best overall fit; residuals show minimal autocorrelations and high p-value for difference test.**

Table 2: Key results and diagnostics for each model

The ARIMA(2,0,1) model is the best fit due to its lowest AIC value of 752.43, indicating an efficient balance between model fit and complexity. Residual analysis shows minimal autocorrelations, suggesting the model captures underlying dependencies effectively. Additionally, its low variance estimate (Sigma²), indicates minimal error and strong predictive accuracy. These factors confirm ARIMA(2,0,1) as the optimal model, balancing complexity with performance.

2.4 Dynamic Linear Models

Model Fit: Dynamic Linear Models (DLMs) model components include trend, which is modeled as a second-order polynomial, and seasonality, represented using trigonometric functions (s=12) with seasonal frequency to account for monthly seasonality. This combination allows DLMs to effectively capture both long-term trends and seasonal variations in time series data.

```
build_sales = function(params) {
  dlmModPoly(order=2,dV=exp(params[1]),dW=exp(params[2:3]))+
dlmModTrig(s=12,dV=0,dW=exp(rep(params[4],11)))
}
sales_fit = dlmMLE(time_series, parm=c(0,0,0,0), build=build_sales)
sales_fit$convergence
```

```
## [1] 0
```

The model parameters are estimated using maximum likelihood estimation (MLE) through the dlmMLE function, which converges successfully, as indicated by a convergence value of 0.

The dynamic linear model (DLM) is then refined by smoothing with the dlmSmooth function to estimate the trend and seasonal components. The smoothed results separate the observed sales data into trend, capturing long-term movement, and seasonality, representing recurring monthly patterns. The resulting plot in figure 3 illustrates how the DLM effectively separates the observed sales data into these components, providing deeper insights into the time series structure (see Appendix A Code A.5).

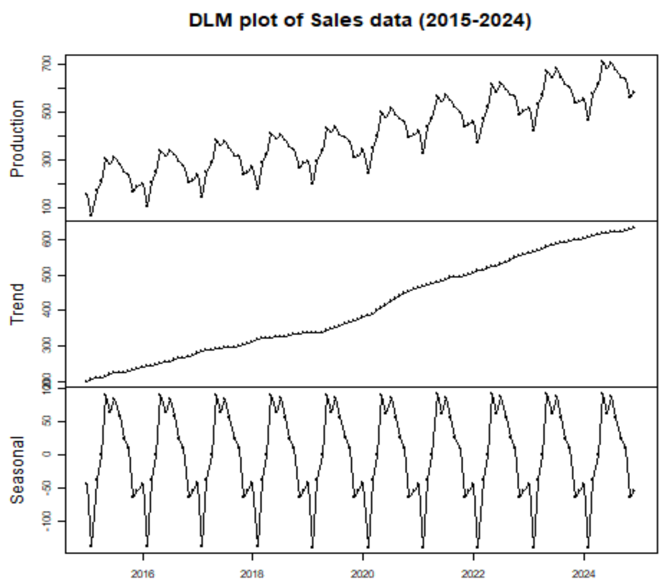


Figure 3: DLM fitted plot of sales data

3. Forecasting

3.1 Forecast for January to June 2025

Forecasting sales using time series models involves analysing historical sales data to predict future trends and patterns. For this analysis, SARIMA (Seasonal AutoRegressive Integrated Moving Average), Time-Series Regression, and Dynamic Linear Models (DLM) are used which are designed in section 2. Each method captures different aspects of the data, such as seasonality, trends, and potential structural changes. These techniques allow for robust predictions of future monthly sales, accompanied by 95% prediction intervals to account for uncertainty, providing valuable insights for decision-making.

3.1.1 ARIMA Forecast

Using the SARIMA model configured as **ARIMA(1,1,0)(0,1,1)[12]**, a sales forecast for the period January to June 2025 was generated. The forecast includes predicted monthly sales along with 95% prediction intervals to account for uncertainty, as shown in Figure 4.

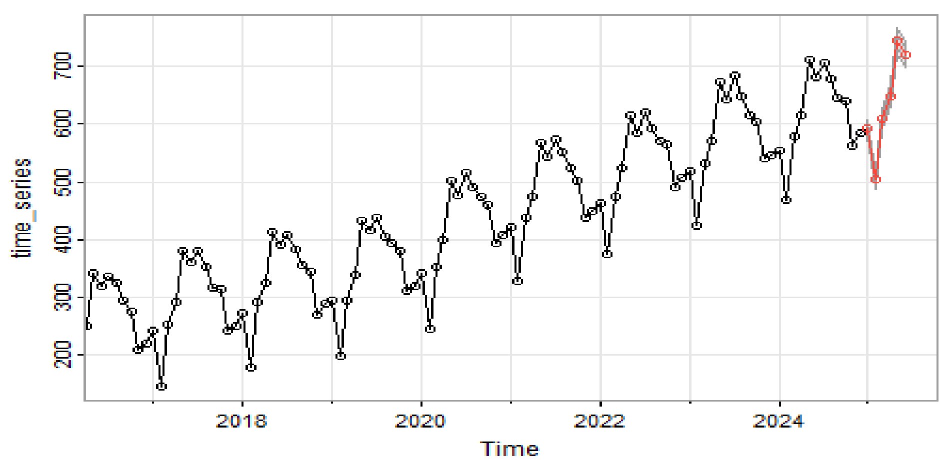


Figure 4: ARIMA forecasted plot for Jan-June 2025 using Sales data (2015-2024)

These results provide valuable insights into the anticipated sales trajectory, supporting future planning and decision-making (for the code, see Appendix B, Code B.1.1).

3.1.2 Time-Series Regression Forecast

Using a time-series regression model fitted in section 2.3, the forecast for monthly sales from January to June 2025 was generated by combining trend and seasonal components with residual adjustments. The model first extrapolates the trend for the forecasted months using the linear coefficients estimated from the regression. Seasonal effects, represented by the estimated seasonal coefficients, were added to the trend values to account for recurring patterns in the data. The residuals were then modeled using an ARIMA(2,0,0) process, and their predictions were incorporated to refine the forecast further.

The final forecasted values combine these three components—trend, seasonality, and residual adjustments—providing a more accurate prediction of future sales. Additionally, 95% prediction intervals were calculated to reflect the associated uncertainty, with upper and lower bounds generated using the forecast error.

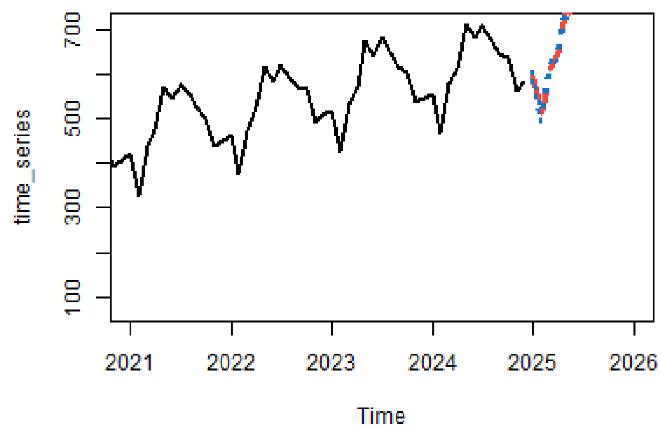


Figure 5: Time Series regression forecasted model for Jan -June 2025

The plot illustrates the historical time series along with the forecasted values, prediction intervals, and expected trajectory for the forecasted period. The forecasted sales, shown as a dashed red line, align closely with the observed trend and seasonality. The blue dashed lines represent the upper and lower 95% prediction intervals, capturing the range within which the actual values are likely to fall. This visualization shown below confirms that the model expects the upward trend to continue while retaining the seasonal fluctuations observed in historical data. (see Appendix B Code B.1.2)

3.1.3 Dynamic Linear Model Forecast

A Dynamic Linear Model (DLM) designed in section 2.4 is used to forecast sales for the next six months, providing point estimates and 95% prediction intervals. The forecasted values reflect the observed trend and seasonality, with prediction intervals offering bounds for potential variability. The plot in figure 6 shows historical data (black), forecasts (red), and dashed red lines for prediction intervals, illustrating the expected sales trajectory and associated uncertainty (see Appendix Code B.1.3).

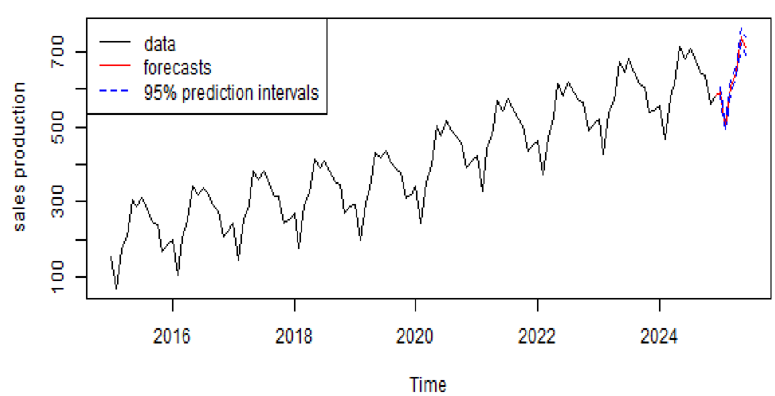


Figure 6: DLM forecasted model for Jan-June 2025

3.2 Reliability of Forecasting

To evaluate the reliability of the forecasting methods, each model was trained on data from 2015 to 2023 (the first nine years), and forecasts were made for the subsequent twelve months (2024). The forecasts from the ARIMA, Time-Series Regression, and Dynamic Linear Model (DLM) were then compared to the actual observed values.

3.2.1 Forecasting for the last 12 months Using ARIMA

The ARIMA model is used to forecast sales for the last 12 months (January to December 2024). The model is fitted on historical data from January 2015 to December 2023. The actual sales data (blue line) and the forecasted values are plotted for comparison, with the forecast covering the final 12 months. The graph in figure 7 shows this comparison for verification.(see Appendix B Code B.2.1)

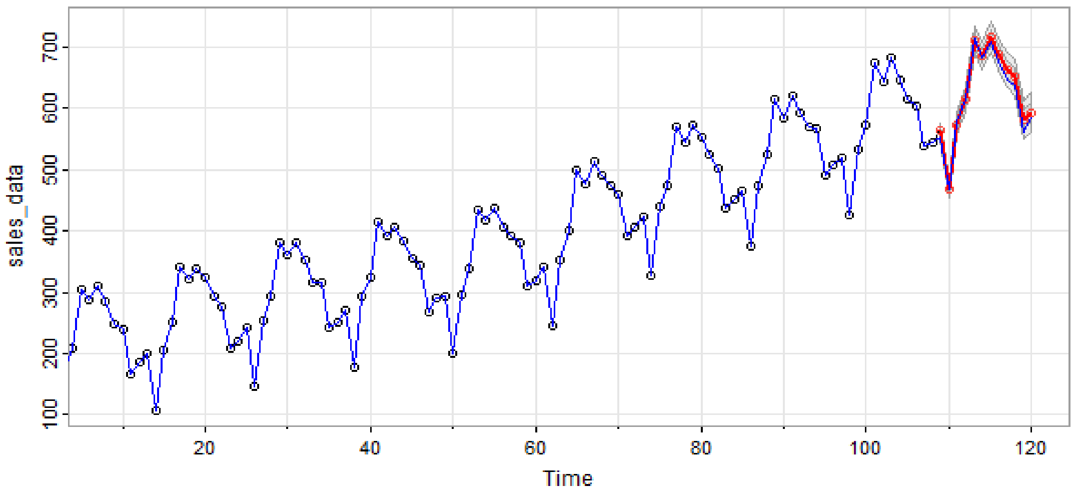


Figure 7: Monthly Sales Data with Forecast and Prediction Intervals Using ARIMA

3.2.2 Forecasting for the last 12 months Using Regression model

A linear model is first fitted to the sales data from January 2015 to December 2023 to account for trend and seasonality. The residuals from this model are subsequently fitted to an ARIMA(2,0,1) model, identified as the best-fit model for time series dataset as determined in Section 2.3.

The forecast for January to December 2024 is obtained by combining the trend, seasonal effects, and ARIMA predictions, along with 95% prediction intervals (see Appendix B Code B.2.2). Figure 8 (Left) visualizes this, showing the actual sales data (black line), forecasted values (red line), and the 95% prediction intervals (blue dashed lines).

3.2.3 Forecasting for the last 12 months Using DLM models

Similar to the analysis of the models above, a Dynamic Linear Model (DLM) is fitted to the first 9 years of data, and forecasts are generated for the next 12 months. The graph in Figure 8 (Right) illustrates the actual sales data (black line), forecasted values (red line), and 95% prediction intervals (blue dashed lines). A legend is included to aid in interpreting the plot (see Appendix B, Code B.2.3).

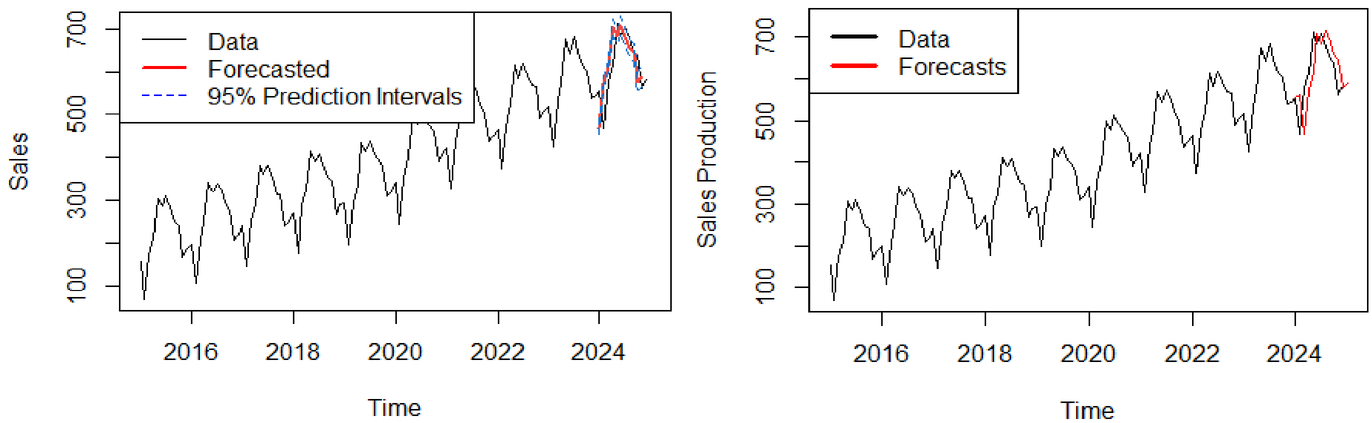


Figure 8: Monthly Sales Data with Forecast and Prediction Intervals Using Regression (Left) and DLM (Right)

4. Model Comparison

Metric	ARIMA	Time-Series Regression	Dynamic Linear Models (DLM)
Short-term Accuracy	High	Moderate	High
Long-term Accuracy	High	Low	High
Complexity	Moderate	Low	High
Interpretability	Moderate	High	Moderate
AIC/BIC	Lowest	N/A	Competitive

Table 3: Comparison of Model Performance Metrics

ARIMA demonstrated the lowest forecast error, showcasing strong reliability for short-term predictions by effectively capturing patterns in time series data. It strikes a balance between accuracy and moderate complexity, making it a preferred choice for accurate forecasts. In contrast, Time-Series Regression, while simple and interpretable, produced moderate errors due to its reliance on linear assumptions, limiting its robustness for handling complex seasonal fluctuations and its effectiveness for extended forecasting. Dynamic Linear Models (DLM) exhibited competitive performance, with slightly higher errors than ARIMA but demonstrated superior adaptability to structural changes. While DLM offers flexibility and versatility for various forecasting scenarios, this comes at a higher computational cost compared to ARIMA.

5. Conclusions

Sales data show an upward trend accompanied by consistent seasonality. Both ARIMA and DLM models prove reliable for forecasting, with ARIMA offering better accuracy and being more suitable for operational use. Time-series regression, while useful for explanatory analysis, is less effective for long-term predictions due to its simplicity and limited ability to capture complex patterns.

Recommendation: Given its accuracy and computational efficiency, ARIMA is recommended for operational forecasting and planning.

6. Appendix

6.1 Appendix A: Code and Plot of Modelling

Figure A.1: Differenced timeseries Data and ACF & PACF of Differenced Data

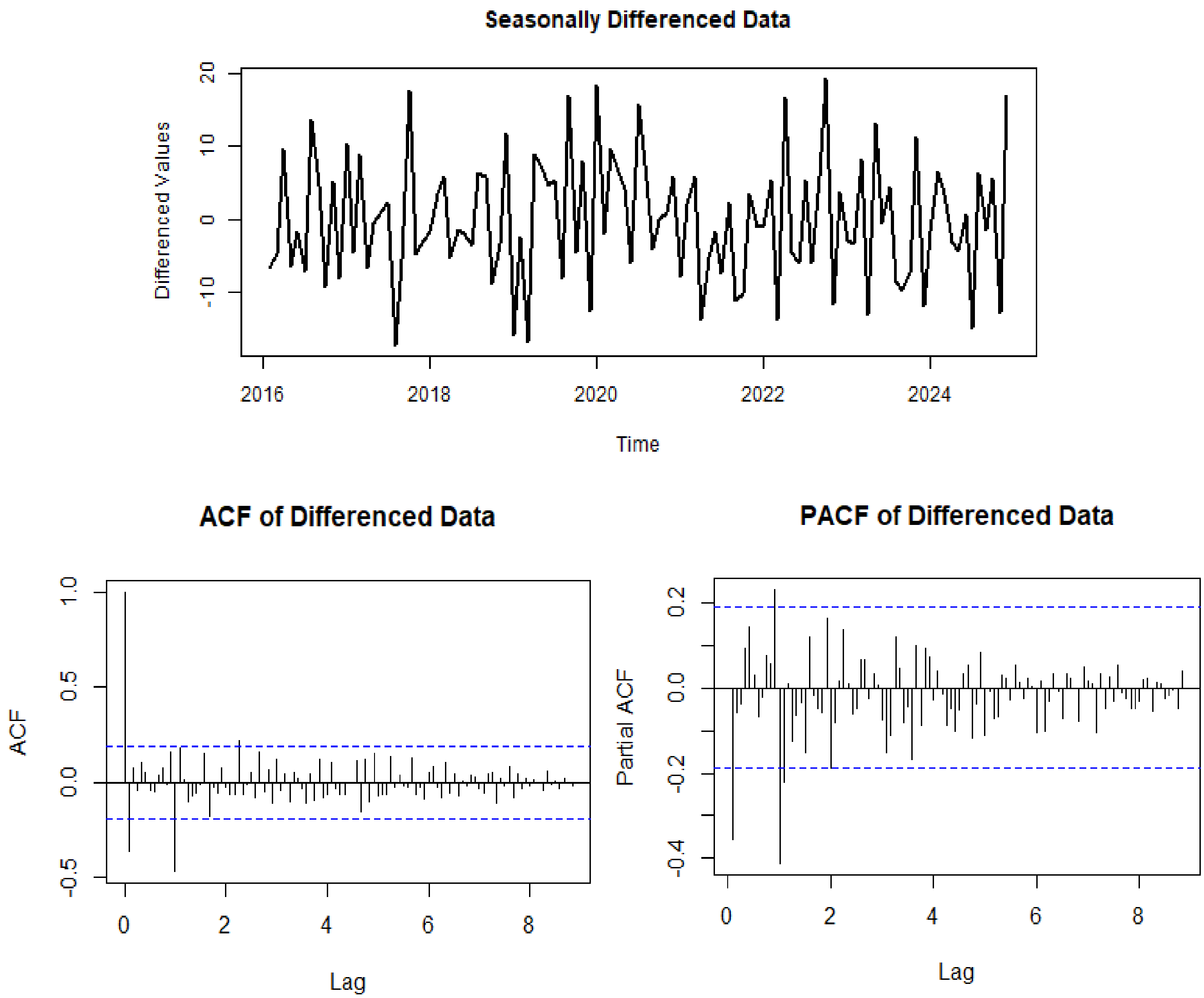


Figure A.2: SARIMA Model

```
## Coefficients:
##       Estimate     SE t.value p.value
## ar1    -0.3903 0.0926 -4.2163   1e-04
## sma1   -0.8528 0.1645 -5.1841   0e+00
##
## sigma^2 estimated as 36.03232 on 105 degrees of freedom
## AIC = 6.620741  AICc = 6.62182  BIC = 6.69568
```

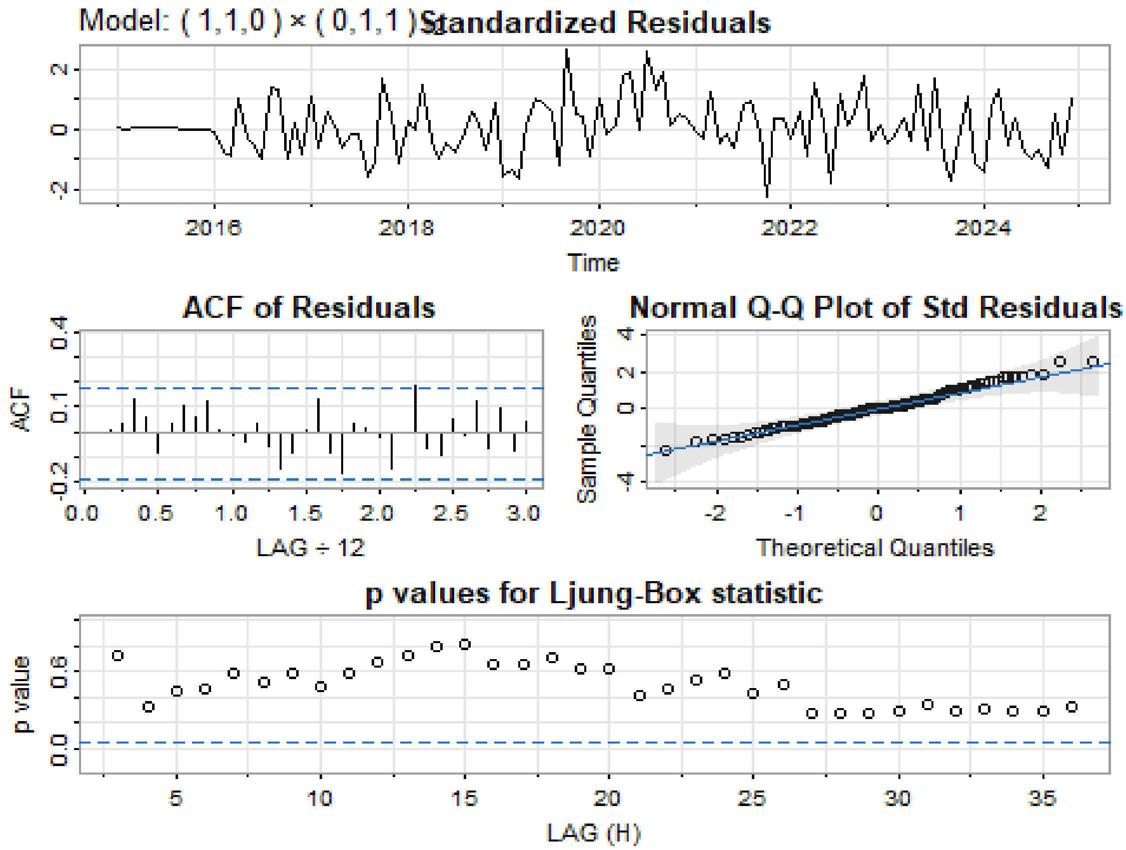
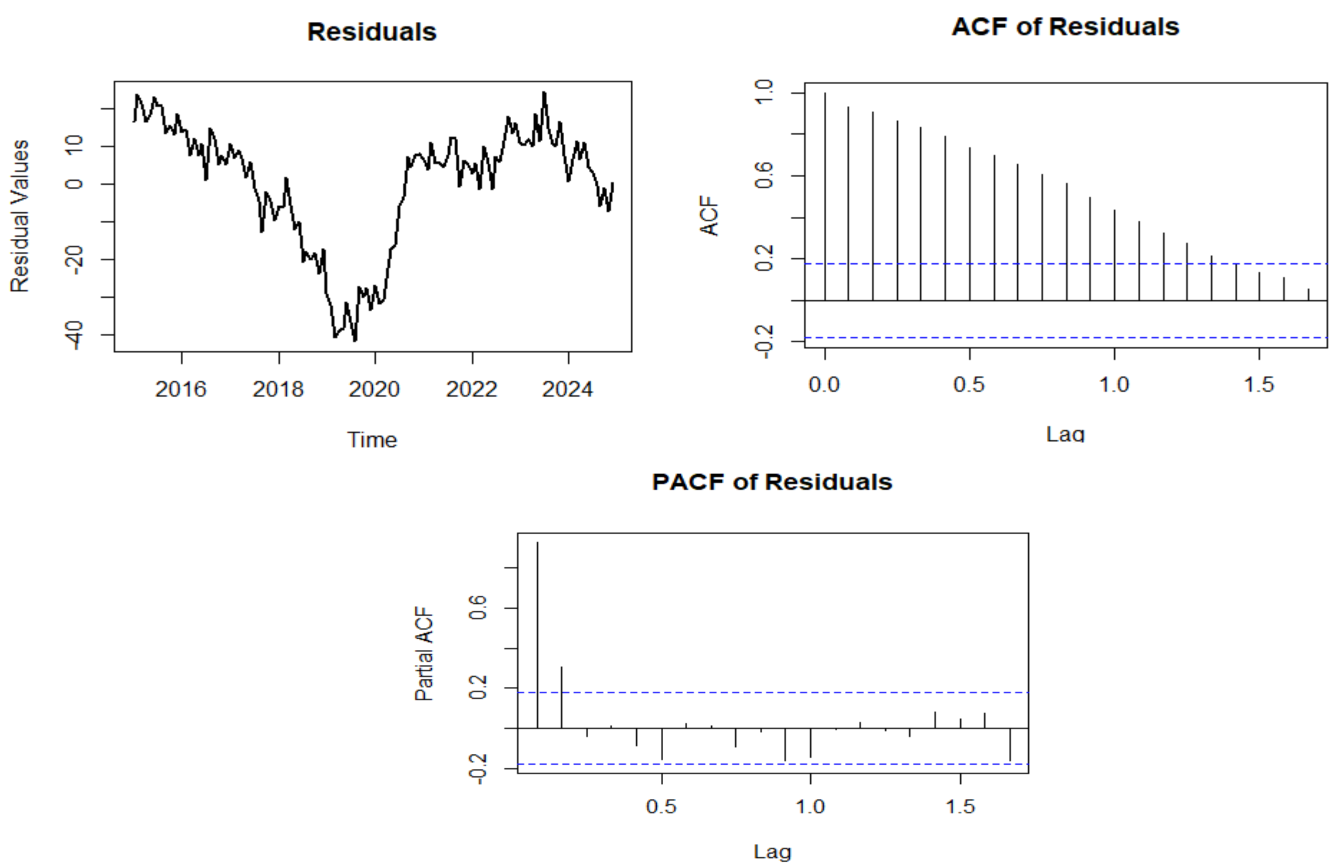


Figure A.3: Time-series regression modeling



Code A.1: Differenced timeseries Data and ACF & PACF of Differenced Data

```
library(astsa)
library(datasets)
library(forecast)
library(tseries)
library(dlm)

# Load and extract dataset
filename <- "projectdata.txt"
data <- read.table(filename, header = FALSE)
y <- data[, 3]
str(y)

# Visualize data
time_series <- ts(y, start = c(2015, 1), frequency = 12)
plot(time_series, main = "Monthly Sales (2015-2024)", ylab = "Sales", xlab = "Year",lw=2)

# Apply differencing to remove trend and seasonality
diff_ts <- diff(diff(time_series, lag = 12)) # Seasonal differencing
plot(diff_ts, main = "Seasonally Differenced Data", ylab = "Differenced Values",
xlab = "Time", col = "black", lw = 2)

# ACF and PACF plots
acf(diff_ts, main = "ACF of Differenced Data",lag.max = 1000)
pacf(diff_ts, main = "PACF of Differenced Data",lag.max = 1000)
# Hypothesis: Mean of differenced series is equal to 0
t_test_result <- t.test(diff_ts, mu = 0)
# Print the result
print(t_test_result)
```

Code A.2: ARIMA Model coding

```
# ARIMA Models
# Fit ARIMA model (ARIMA(1,1,0) Seasonal(0,1,0))
(model_ar <- arima(time_series, order = c(1, 1, 0), seasonal = c(0, 1, 0)))
## Extract residuals and plot residuals
resids = model_ar$residuals
plot(resids, ylab = "residuals")

# ACF and PACF of residuals
acf(resids, lag.max = 500)
pacf(resids)
# Fit ARIMA model (ARIMA(1,1,0) Seasonal(0,1,1))
(model_sma <- arima(time_series, order = c(1, 1, 0), seasonal = c(1, 1, 0)))
## Extract residuals and plot residuals
resids = model_sma$residuals
plot(resids, ylab = "residuals")

# ACF and PACF of residuals
acf(resids, lag.max = 500)
pacf(resids)
# Fit ARIMA model (ARIMA(1,1,0) Seasonal(0,1,1))
(model_arima = arima(time_series, order = c(1, 1, 0), seasonal = c(0, 1, 1)))
```

```
## Extract residuals and plot residuals
resids = model_arima$residuals
plot(resids, ylab = "residuals")

# ACF and PACF of residuals
acf(resids, lag.max = 500)
pacf(resids)
# Fit ARIMA model (ARIMA(2,1,0) Seasonal(0,1,1))
(model_2arima = arima(time_series, order = c(2, 1, 0), seasonal = c(0, 1, 1)))
## Extract residuals and plot residuals
resids = model_2arima$residuals
plot(resids, ylab = "residuals")

# ACF and PACF of residuals
acf(resids, lag.max = 500)
pacf(resids)
```

Code A.3: SARIMA model

```
model_sarima1 <- sarima(time_series, 2, 1, 0, 0, 1, 1, 12)
model_sarima <- sarima(time_series, 1, 1, 0, 0, 1, 1, 12)
```

Code A.4.1: Time-series regression modeling fitting

```
# Time-series regression modeling
time = c(1:120)
season = gl(12, 1, 120)
fit = lm(time_series ~ time + season)
# R-squared and adjusted R-squared
c(summary(fit)$r.squared, summary(fit)$adj.r.squared)

# Plot the fitted values overlaid on the original time series
fitted = ts(fitted.values(fit), start = c(2015, 1), frequency = 12)
plot(time_series, main = "Monthly Sales (2015-2024)", lw = 2)
lines(fitted, col = 2, lty = 2, lw = 3)

# Plot residuals from the time-series regression model
resid <- ts(residuals(fit), start = c(2015, 1), frequency = 12)
plot(resid, main = "Residuals", ylab = "Residual Values", xlab = "Time",
col = "black", lw = 2)
acf(resid, main = "ACF of Residuals")
pacf(resid, main = "PACF of Residuals")
```

Code A.4.2: Time-series regression modeling checking which is best fit

```
# Fit various ARIMA models for residuals
mod1 = arima(resid, order = c(1, 0, 1), include.mean = FALSE)
plot(mod1$residuals, ylab = "residuals")
acf(mod1$residuals)
pacf(mod1$residuals)
# Fit another ARIMA model for residuals (ARIMA(2,0,1))
mod2a = arima(resid, order = c(2, 0, 1), include.mean = FALSE)

# Compare log-likelihood between models
```

```
m1 = mod2a$loglik
m = mod1$loglik
(diff_pvalue <- m1 - m)
# p-value for difference in log-likelihood
1 - pchisq(diff_pvalue, df = 1)
```

Code A.5: Dynamic linear model (DLM)

```
# Dynamic linear model (DLM) using maximum likelihood estimation
build_sales = function(params) {
  dlmModPoly(order = 2, dV = exp(params[1]), dW = exp(params[2:3])) +
  dlmModTrig(s = 12, dV = 0, dW = exp(rep(params[4], 11)))
}
sales_fit = dlmMLE(time_series, parm = c(0, 0, 0, 0), build = build_sales)
sales_fit$convergence
# Display fitted model
sales_mod = build_sales(sales_fit$par)
str(sales_mod)
# Smooth the dynamic linear model and extract components
sales_smooth = dlmSmooth(time_series, sales_mod)
sales_level = sales_smooth$s[, 1]

x = cbind(time_series, dropFirst(sales_smooth$s[, 1]),
dropFirst(sales_smooth$s[, - (1:2)]) %*% t(sales_mod$FF)[-(1:2)])
colnames(x) = c("Production", "Trend", "Seasonal")
plot(x, type = "o", main = "Sales per Month")
```

6.2 Appendix B: Code and Plot of Forecasting

Code B.1: Future Prediction

Code B.1.1: ARIMA forecast for 6 months (January to June 2025)

```
# Generate ARIMA forecast for 6 months (January to June 2025)
forecast <- sarima.for(time_series, 6, 1, 1, 0, 0, 1, 1, 12)
```

Code B.1.2: Time-Series Regression Forecast

```
# ARIMA model for residuals
ar1 = arima(fit$resid, order=c(2,0,0))
ar1F = predict(ar1, n.ahead=6)
ar1F$pred = ts(ar1F$pred, start=c(2025, 1), frequency=12)

# Predict trend for next 6 months
predictedTrend = fit$coef[1] + fit$coef[2] * (121:126)
# Predict seasonal effects for next 6 months
Seffects = c(0, fit$coef[3:7])
# Combined trend + seasonality forecast
predTS = predictedTrend + Seffects

# Final forecast including residuals and trend + seasonality
ar1FT = ar1F$pred + predTS
ar1F$se = ts(ar1F$se, start=c(2025, 1), frequency=12)
```

```
# Upper and lower 95% prediction intervals
ar1FTU = ar1FT + 2 * ar1F$se
ar1FTL = ar1FT - 2 * ar1F$se

plot(time_series,xlim=c(2021,2026), lw=3)
lines(ar1FT,col=2,lty=2, lw=3)
lines(ar1FTU,col=4,lty=3, lw=2)
lines(ar1FTL,col=4,lty=3, lw=2)
```

Code B.1.3: Dynamic Linear Model Forecast

```
# Generate forecasts for DLM
sales_filt = dlmFilter(time_series, sales_mod)
sales_preds = dlmForecast(sales_filt, nAhead=6)

# Extract point forecasts and prediction interval limits
preds = sales_preds$f[,1]
lower = preds - 1.96 * sqrt(unlist(sales_preds$Q))
upper = preds + 1.96 * sqrt(unlist(sales_preds$Q))

sales_f = ts(c(time_series, upper), start=start(time_series),
frequency=frequency(time_series))
preds_f = ts(c(time_series[length(time_series)], preds),
start=end(time_series),frequency=frequency(time_series))
## Plot data, forecasts and prediction intervals
plot(sales_f, col="red", ylab="sales production", type="n")
lines(time_series, col="black")
lines(preds_f, col="red")
lines(lower, lty=2, col="blue")
lines(upper, lty=2, col="blue")
legend("topleft", c("data", "forecasts", "95% prediction intervals"),
col=c("black", "red", "blue"), lty=c(1, 1, 2))
```

Code B.2: Reliability of Forecasting

Code B.2.1: ARIMA forecast for the last 12 months

```
##Reliability of Forecasting
# Generate ARIMA forecast for the last 12 months (January to December 2024)
forecast <- sarima.for(ts(time_series[1:(length(time_series) - 12)]),
12, 1, 1, 0, 0, 1, 1, 12)
# Plot the actual time series with forecasted values
lines(ts(time_series[1:length(time_series)]), col='blue')
```

Code B.2.2: Regression model Fit for the last 12 months

```
time = c(1:108)
season = factor(rep(1:12, 9))
# Fit a linear model to the time series (trend + seasonality)
fit1 = lm(time_series[1:108] ~ time + season)

# Fit an ARIMA model to the residuals of the linear model
ar11 = arima(fit1$resid, order=c(2,0,0))
```

```
ar11F = predict(ar11, n.ahead=12)
ar11F$pred = ts(ar11F$pred, start=c(2024, 1), frequency=12)

predictedTrend1 = fit1$coef[1] + fit1$coef[2]*(109:120)
Seffects1 = fit1$coef[3:14]  # The seasonal coefficients (12 months)
predTS1 = predictedTrend1 + Seffects1
# Combine the forecast with the trend and seasonal components
ar11FT = ar11F$pred + predTS1
ar11F$se = ts(ar11F$se, start=c(2024, 1), frequency=12)
ar11FTU = ar11FT + 2*ar11F$se  # Upper bound
ar11FTL = ar11FT - 2*ar11F$se  # Lower bound

# Plot the original data (Jan 2015 - Dec 2024),
forecasted values, and prediction intervals
plot(time_series, xlim=c(2015, 2025), type='l', ylab='Sales',
xlab='Time', col='black')
lines(ar11FT, col=2, lwd = 2)
lines(ar11FTU, col=4, lty=2)
lines(ar11FTL, col=4, lty=2)
legend("topleft", c("Data", "Forecasted", "95% Prediction Intervals"),
col=c("black", "red", "blue"), lty=c(1, 1, 2), lwd=c(1, 2, 1))
```

Code B.2.3: DLM model fit for the last 12 months

```
# Fit the model using the first 9 years (Jan 2015 - Dec 2023)
sales_filt = dlmFilter(time_series[1:108], sales_mod)

# Forecast the next 24 months (Jan 2024 - Dec 2024)
sales_preds = dlmForecast(sales_filt, nAhead=12)

# Extract point forecasts (predicted values) and prediction interval limits
preds = sales_preds$f[, 1]  # Point forecasts
lower = preds - 1.96 * sqrt(unlist(sales_preds$Q))  # Lower bound
upper = preds + 1.96 * sqrt(unlist(sales_preds$Q))  # Upper bound

# Combine the original data (Jan 2015 - Dec 2023)
and forecasted values (Jan 2024 - Dec 2024)
sales_f = ts(c(time_series[1:108], upper), start=start(time_series),
frequency=frequency(time_series))
preds_f = ts(c(time_series[109], preds), start=c(2024, 1),
frequency=frequency(time_series))

# Plot the original data, forecasted values, and prediction intervals
plot(sales_f, col="red", ylab="Sales Production", type="n",
xlim=c(2015, 2025))  # Set xlim to span 2015-2025
lines(time_series, col="black")  # Original data line (Jan 2015 to Dec 2023)
lines(preds_f, col="red")  # Forecast line (Jan 2024 to Dec 2024)
lines(lower, lty=2, col="blue")  # Lower bound of prediction intervals
lines(upper, lty=2, col="blue")  # Upper bound of prediction intervals

# Add a legend to the plot
legend("topleft", c("Data", "Forecasts"),
       col=c("black", "red"), lty=c(1, 1), lwd=c(3, 3))
```