DATA 624 Project 1

Warner Alexis

2025-03-30

Project 1

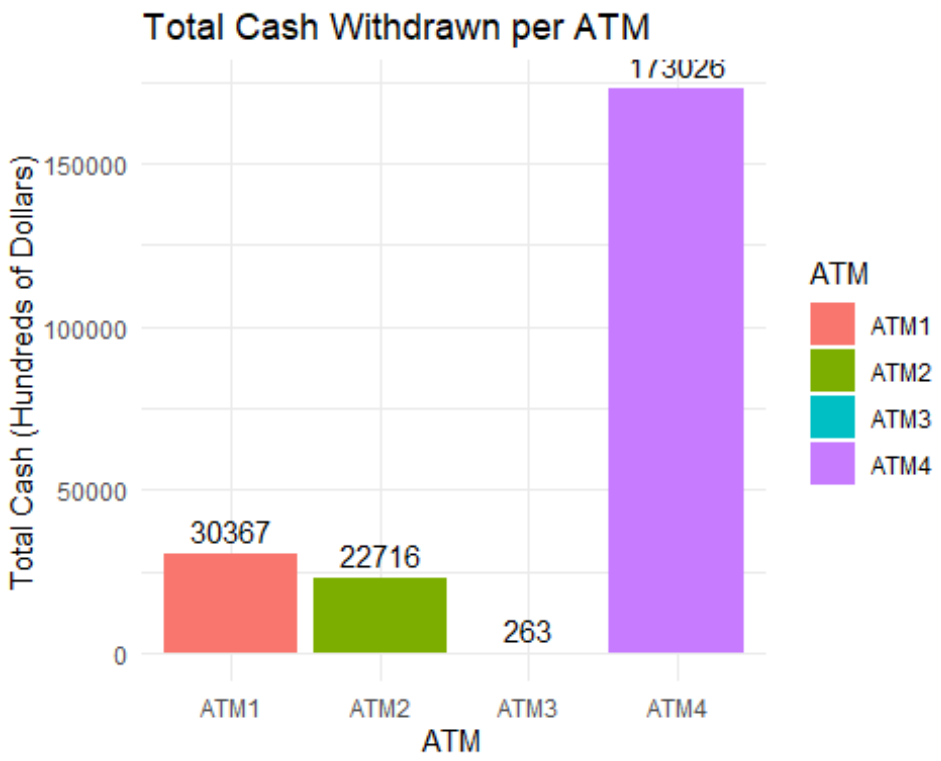
This project consists of 3 parts - two required and one bonus and is worth 15% of your grade. The project is due at 11:59 PM on Sunday Oct 25. I will accept late submissions with a penalty until the meetup after that when we review some projects.

ATM Forecast The bar graph illustrates the total cash withdrawals (in hundreds of dollars) for each ATM machine across the dataset. ATM4 stands out with the highest total cash withdrawn at **173,026**, followed by ATM1 and ATM2 with **30,367** and **22,716** respectively. ATM3, on the other hand, shows a significantly lower total of just **263**, suggesting it was either inactive or only recently deployed during the data collection period. The underlying dataset contains **3 columns** (`DATE`, `ATM`, `Cash`) and **1,474 rows**, representing individual withdrawal records across different dates and machines. This analysis helps identify usage intensity and transaction volume per ATM, which is essential for forecasting and operational planning.

```
# Loading Libraries
library(readxl)
library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(ggplot2)
library(lubridate)

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

library(forecast)
```

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

library(tidyr)
library(tsibble)

## Registered S3 method overwritten by 'tsibble':
##   method               from
##   as_tibble.grouped_df dplyr

##
## Attaching package: 'tsibble'

## The following object is masked from 'package:lubridate':
##
##     interval

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, union

library(feasts)

## Loading required package: fabletools

library(fable)
library(urca)
library(writexl)

df <- read_excel("ATM624Data.xlsx", sheet = "ATM Data")

# change df timestamps because of the way excell interpret it
df <- df %>% mutate(DATE = as.Date(DATE, origin = "1899-12-30"))
head(df)

## # A tibble: 6 × 3
##   DATE       ATM    Cash
##   <date>     <chr> <dbl>
## 1 2009-05-01 ATM1     96
## 2 2009-05-01 ATM2    107
## 3 2009-05-02 ATM1     82
## 4 2009-05-02 ATM2     89
## 5 2009-05-03 ATM1     85
## 6 2009-05-03 ATM2     90

# Summarize total cash per ATM with NA handled
atm_totals <- df |>
  group_by(ATM) |>
  summarise(total_cash = sum(Cash, na.rm = TRUE)) |>
  filter(!is.na(ATM))  # Remove any NA ATM labels
```

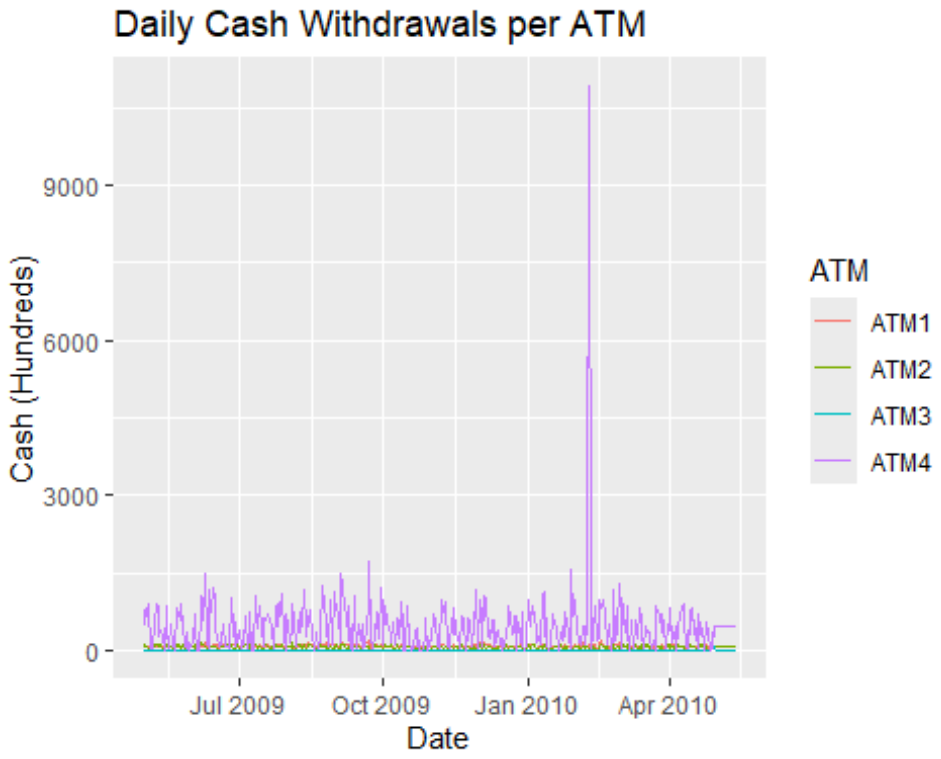
```r
# Create bar plot
ggplot(atm_totals, aes(x = ATM, y = total_cash, fill = ATM)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = round(total_cash, 0)), vjust = -0.5) +
  labs(title = "Total Cash Withdrawn per ATM",
       x = "ATM",
       y = "Total Cash (Hundreds of Dollars)") +
  theme_minimal()
```



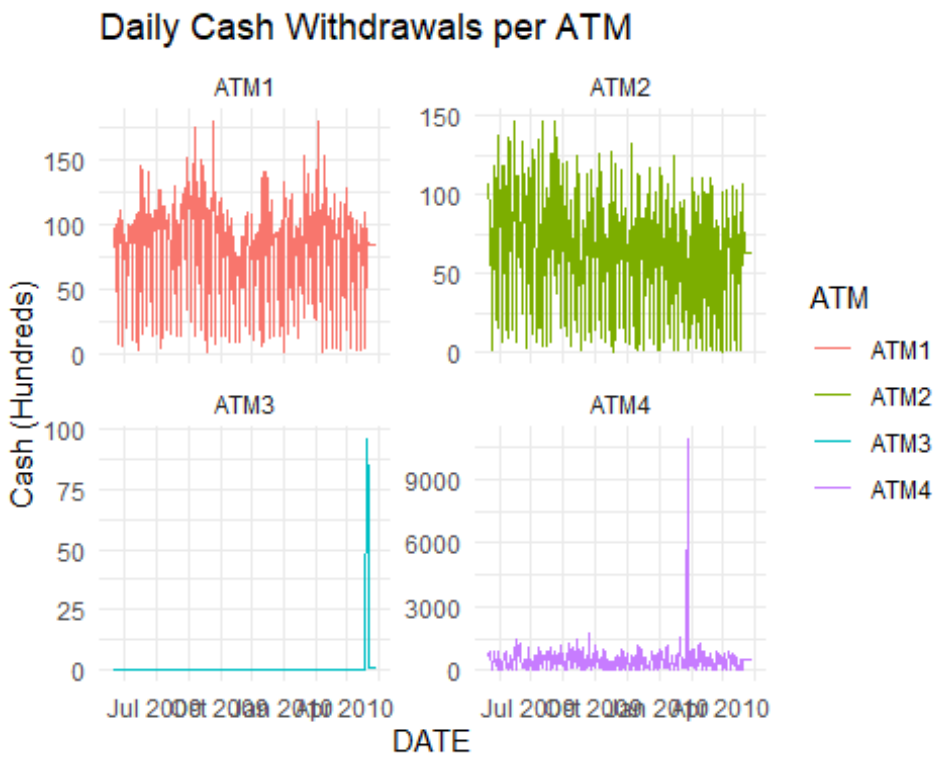
we first created a complete daily date sequence and merged it with our ATM withdrawal data to ensure no days are missing. Any missing values (i.e., days with no withdrawals) were filled with mean values for each ATM. Then, we reshaped the data from wide to long format and converted it into a tsibble, which is required for time series plotting using the autoplot() function. Finally, we generated a faceted time series plot with individual y-axis scales for each ATM, allowing us to clearly visualize daily withdrawal patterns across all machines.

```r
# Ensure complete time series per ATM
data_wide <- df %>%
  group_by(ATM, DATE) %>%
  summarise(Cash = sum(Cash), .groups = 'drop') %>%
  tidyr::pivot_wider(names_from = ATM, values_from = Cash)
# Fill in missing days with 0s (if needed)
full_dates <- seq(min(df$DATE), max(df$DATE), by = "day")
atm_means <- data_wide %>%
  summarise(
```

```
    ATM1 = mean(ATM1, na.rm = TRUE),
    ATM2 = mean(ATM2, na.rm = TRUE),
    ATM3 = mean(ATM3, na.rm = TRUE),
    ATM4 = mean(ATM4, na.rm = TRUE)
  )

# Fill missing values with those means
data_full <- data.frame(DATE = full_dates) %>%
  left_join(data_wide, by = "DATE") %>%
  tidyr::replace_na(list(
    ATM1 = atm_means$ATM1,
    ATM2 = atm_means$ATM2,
    ATM3 = atm_means$ATM3,
    ATM4 = atm_means$ATM4
  ))

# Convert to long format and tsibble
data_long <- data_full %>%
  pivot_longer(cols = starts_with("ATM"), names_to = "ATM", values_to =
"Cash") %>%
  as_tsibble(index = DATE, key = ATM)

# Plot using autoplot
autoplot(data_long, Cash) +
  labs(title = "Daily Cash Withdrawals per ATM", y = "Cash (Hundreds)", x =
"Date")
```



```r
# Plot with autoplot and free scales
autoplot(data_long, Cash) +
  facet_wrap(~ATM, scales = "free_y") +
  labs(title = "Daily Cash Withdrawals per ATM",
       x = "DATE",
       y = "Cash (Hundreds)") +
  theme_minimal()
```



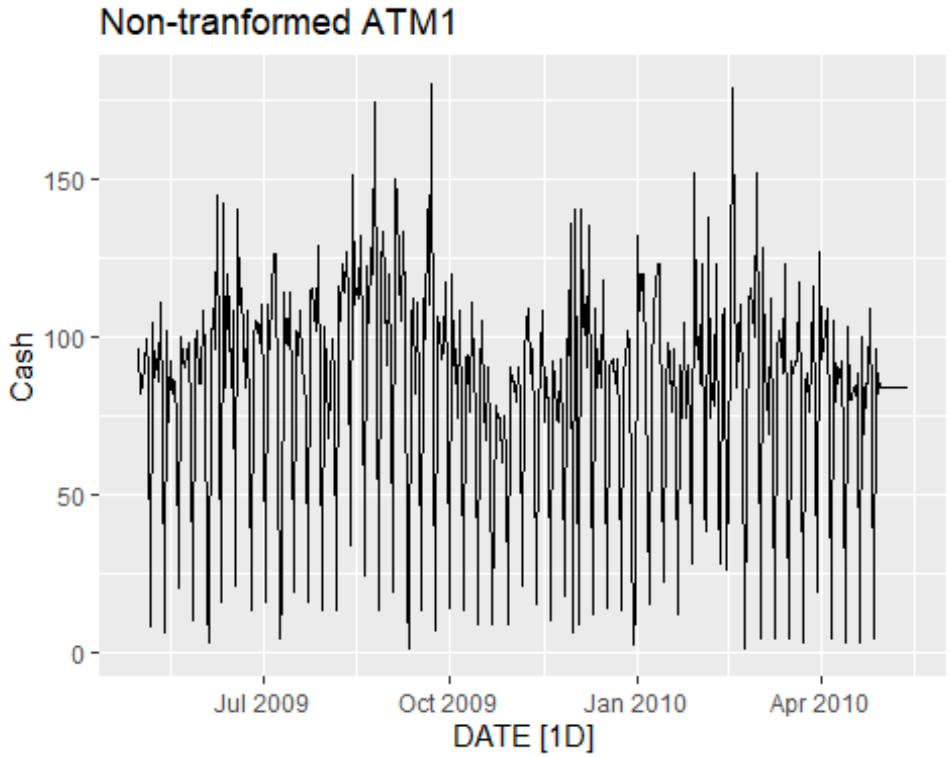
ATM Forecast

ATM1

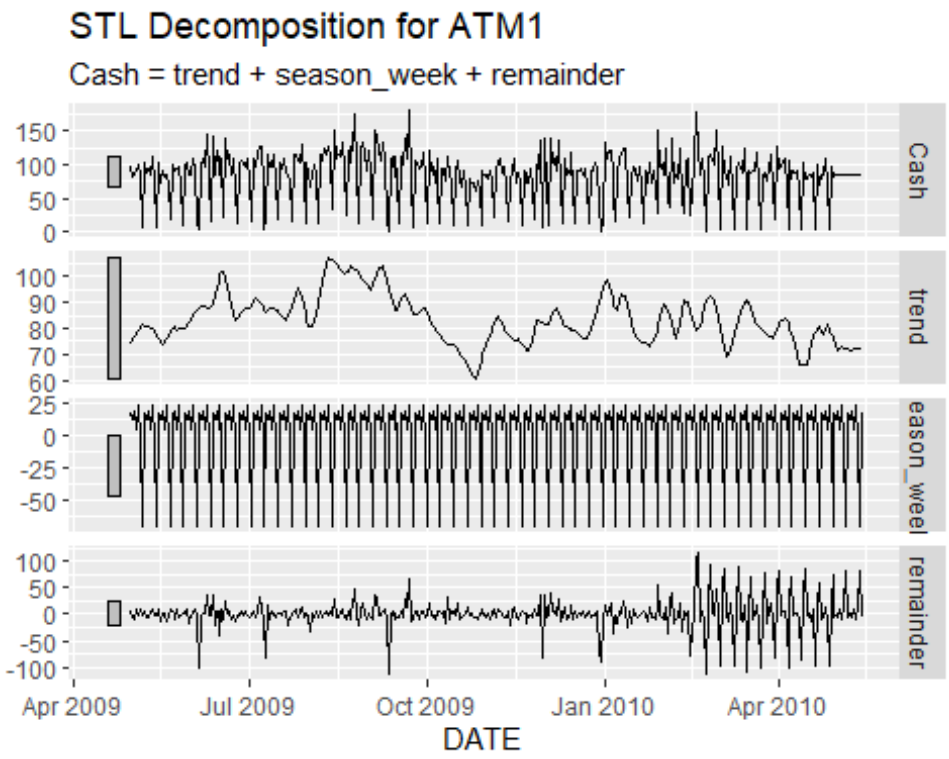
The Augmented Dickey-Fuller (ADF) test results indicate that the differenced time series is stationary. The test statistic (tau) is -23.331, which is far below the 1% critical value of -3.44. This suggests strong evidence against the null hypothesis of a unit root, meaning the time series does not have a unit root and is therefore stationary. Additionally, the p-value associated with the test is less than 0.01 (often reported as `< 2.2e-16`), which reinforces this conclusion. A low p-value (typically < 0.05) indicates that the observed data is highly inconsistent with the assumption of non-stationarity. Therefore, we reject the null hypothesis and conclude that the differenced series is stationary, making it suitable for time series modeling approaches that assume stationarity.

```r
# Create tsibble
ts_data <- data_long %>%
  as_tsibble(index = DATE, key = ATM)
```

```r
# for atm1

ts_data %>%
  filter(ATM == "ATM1") %>%
  autoplot(Cash) +
  ggtitle("Non-tranformed ATM1")
```




```r
ts_data %>%
  filter(ATM == "ATM1") %>%
  model(STL(Cash ~ season(window = "periodic"), robust = TRUE)) %>%
  components() %>%
  autoplot() +
  labs(title = "STL Decomposition for ATM1")
```




```r
ts_data <- ts_data %>% select(-`NA`)
# STL decomposition
stl_model <- ts_data %>%
  filter(ATM == "ATM1") %>%
  model(STL(Cash ~ season(window = "periodic")))

# Extract components
components <- components(stl_model)

# Check if remainder is stationary (Augmented Dickey-Fuller test)

adf_test <- ur.df(components$remainder, type = "drift", selectlags = "AIC")
summary(adf_test)
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression drift
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)
##
## Residuals:
```

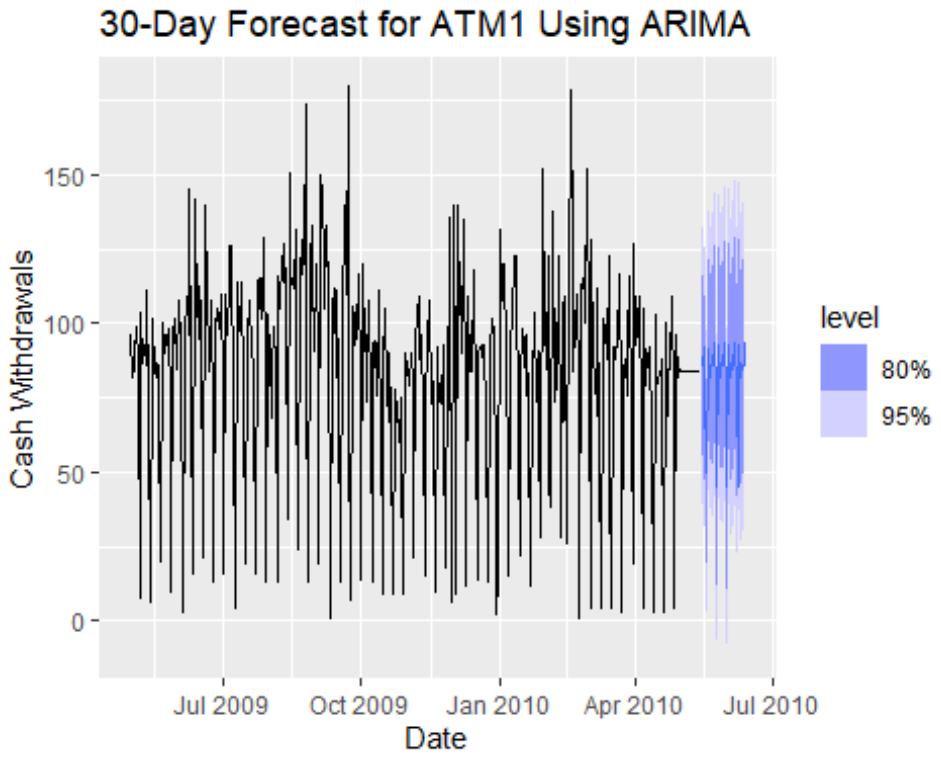
```
##     Min      1Q  Median      3Q     Max
## -87.595  -9.177   0.089  12.061  76.842
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.01938    1.17240  -0.017    0.987
## z.lag.1     -1.58670    0.06801 -23.331   <2e-16 ***
## z.diff.lag   0.45778    0.04623   9.902   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.76 on 374 degrees of freedom
## Multiple R-squared:  0.6389, Adjusted R-squared:  0.637
## F-statistic: 330.9 on 2 and 374 DF,  p-value: < 2.2e-16
##
##
## Value of test-statistic is: -23.331 272.1682
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau2 -3.44 -2.87 -2.57
## phi1  6.47  4.61  3.79
```

```r
# Step 1: Filter data for ATM1
atm1_tsibble <- ts_data %>%
  filter(ATM == "ATM1")

# Step 2: Fit ARIMA model (automatically selected orders)
arima_model <- atm1_tsibble %>%
  model(ARIMA(Cash))

# Step 3: Model report
report(arima_model)
```

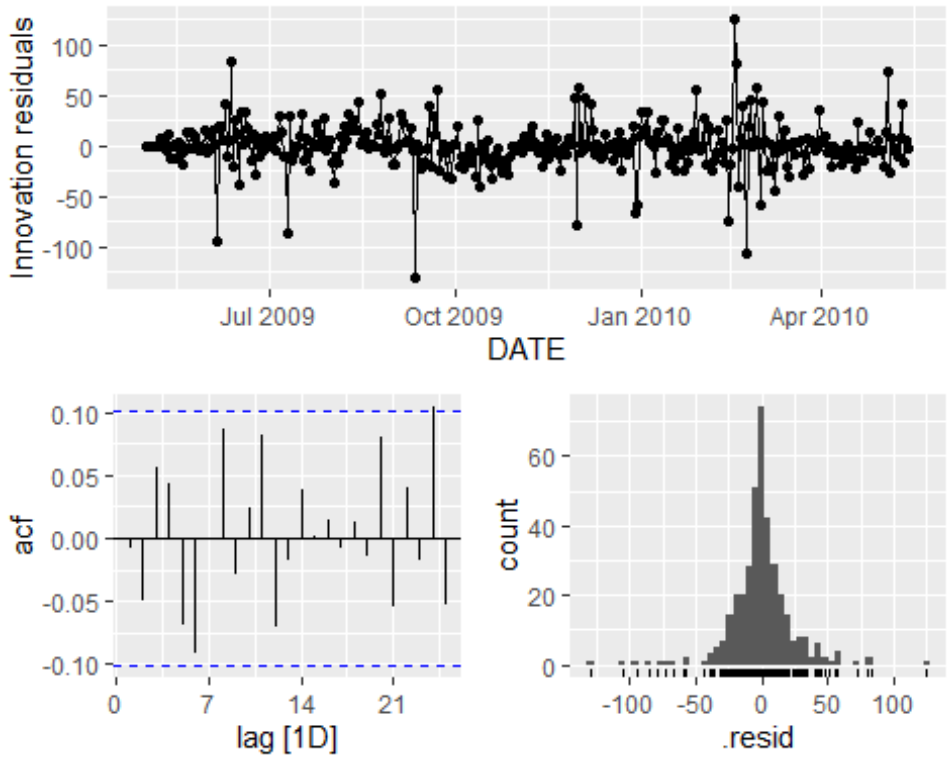
```
## Series: Cash
## Model: ARIMA(0,0,1)(0,1,2)[7]
##
## Coefficients:
##          ma1     sma1     sma2
##       0.1543  -0.5794  -0.1216
## s.e.  0.0548   0.0500   0.0510
##
## sigma^2 estimated as 566.4:  log likelihood=-1707.56
## AIC=3423.12   AICc=3423.23   BIC=3438.8
```

```r
# Step 4: Forecast 30 days ahead
forecast_arima <- arima_model %>%
  forecast(h = "30 days")

# Step 5: Plot forecast with historical data
```

```r
autoplot(forecast_arima, atm1_tsibble) +
  labs(title = "30-Day Forecast for ATM1 Using ARIMA",
       y = "Cash Withdrawals", x = "Date")
```



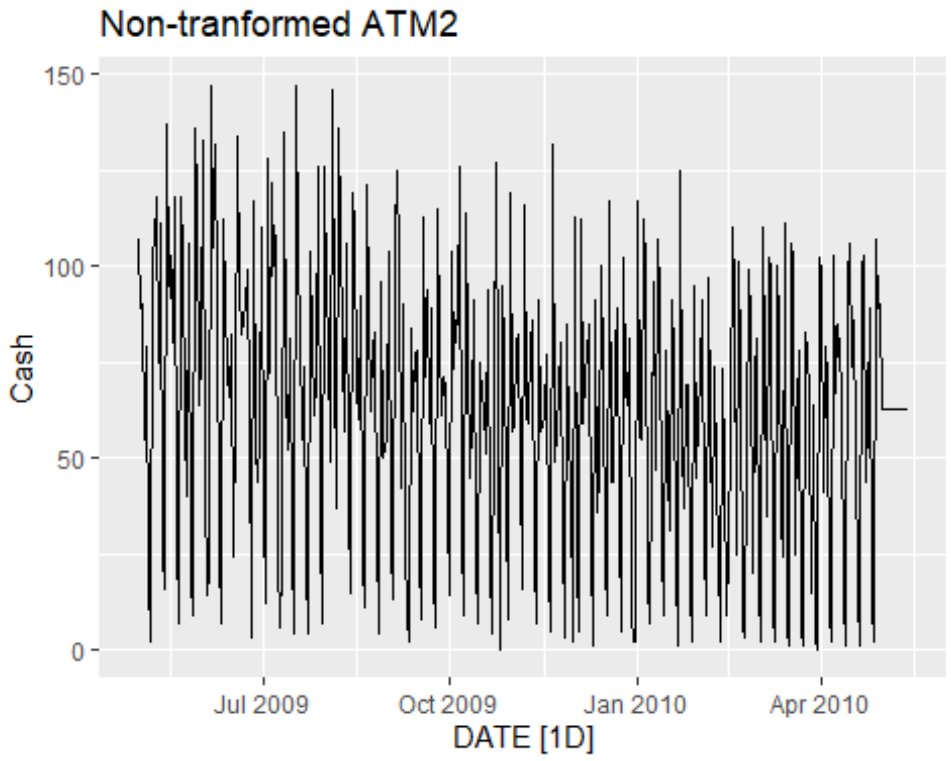

```r
# Step 6: Residual diagnostics
# a) ACF plot of residuals
arima_model %>%
  gg_tsresiduals()
```



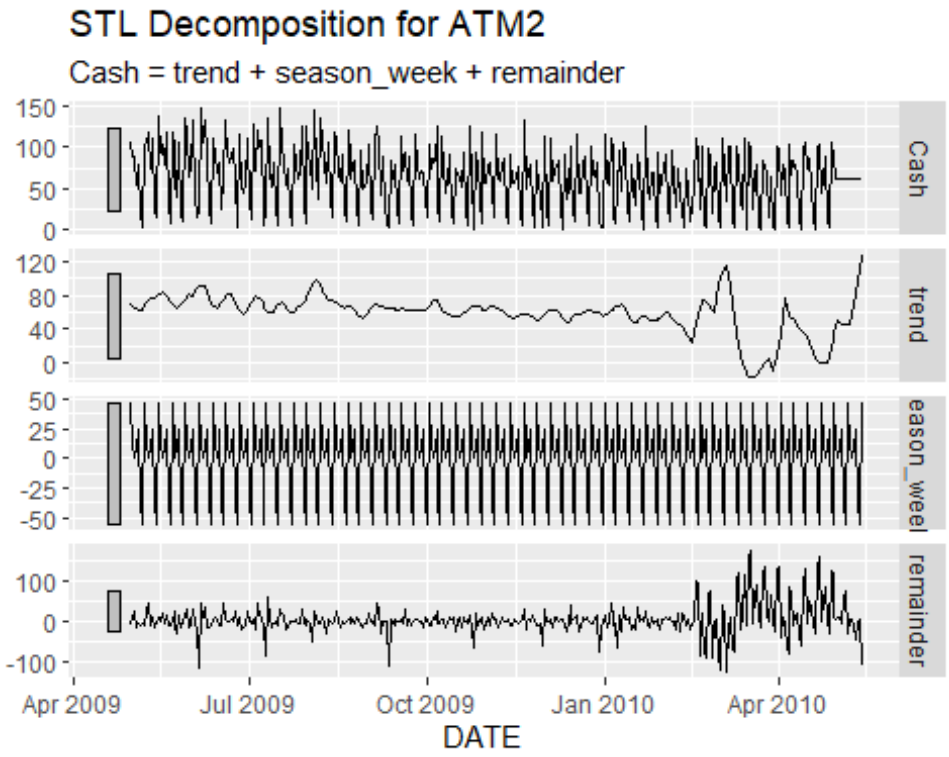

```
# b) Ljung-Box test for white noise residuals
arima_model %>%
  augment() %>%
  features(.innov, ljung_box, lag = 24, dof = 3)

## # A tibble: 1 × 4
##   ATM   .model      lb_stat lb_pvalue
##   <chr> <chr>         <dbl>     <dbl>
## 1 ATM1  ARIMA(Cash)    26.2     0.200

# extract point from forecast only
forecast_atm1 <- forecast_arima %>%
  as_tibble() %>%
  select(DATE, ATM, .mean) %>%
  pivot_wider(names_from = ATM, values_from = .mean)

# Save to Excel
write_xlsx(forecast_atm1, "ATM1_Forecast_May2010.xlsx")
```

ATM2

```
ts_data %>%
  filter(ATM == "ATM2") %>%
  autoplot(Cash) +
  ggtitle("Non-tranformed ATM2")
```




```r
ts_data %>%
  filter(ATM == "ATM2") %>%
  model(STL(Cash ~ season(window = "periodic"), robust = TRUE)) %>%
  components() %>%
  autoplot() +
  labs(title = "STL Decomposition for ATM2")
```



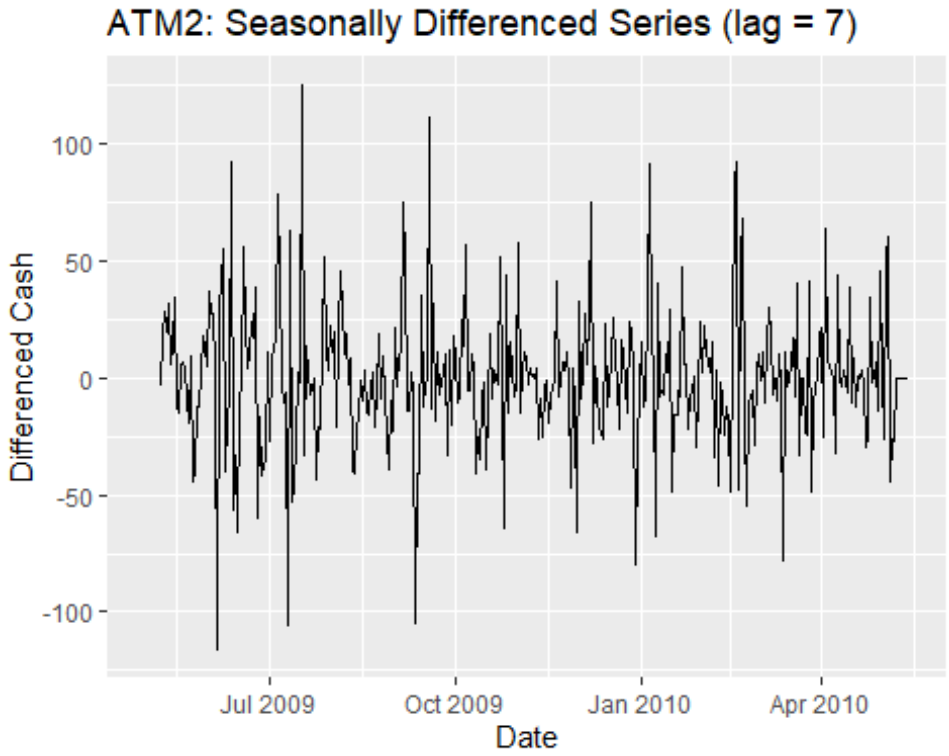

```
# --- 1. Extract ATM2 from data_long ---
atm2_data <- ts_data %>%
  filter(ATM == "ATM2")

# --- 2. Check if it's stationary (ADF test) ---
atm2_data %>% features(Cash, unitroot_kpss)

## # A tibble: 1 × 3
##   ATM   kpss_stat kpss_pvalue
##   <chr>     <dbl>       <dbl>
## 1 ATM2       2.07        0.01

# --- 3. Differencing to remove trend and seasonality ---
# First difference: removes trend
# Seasonal difference: removes weekly seasonality (lag = 7)
atm2_stationary <- atm2_data %>%
  mutate(diff_cash = difference(Cash, differences = 1),
         diff_seasonal = difference(Cash, lag = 7))
```

The null hypothesis of the KPSS test is that the series is stationary Since the p-value is 0.01 , we reject the null hypothesis. the unitroop_kpps test confirms that the ATM2 series is non-stationary and requires differencing before modeling.

These plots help you visually inspect if: - Residuals are approximately normally distributed - There's no autocorrelation left (ACF spikes should be within the blue bands) - The model is well-behaved (i.e., white noise residuals)

The Ljung-Box test was conducted on the residuals of the ARIMA model fitted to ATM2 in order to assess whether any autocorrelation remained after model fitting. The test returned a statistic of 8.47 with a p-value of 0.583 at lag 10. Since the p-value is greater than the conventional threshold of 0.05, we fail to reject the null hypothesis that the residuals are independently distributed. This indicates that the residuals behave like white noise and do not exhibit significant autocorrelation, suggesting that the ARIMA model for ATM2 provides an adequate fit to the data.

```
# Plot transformed (stationary) series
atm2_stationary %>%
  autoplot(diff_seasonal, na.rm = TRUE) +
  labs(title = "ATM2: Seasonally Differenced Series (lag = 7)",
       y = "Differenced Cash", x = "Date")
```




```
# Apply first-order differencing
atm2_diff <- atm2_data %>%
  mutate(diff_cash = difference(Cash, differences = 1)) %>%
  filter(!is.na(diff_cash))

# Test again after differencing
atm2_diff %>% features(diff_cash, unitroot_kpss)
```

```
## # A tibble: 1 × 3
##   ATM   kpss_stat kpss_pvalue
##   <chr>     <dbl>       <dbl>
## 1 ATM2     0.0104         0.1
```

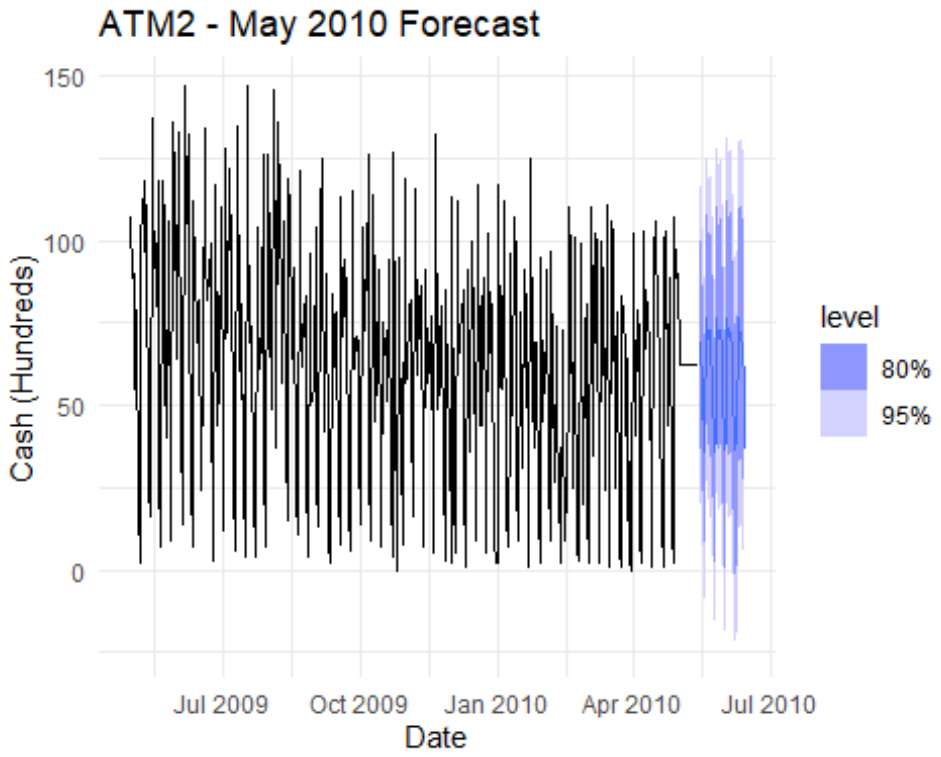
```r
atm2_model <- atm2_data %>%
  model(ARIMA = ARIMA(Cash))

report(atm2_model)
```

```
## Series: Cash
## Model: ARIMA(2,0,2)(0,1,1)[7]
##
## Coefficients:
##           ar1      ar2     ma1     ma2     sma1
##       -0.4380  -0.9154  0.4921  0.8017  -0.7701
## s.e.   0.0525   0.0383  0.0814  0.0560   0.0392
##
## sigma^2 estimated as 602.2:  log likelihood=-1718.39
## AIC=3448.78   AICc=3449.01   BIC=3472.29
```

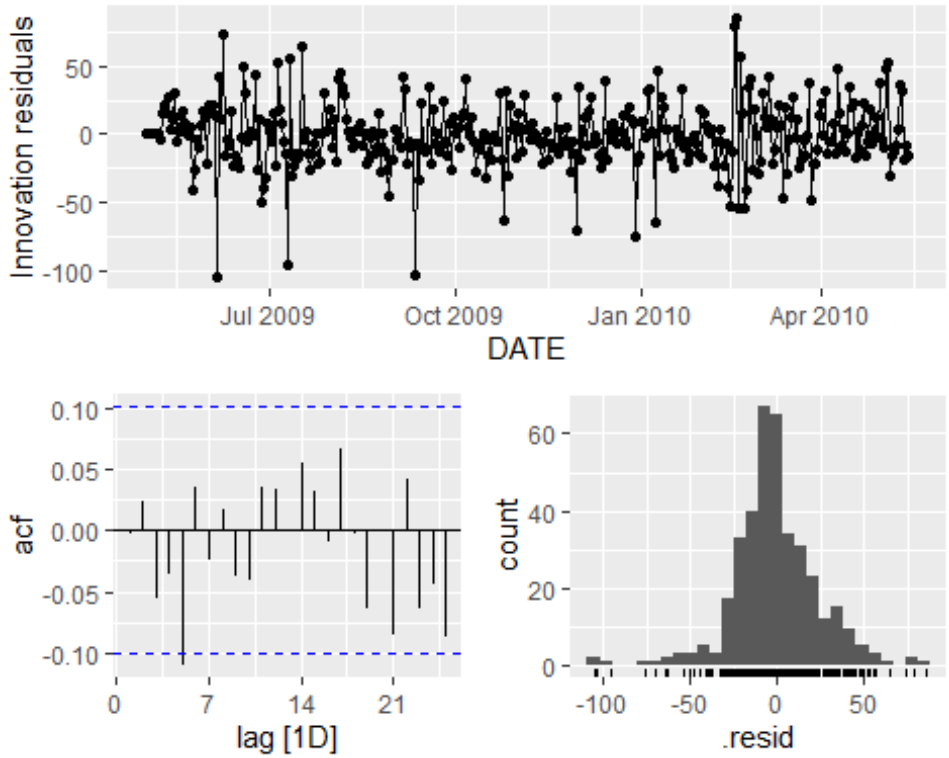
```r
atm2_model <- atm2_data %>%
  model(ARIMA = ARIMA(Cash))

report(atm2_model)
```

```
## Series: Cash
## Model: ARIMA(2,0,2)(0,1,1)[7]
##
## Coefficients:
##           ar1      ar2     ma1     ma2     sma1
##       -0.4380  -0.9154  0.4921  0.8017  -0.7701
## s.e.   0.0525   0.0383  0.0814  0.0560   0.0392
##
## sigma^2 estimated as 602.2:  log likelihood=-1718.39
## AIC=3448.78   AICc=3449.01   BIC=3472.29
```

```r
atm2_forecast <- atm2_model %>%
  forecast(h = "31 days")

autoplot(atm2_forecast, atm2_data) +
  labs(title = "ATM2 - May 2010 Forecast",
       y = "Cash (Hundreds)", x = "Date") +
  theme_minimal()
```



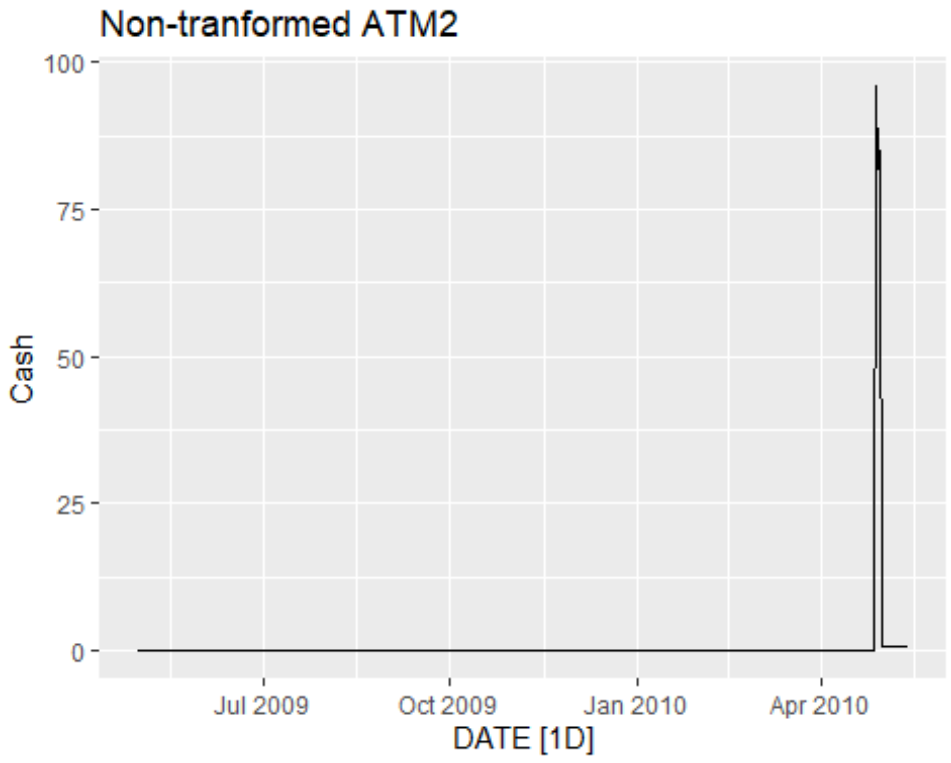

```
atm2_model %>%
  gg_tsresiduals()
```



```
atm2_model %>%
  augment() %>%
  features(.resid, ljung_box, lag = 10)

## # A tibble: 1 × 4
##   ATM   .model lb_stat lb_pvalue
##   <chr> <chr>    <dbl>     <dbl>
## 1 ATM2  ARIMA     8.47     0.583

# print the files
forecast_atm2 <- atm2_forecast %>%
  as_tibble() %>%
  select(DATE, ATM, .mean) %>%
  pivot_wider(names_from = ATM, values_from = .mean)

# Save to Excel
write_xlsx(forecast_atm2, "ATM2_Forecast_May2010.xlsx")
```

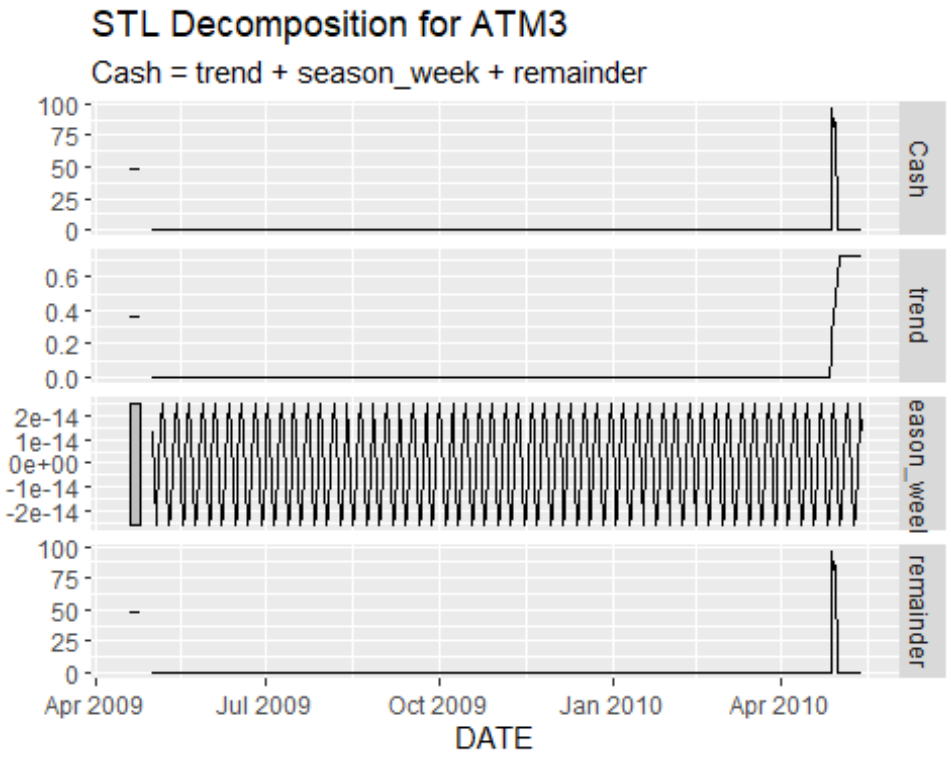
ATM3

```
ts_data %>%
  filter(ATM == "ATM3") %>%
  autoplot(Cash) +
  ggtitle("Non-tranformed ATM2")
```




```
ts_data %>%
  filter(ATM == "ATM3") %>%
  model(STL(Cash ~ season(window = "periodic"), robust = TRUE)) %>%
```

```
  components() %>%
  autoplot() +
  labs(title = "STL Decomposition for ATM3")
```




```
# --- 1. Extract ATM2 from data_long ---
atm3_data <- ts_data %>%
  filter(ATM == "ATM3")

# --- 2. Check if it's stationary (ADF test) ---
atm3_data %>% features(Cash, unitroot_kpss)
```

```
## # A tibble: 1 × 3
##   ATM   kpss_stat kpss_pvalue
##   <chr>     <dbl>       <dbl>
## 1 ATM3      0.375      0.0880
```

Based on the grapgh, ATM3 appear to be new and lack historical pattern. The KPSS test was conducted on the ATM3 cash withdrawal time series to evaluate its stationarity. The test yielded a KPSS statistic of 0.375 with a corresponding p-value of 0.088. Since the p-value is greater than the conventional significance level of 0.05, we fail to reject the null hypothesis that the series is stationary. This indicates there is no strong evidence of a unit root, and the data can be considered stationary. Therefore, based on the KPSS test results, the ATM3 time series does not exhibit significant non-stationarity and may be suitable for modeling without further differencing.

This ARIMA(0,0,2) model assumes the data is already stationary and captures the short-term autocorrelation using two moving average components. The model fits the data well, as indicated by the significant MA coefficients and relatively low AIC. It's a suitable model for forecasting if the residual diagnostics confirm white noise behavior.

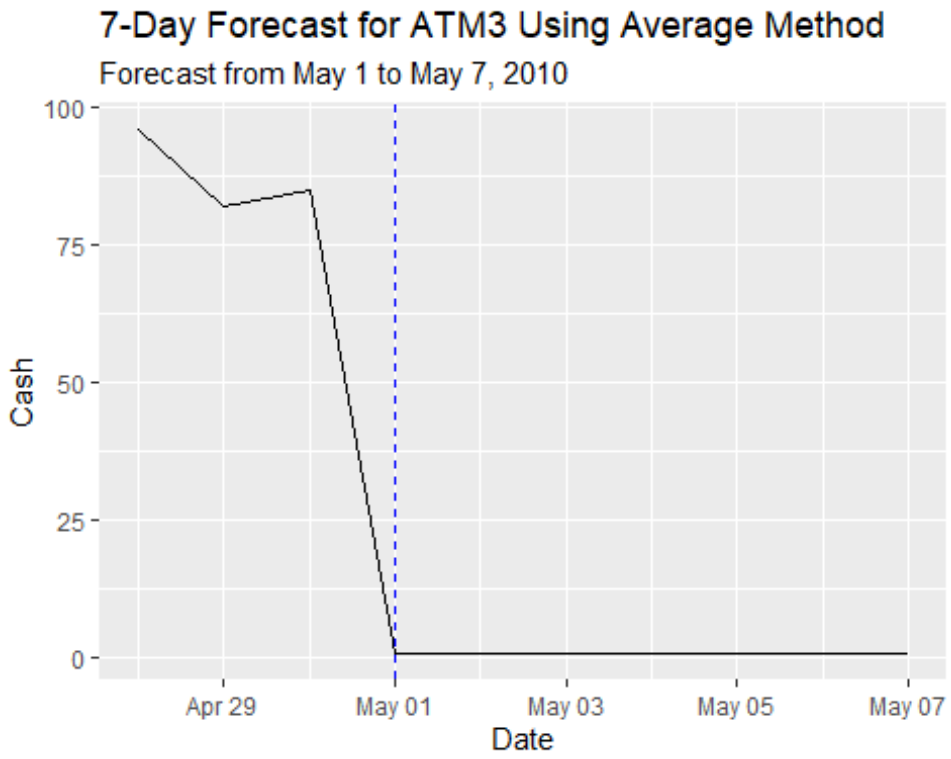
In Order to make this model better, we need more historical data. This prediction is only based on 3 days. Although an ARIMA(0,0,2) model was successfully fitted to the ATM3 cash withdrawal series, several factors indicate that the model should not be relied upon for meaningful forecasting. The historical data for ATM3 is extremely limited, with most values either zero or constant imputed values, and only a few days reflecting real activity. While the KPSS test suggests the series is stationary (p = 0.088), the residual diagnostics show a lack of meaningful variation, with nearly all residuals close to zero and a single large spike occurring near the end. This pattern suggests the model is likely overfitting a small number of recent values rather than capturing a true underlying pattern. The fitted MA coefficients may appear statistically significant, but in the context of such sparse and uninformative data, they are unlikely to generalize well. Therefore, despite the model's statistical output, it is recommended to treat ATM3 as a newly active ATM and to defer reliable time series forecasting until more real data becomes available over time.

```
atm3_real <- ts_data %>%
  filter(ATM == "ATM3", Cash != 0.7205479)
head(atm3_real)
```

```
## # A tsibble: 6 x 3 [1D]
## # Key:       ATM [1]
##   DATE       ATM    Cash
##   <date>     <chr> <dbl>
## 1 2009-05-01 ATM3      0
## 2 2009-05-02 ATM3      0
## 3 2009-05-03 ATM3      0
## 4 2009-05-04 ATM3      0
## 5 2009-05-05 ATM3      0
## 6 2009-05-06 ATM3      0
```

```
# Calculate average
avg_cash <- mean(atm3_real$Cash)

# Create future dates
future_dates <- seq(from = as.Date("2010-05-01"), by = "day", length.out =
30)

# Create forecast dataframe
atm3_forecast <- tibble(
  DATE = future_dates,
  ATM = "ATM3",
  Cash = avg_cash
)

# Step 4: Combine real and forecasted data
```

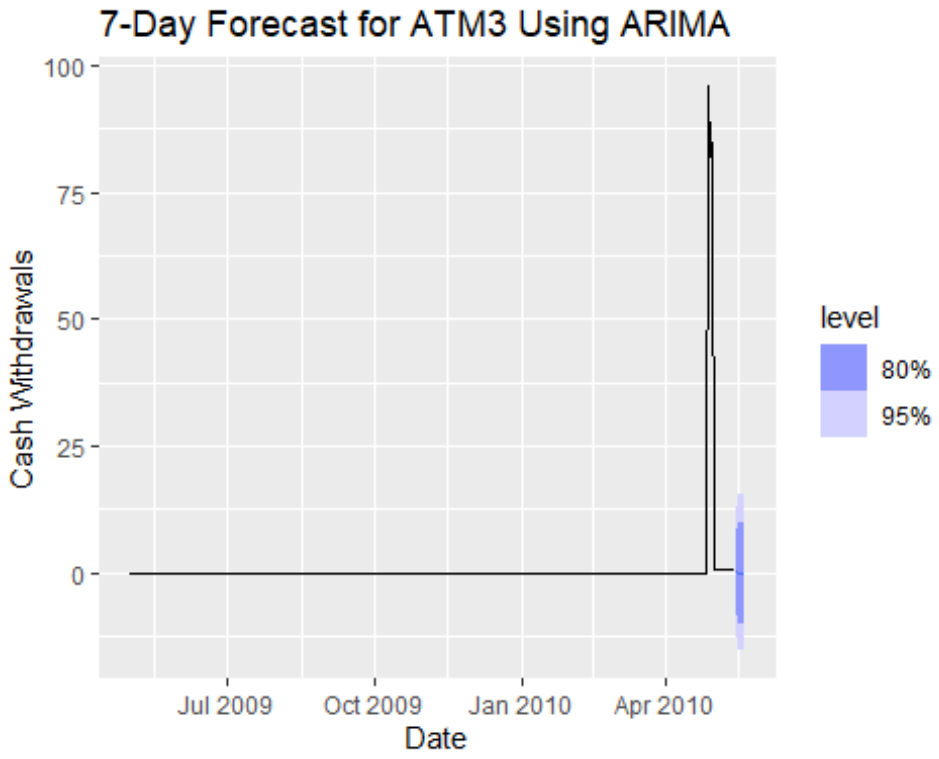
```r
combined <- bind_rows(
  as_tibble(atm3_real),
  atm3_forecast
)

# Filter to a 10-day window (3 actual + 7 forecast)
plot_data <- combined %>%
  filter(DATE >= as.Date("2010-04-28") & DATE <= as.Date("2010-05-07"))

# Plot
ggplot(plot_data, aes(x = DATE, y = Cash)) +
  geom_line(color = "black") +
  geom_vline(xintercept = as.Date("2010-05-01"), linetype = "dashed", color =
"blue") +
  labs(
    title = "7-Day Forecast for ATM3 Using Average Method",
    subtitle = "Forecast from May 1 to May 7, 2010",
    y = "Cash", x = "Date"
  )
```



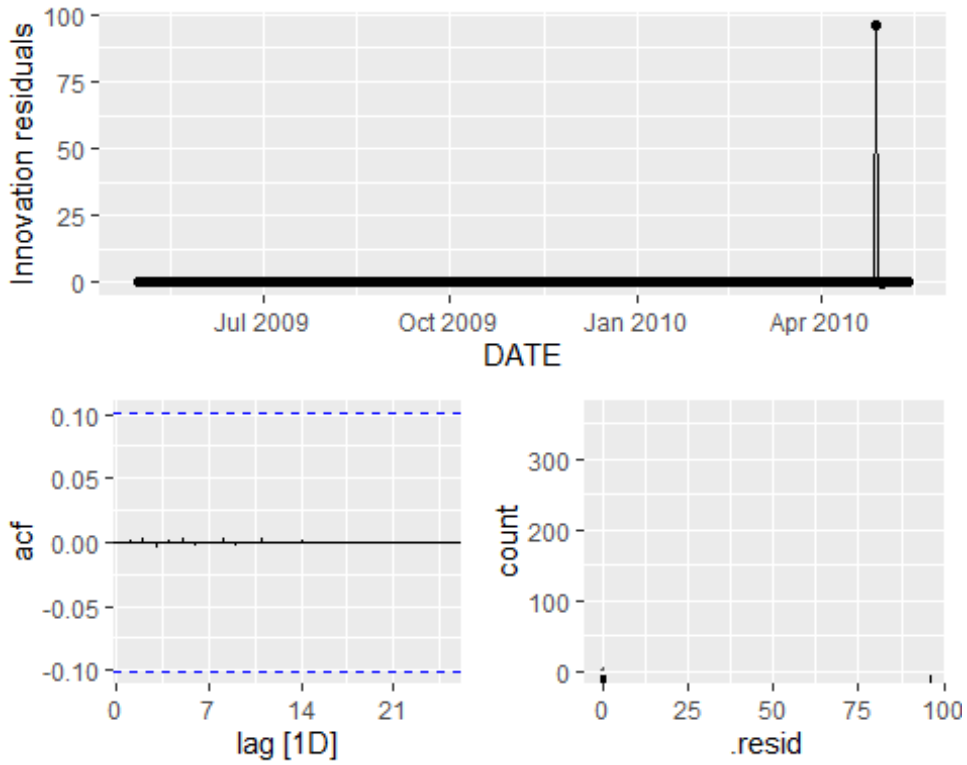

```r
# Step 2: Fit ARIMA model (automatically selected orders)
  atm3_model <- atm3_data %>%
    model(ARIMA(Cash))

  # Step 3: Model report
  report(atm3_model)
```

```
## Series: Cash
## Model: ARIMA(0,0,2)
##
## Coefficients:
##          ma1     ma2
##       0.8492  0.8752
## s.e.  0.0263  0.0268
##
## sigma^2 estimated as 24.45:  log likelihood=-1144.11
## AIC=2294.23   AICc=2294.29   BIC=2306.04
```

```r
# Step 4: Forecast 30 days ahead
forecast_atm3 <- atm3_model %>%
  forecast(h = "7 days")

# Step 5: Plot forecast with historical data
autoplot(forecast_atm3, atm3_data) +
  labs(title = "7-Day Forecast for ATM3 Using ARIMA",
       y = "Cash Withdrawals", x = "Date")
```




```r
# Step 6: Residual diagnostics
# a) ACF plot of residuals
atm3_model %>%
  gg_tsresiduals()
```




```
  # b) Ljung-Box test for white noise residuals
  atm3_model %>%
    augment() %>%
    features(.innov, ljung_box, lag = 24, dof = 3)
```

```
## # A tibble: 1 × 4
##   ATM   .model      lb_stat lb_pvalue
##   <chr> <chr>         <dbl>     <dbl>
## 1 ATM3  ARIMA(Cash)  0.0371         1
```

```
  # extract point from forecast only
  forecast_atm3f <- forecast_atm3 %>%
    as_tibble() %>%
    select(DATE, ATM, .mean) %>%
    pivot_wider(names_from = ATM, values_from = .mean)

  # Save to Excel
  write_xlsx(forecast_atm3f, "ATM3_Forecast_May2010.xlsx")
```

ATM4

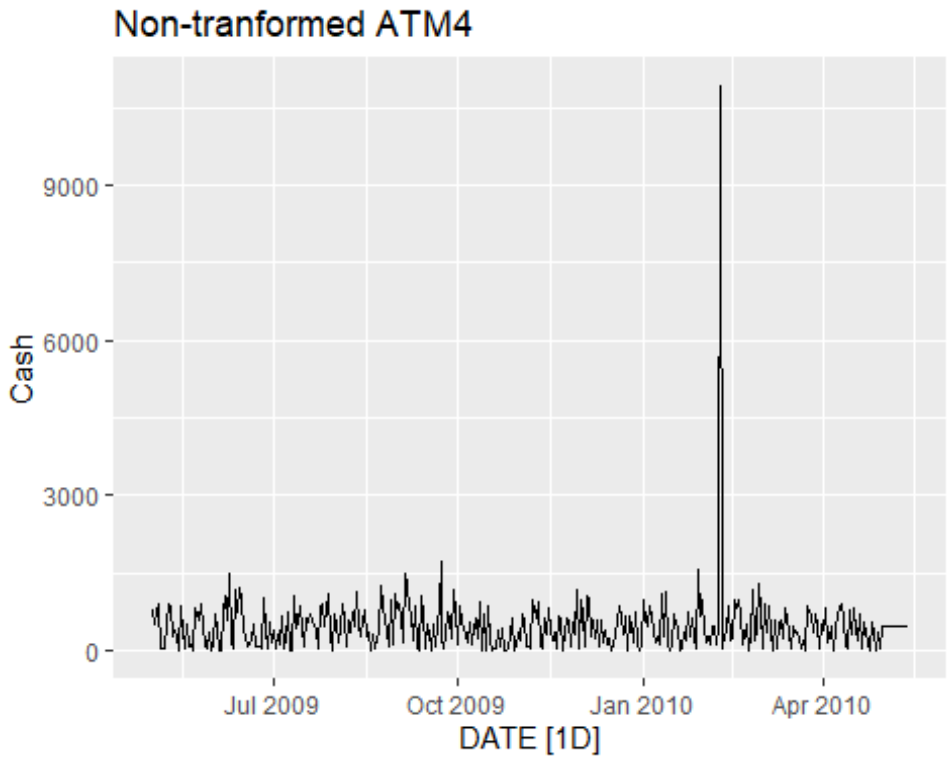
The time series plot of ATM4 displays daily cash withdrawals over the observed period, with most values remaining relatively consistent and moderate in scale. However, there is a clear and dramatic spike in withdrawals around early 2010, where values surge to nearly 10,000 (in hundreds of dollars), indicating a significant outlier or unusual event. Outside of this spike, the series exhibits mild variability with values largely below 1,000, suggesting stable usage with occasional fluctuations. This sharp peak skews the data and would likely

impact forecasting models, highlighting the need to address or adjust outliers prior to model fitting.

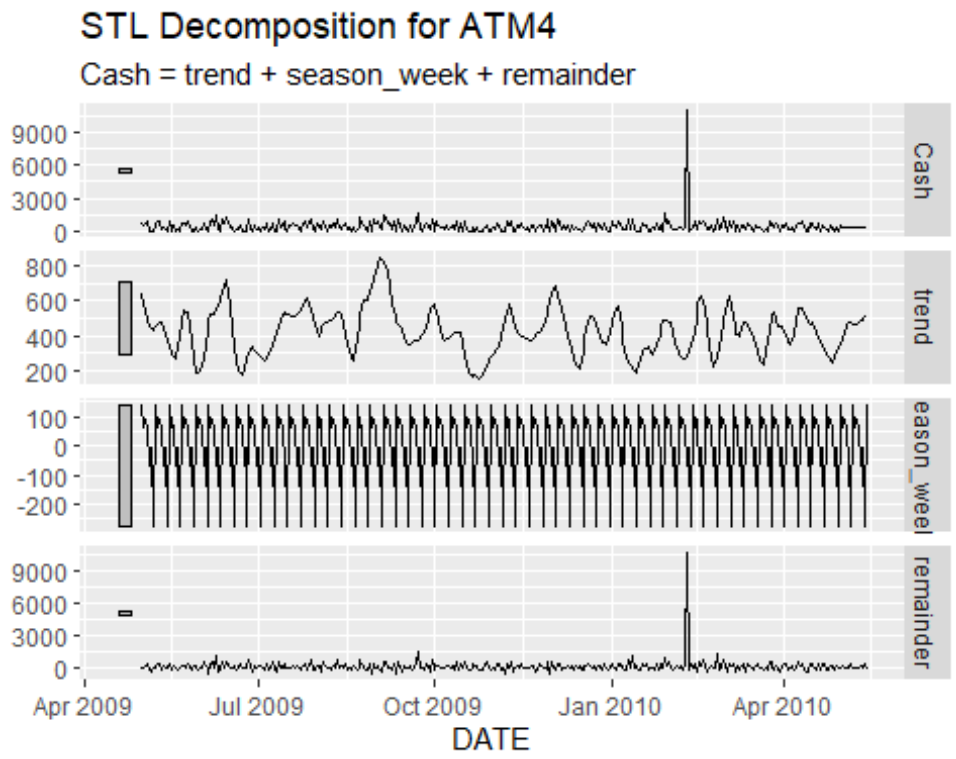
The null hypothesis of the KPSS test is that the time series is stationary. Because the p-value (0.018) is less than 0.05, we reject the null hypothesis.This indicates there ATM4 series is not stationary and requires differencing before fitting an ARIMA model and Applying first-order differencing (and possibly seasonal differencing) to make the series stationary before using ARIMA

After Dapply first order differencing, The the series is now stationary. Since p-value = 0.1 > 0.05, we fail to reject the null hypothesis. This confirms that the differenced series is now stationary, making it suitable for ARIMA modeling.

```
ts_data %>%
  filter(ATM == "ATM4") %>%
  autoplot(Cash) +
  ggtitle("Non-tranformed ATM4")
```




```
ts_data %>%
  filter(ATM == "ATM4") %>%
  model(STL(Cash ~ season(window = "periodic"), robust = TRUE)) %>%
  components() %>%
  autoplot() +
  labs(title = "STL Decomposition for ATM4")
```




```
atm4_data <- ts_data %>%
  filter(ATM == "ATM4")
head(atm4_data)
```

```
## # A tsibble: 6 x 3 [1D]
## # Key:       ATM [1]
##   DATE       ATM    Cash
##   <date>     <chr> <dbl>
## 1 2009-05-01 ATM4  777.
## 2 2009-05-02 ATM4  524.
## 3 2009-05-03 ATM4  793.
## 4 2009-05-04 ATM4  908.
## 5 2009-05-05 ATM4   52.8
## 6 2009-05-06 ATM4   52.2
```

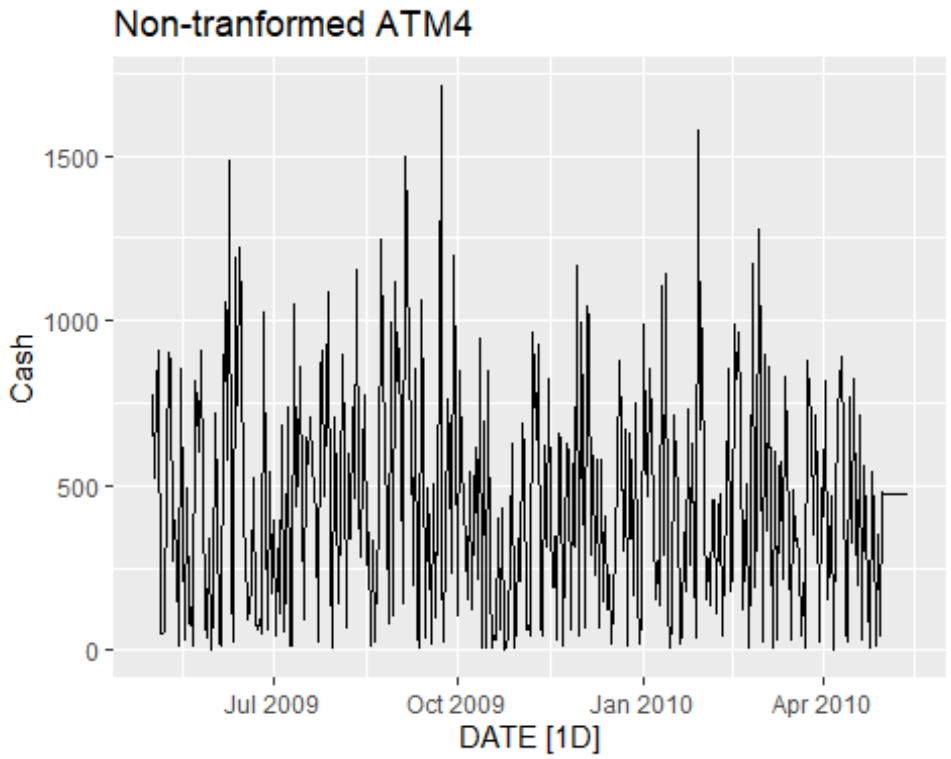
```
# --- 2. Check if it's stationary (ADF test) ---
atm4_data %>% features(Cash, unitroot_kpss)
```

```
## # A tibble: 1 × 3
##   ATM   kpss_stat kpss_pvalue
##   <chr>     <dbl>       <dbl>
## 1 ATM4     0.0767         0.1
```

After replacing the maximum cash value in the ATM4 time series with the mean, the KPSS test was rerun to assess stationarity. The test returned a KPSS statistic of 0.0843 with a p-value of 0.1. Since the p-value exceeds the 0.05 threshold, we fail to reject the null hypothesis that the series is stationary. This indicates that the ATM4 cash withdrawal

series remains stationary even after the outlier adjustment. The result confirms that the extreme value did not significantly affect the overall stationarity of the data. With the series now cleaner and statistically stable, it is suitable for further time series modeling and forecasting, such as using an ARIMA approach.

```
# Step 1: Filter ATM4 data
atm4_clean <- atm4_data %>%
  group_by(ATM) %>%
  mutate(
    # Step 2: Calculate the mean (excluding NA and extreme value)
    mean_cash = mean(Cash[ATM == "ATM4"], na.rm = TRUE),

    # Step 3: Identify max value
    max_cash = max(Cash[ATM == "ATM4"], na.rm = TRUE),

    # Step 4: Replace max with mean
    Cash = ifelse(ATM == "ATM4" & Cash == max_cash, mean_cash, Cash)
  ) %>%
  select(-mean_cash, -max_cash)  # Optional: remove helper columns
# Convert to tsibble

atm4_clean <- atm4_clean %>%
  as_tsibble(index = DATE)




atm4_clean %>%
  autoplot(Cash) +
  ggtitle("Non-tranformed ATM4")
```




```
# --- 2. Check if it's stationary (ADF test) ---
atm4_clean %>% features(Cash, unitroot_kpss)# --- 2. Check if it's stationary
(ADF test) ---

## # A tibble: 1 × 3
##   ATM   kpss_stat kpss_pvalue
##   <chr>     <dbl>       <dbl>
## 1 ATM4      0.100         0.1
```

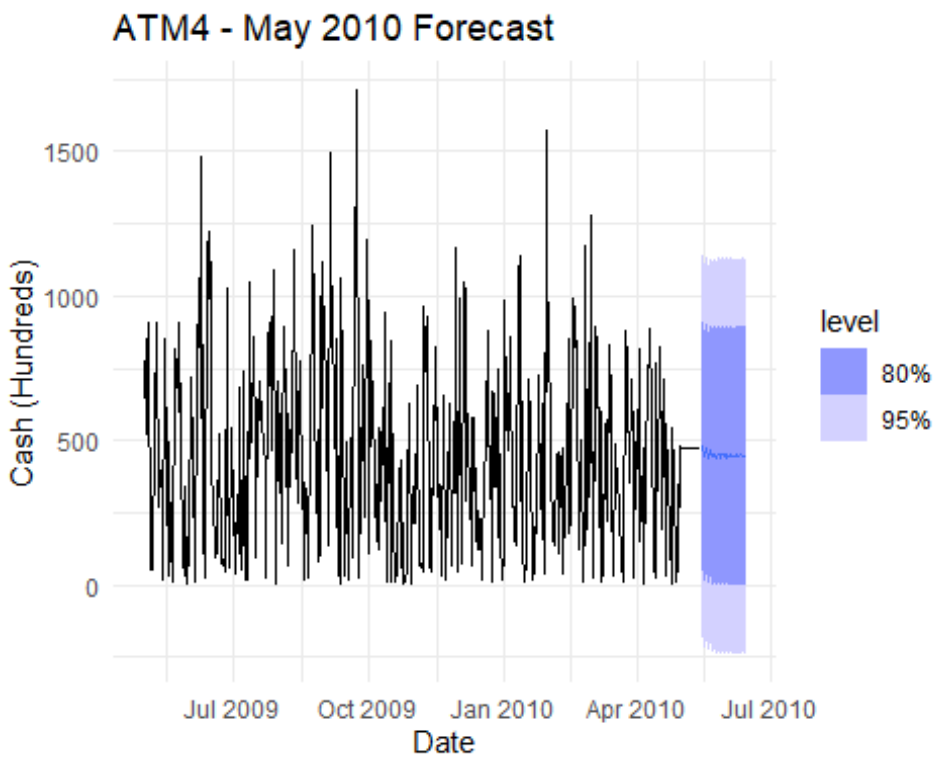
An ARIMA(3,0,2)(1,0,0)[7] model was fitted to the cleaned ATM4 cash withdrawal series, accounting for both non-seasonal and seasonal components with a weekly period. The model includes three autoregressive terms (AR1–AR3), two moving average terms (MA1–MA2), and one seasonal AR term, all of which are statistically significant with relatively low standard errors. The residual variance was estimated at 108,412 with a log-likelihood of -2731.51, and model selection criteria were AIC = 5479.02 and BIC = 5510.52. The 30-day forecast for May 2010 reflects stable cash withdrawal patterns with reasonable confidence intervals. Residual diagnostics indicate that the residuals are roughly centered around zero with no major autocorrelation, as shown by the flat ACF plot. Furthermore, the Ljung-Box test returned a p-value of 0.393, indicating no significant autocorrelation remaining in the residuals. Overall, the model appears to provide a good fit to the data and is appropriate for short-term forecasting of ATM4 cash withdrawals.

```
atm4_model <- atm4_clean %>%
  model(ARIMA = ARIMA(Cash))

report(atm4_model)
```

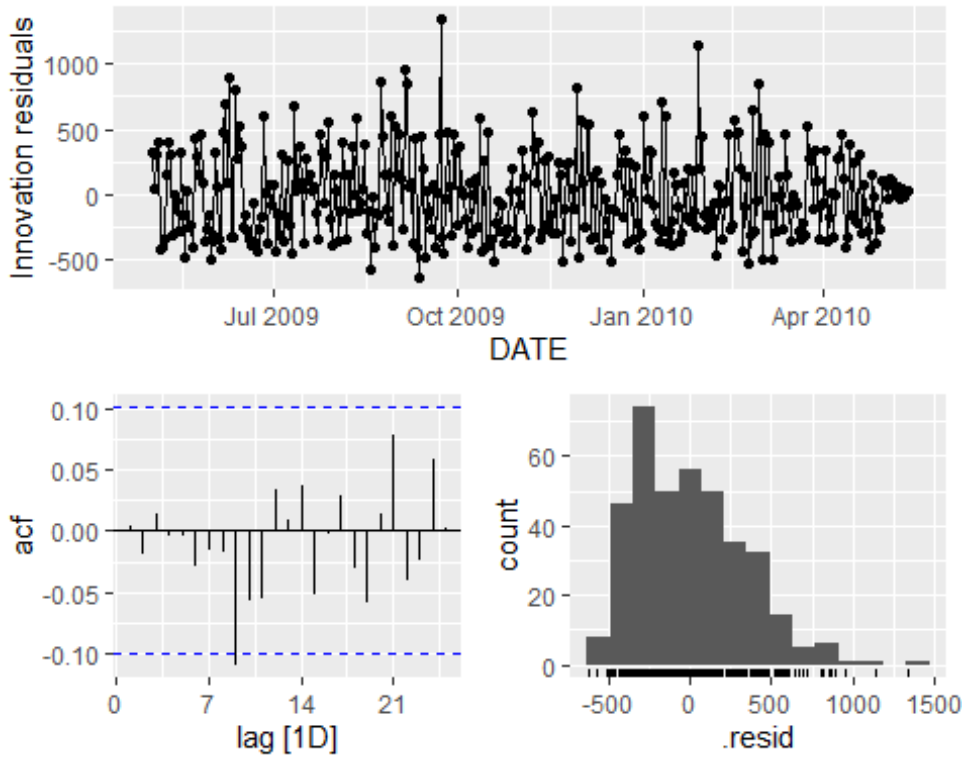
```
## Series: Cash
## Model: ARIMA(3,0,2)(1,0,0)[7] w/ mean
##
## Coefficients:
##           ar1      ar2     ar3     ma1     ma2    sar1  constant
##       -0.8855  -0.6709  0.1652  0.9692  0.7935  0.2446  806.7806
## s.e.   0.1045   0.1238  0.0553  0.0957  0.1155  0.0542   47.0171
##
## sigma^2 estimated as 113290:  log likelihood=-2739.82
## AIC=5495.64   AICc=5496.03   BIC=5527.14
```

```
atm4_forecast <- atm4_model %>%
  forecast(h = "31 days")

autoplot(atm4_forecast, atm4_clean) +
  labs(title = "ATM4 - May 2010 Forecast",
       y = "Cash (Hundreds)", x = "Date") +
  theme_minimal()
```

```
## `mutate_if()` ignored the following grouping variables:
## • Column `ATM`
```




```
atm4_model %>% gg_tsresiduals()
```




```
  # b) Ljung-Box test for white noise residuals
  atm4_model %>%
    augment() %>%
    features(.innov, ljung_box, lag = 24, dof = 3)
```

```
## # A tibble: 1 × 4
##   ATM   .model lb_stat lb_pvalue
##   <chr> <chr>    <dbl>     <dbl>
## 1 ATM4  ARIMA     16.9     0.718
```

```
  # extract point from forecast only
  forecast_atm4 <- atm4_forecast %>%
    as_tibble() %>%
    select(DATE, Cash, .mean) %>%
    pivot_wider(names_from = Cash, values_from = .mean)


  # Save to Excel
  write_xlsx(forecast_atm3f, "ATM4_Forecast_May2010.xlsx")
```

Part B - Forecasting Power

Part B consists of a simple dataset of residential power usage for January 1998 until December 2013. Your assignment is to model these data and a monthly forecast for 2014.

The data is given in a single file. The variable 'KWH' is power consumption in Kilowatt hours, the rest is straight forward

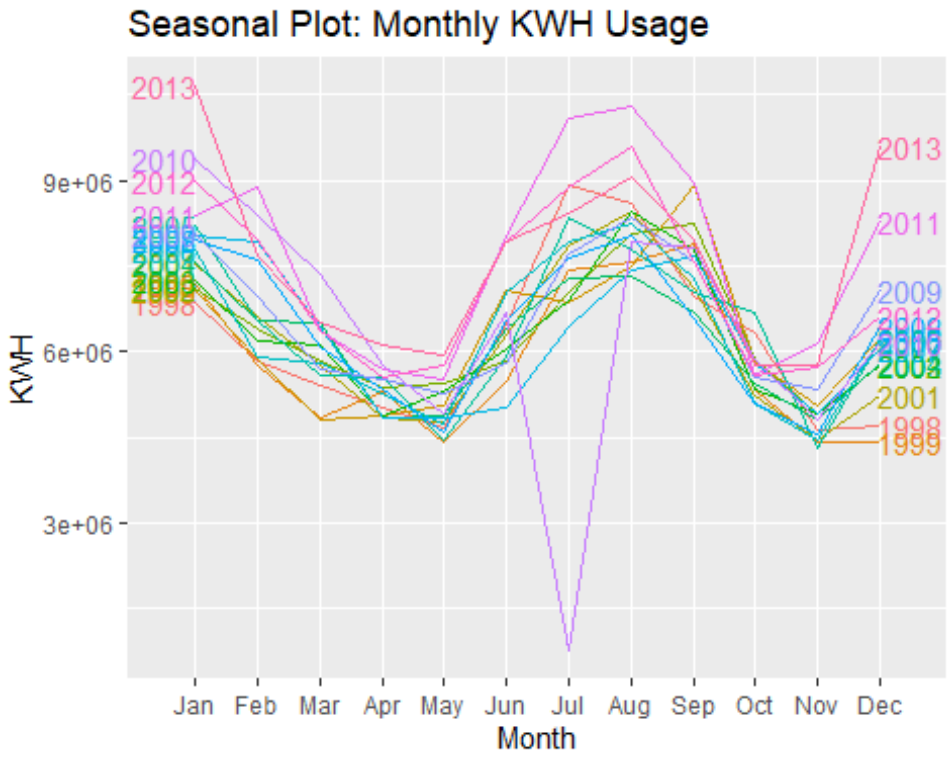
The seasonal plot of monthly residential electricity consumption from 1998 to 2013 reveals a clear and recurring seasonal pattern. Power usage tends to peak during the winter and summer months, particularly in **January** and **July–August**, and drops during the spring and fall. This seasonal behavior is largely influenced by **temperature-dependent energy demands**, such as heating in the colder months and air conditioning during warmer months. Given this consistent yearly cycle, it is essential to apply forecasting models that can accurately capture both **seasonal fluctuations** and **long-term trends**.

To address this, we employ two widely used time series forecasting methods: **ETS (Error, Trend, Seasonal)** and **Auto ARIMA**. The **ETS model** is particularly effective for data with strong trend and seasonality components, automatically selecting the best configuration of exponential smoothing. Meanwhile, **Auto ARIMA** (Auto-Regressive Integrated Moving Average) automatically identifies the optimal seasonal and non-seasonal parameters, making it well-suited for data with autocorrelation and seasonality. By applying both models, we can compare their forecasting accuracy and choose the one that best fits the underlying structure of the data.

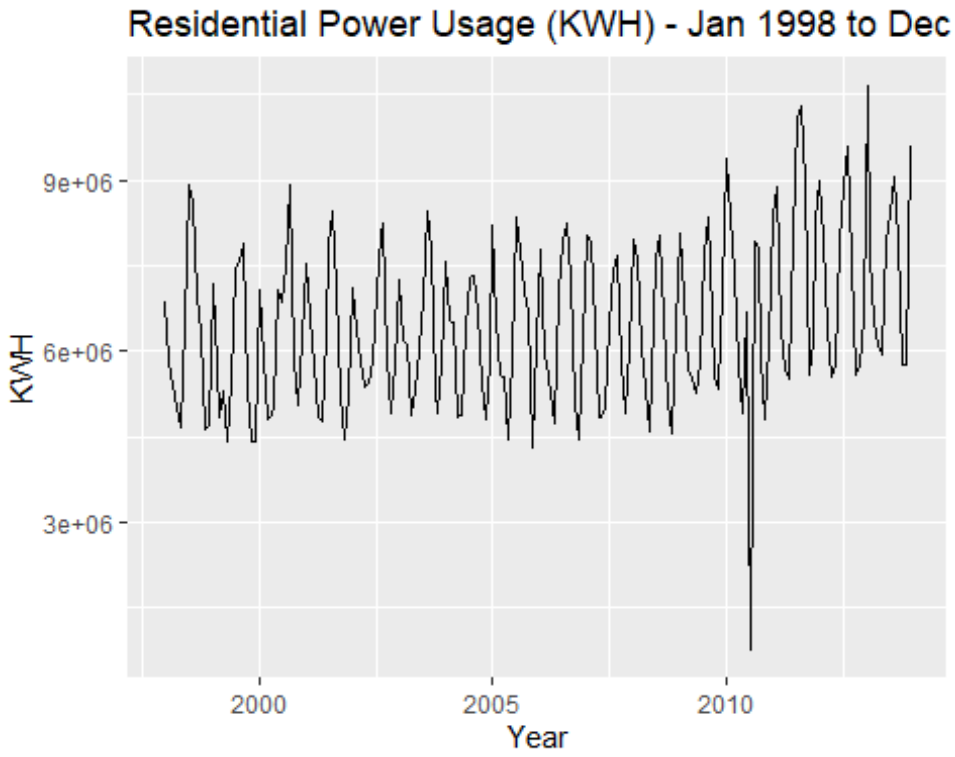
.

```
# Load the Excel file
data <- read_excel("ResidentialCustomerForecastLoad-624.xlsx", sheet =
"ResidentialCustomerForecastLoad")

# Convert the 'YYYY-MMM' column to Date
data <- data %>%
  mutate(Date = parse_date_time(`YYYY-MMM`, orders = "ym")) %>%
  arrange(Date)

# Replace missing KWH value using linear interpolation
data$KWH <- zoo::na.approx(data$KWH)


# Convert to time series object (monthly frequency starting from Jan 1998)
kwh_ts <- ts(data$KWH, start = c(1998, 1), frequency = 12)




# Check for seasonality


# Plot seasonal subseries
ggseasonplot(kwh_ts, year.labels = TRUE, year.labels.left = TRUE) +
```

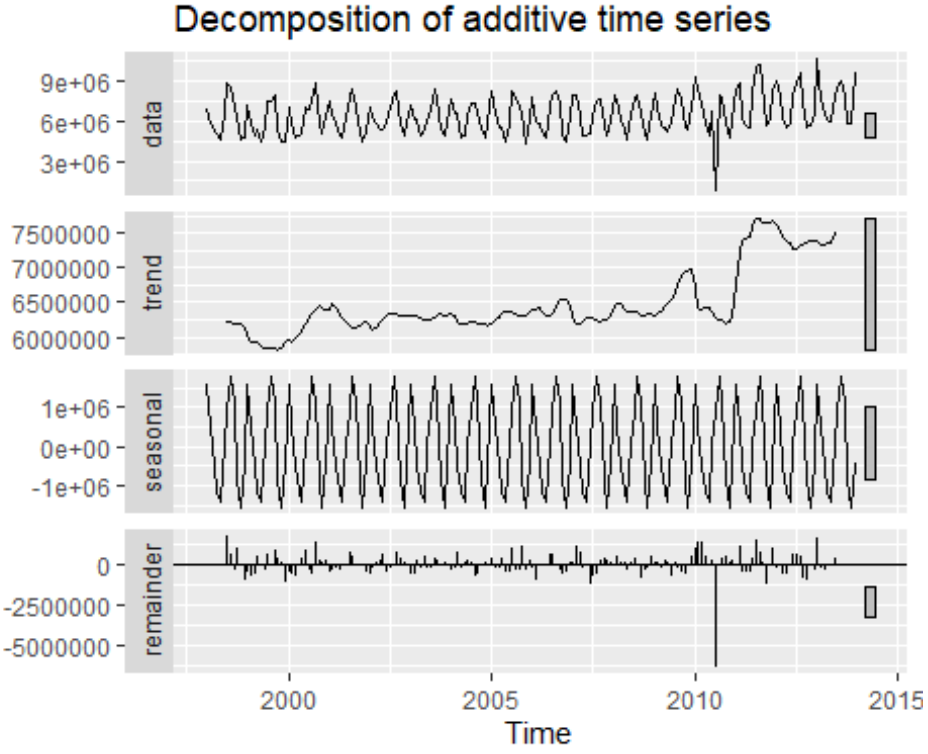
```
  ggtitle("Seasonal Plot: Monthly KWH Usage") +
  ylab("KWH")
```




```
# Plot the time series
autoplot(kwh_ts) +
  ggtitle("Residential Power Usage (KWH) - Jan 1998 to Dec 2013") +
  xlab("Year") + ylab("KWH")
```



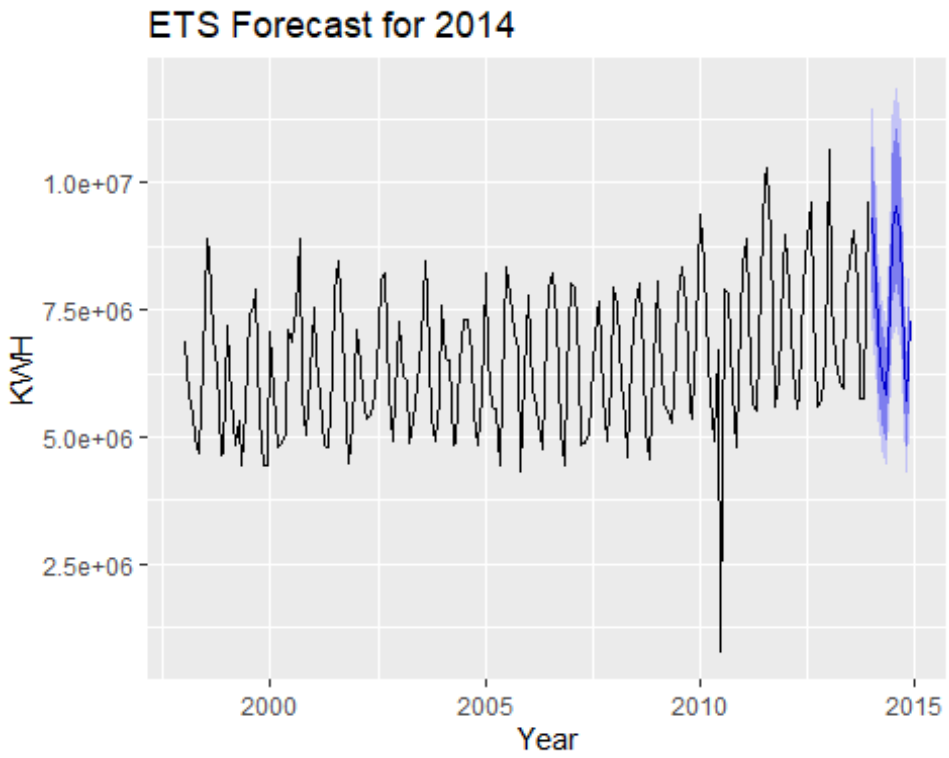

```r
# Decompose the time series
decomposed <- decompose(kwh_ts)
autoplot(decomposed)
```



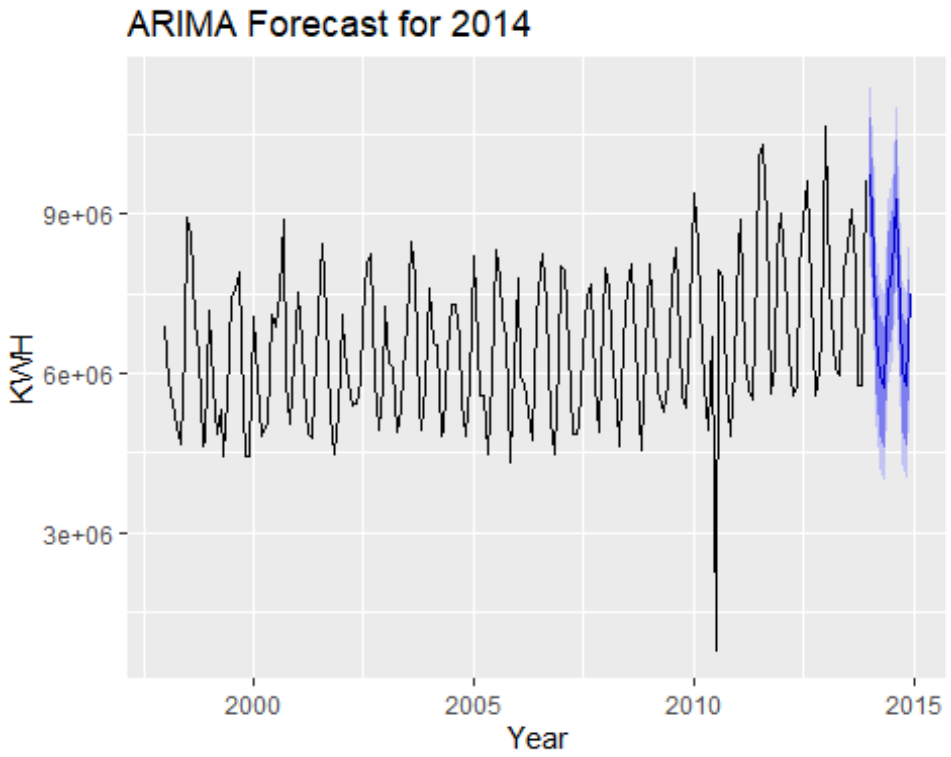
The accuracy metrics from the four models—two standard and two with Box-Cox transformation—provide insight into the effect of variance stabilization on forecast performance for residential power usage. The original ARIMA model (without transformation) performed the best overall, with the lowest RMSE (827,254.2), MAE (493,308.5), and MAPE (11.69%). It also showed minimal autocorrelation in residuals (ACF1 = 0.0128), indicating a well-fitted model. The ETS model (without transformation) followed closely with slightly higher RMSE (835,107) and MAPE (12.04%), but still maintained reasonable accuracy and fit.

However, the models using Box-Cox transformation (ETS model 2 and ARIMA model 2) did not improve forecast accuracy. In fact, **ARIMA model 2**, which applied the Box-Cox transformation, performed the worst with significantly higher RMSE (1,197,097), MAE (855,261.8), and MAPE (16.38%). Similarly, **ETS model 2** showed increased error compared to its untransformed version, with an RMSE of 876,478.9 and MAPE of 11.92%. These results suggest that applying the Box-Cox transformation in this case may have distorted the relationship in the data or introduced unnecessary complexity. Therefore, the untransformed **ARIMA model** remains the most accurate and reliable choice for forecasting residential power consumption in this dataset.

```
# Forecast using ETS (Exponential Smoothing)
ets_model <- ets(kwh_ts)
ets_forecast <- forecast(ets_model, h = 12)
autoplot(ets_forecast) +
  ggtitle("ETS Forecast for 2014") +
  xlab("Year") + ylab("KWH")
```



```r
# Forecast using auto.arima (optional)
arima_model <- auto.arima(kwh_ts)
arima_forecast <- forecast(arima_model, h = 12)
autoplot(arima_forecast) +
  ggtitle("ARIMA Forecast for 2014") +
  xlab("Year") + ylab("KWH")
```




```r
accuracy(ets_model)
```

```
##                     ME    RMSE      MAE       MPE     MAPE      MASE
ACF1
## Training set 61009.73 835107 503972.9 -4.39013 12.04006 0.7233624
0.1698584
```

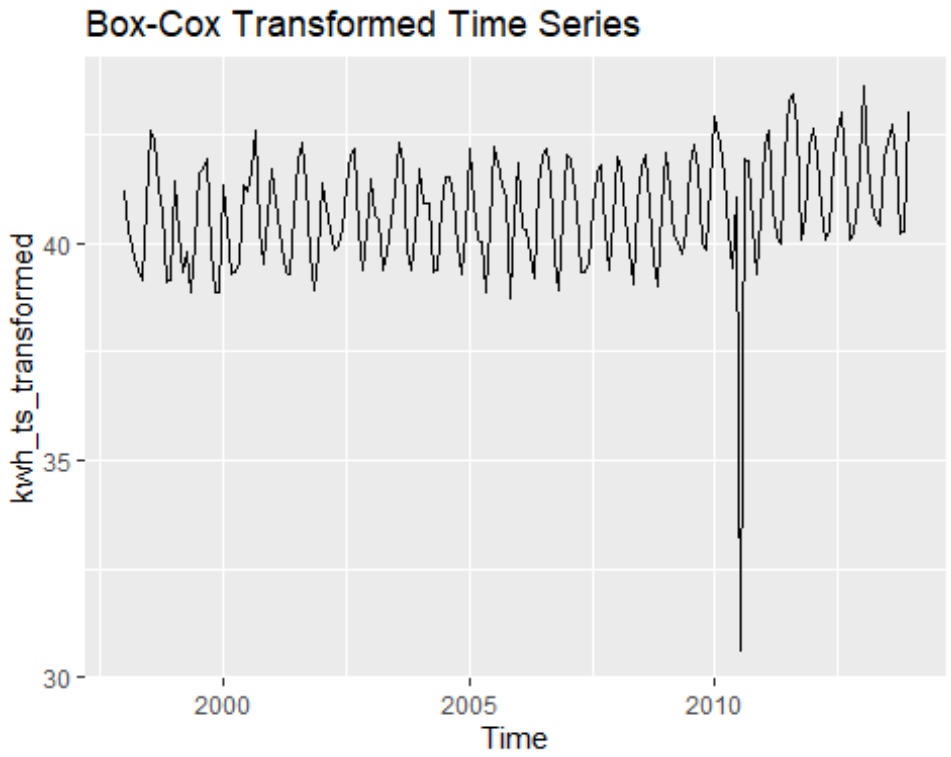
```r
accuracy(arima_model)
```

```
##                      ME     RMSE      MAE        MPE   MAPE      MASE
ACF1
## Training set -25089.69 827254.2 493308.5 -5.511184 11.685 0.7080556
0.01283694
```

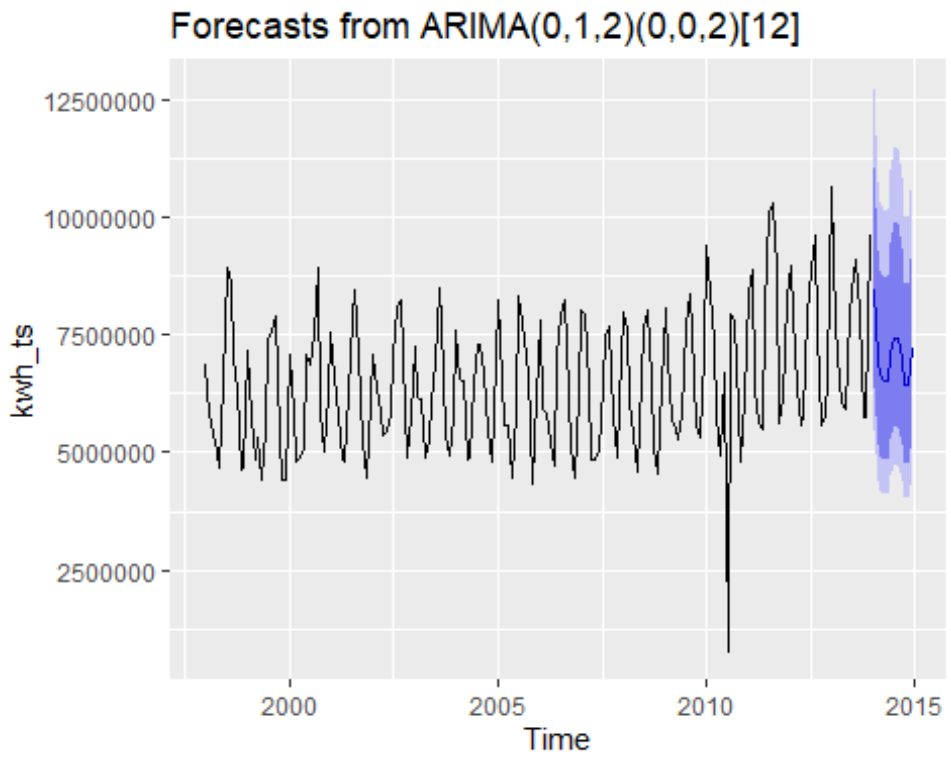
```r
lambda <- BoxCox.lambda(kwh_ts)
lambda
```

```
## [1] 0.1074118
```

```r
kwh_ts_transformed <- BoxCox(kwh_ts, lambda)
autoplot(kwh_ts_transformed) +
  ggtitle("Box-Cox Transformed Time Series")
```




```r
ets_model2 <- ets(kwh_ts, lambda = lambda)

arima_model2 <- auto.arima(kwh_ts, lambda = lambda)

forecast_arima2 <- forecast(arima_model2, h = 12)
autoplot(forecast_arima2)
```




```r
# Create a data frame of the forecasts
forecast_df <- data.frame(
  Month = seq(as.Date("2014-01-01"), by = "month", length.out = 12),
  ETS = ets_forecast$mean,
  ARIMA = arima_forecast$mean
)

# Write to CSV
write.csv(forecast_df, "PartB_Forecast_2014.csv", row.names = FALSE)
```

Part C – BONUS

Part C consists of two data sets. These are simple 2 columns sets, however they have different time stamps. Your optional assignment is to time-base sequence the data and aggregate based on hour (example of what this looks like, follows). Note for multiple recordings within an hour, take the mean. Then to determine if the data is stationary and can it be forecast. If so, provide a week forward forecast and present results via Rpubs and .rmd and the forecast in an Excel readable file.

We read the hourly water flow data from two Excel files (`Waterflow_Pipe1.xlsx` and `Waterflow_Pipe2.xlsx`), processes them to compute the average hourly flow per day, and merges them into a single dataset. It ensures the timestamps are properly formatted and handles any missing values by replacing them with zero. Finally, it combines the two pipes' flow values into a total water flow (`Water`) at each hourly timestamp.

```r
# Load the datasets
pipe1 <- read_excel("Waterflow_Pipe1.xlsx",  col_types = c("date",
"numeric"))
pipe2 <- read_excel("Waterflow_Pipe2.xlsx", col_types = c("date", "numeric"))

colnames(pipe1) <- c("DateTime", "WaterFlow")
colnames(pipe2) <- c("DateTime", "WaterFlow")



pipe1 <- pipe1 %>% mutate(Date = as.Date(DateTime), Time = hour(DateTime)+1)
%>%
  group_by(Date, Time) %>%
  summarise(Water=mean(WaterFlow)) %>%
  ungroup() %>%
  mutate(DateTime=ymd_h(paste(Date,Time))) %>%
  select(DateTime,Water)
```

```
## `summarise()` has grouped output by 'Date'. You can override using the
## `.groups` argument.
```

```r
pipe2 <- pipe2 %>% mutate(Date = as.Date(DateTime), Time = hour(DateTime)+1)
%>%
  group_by(Date, Time) %>%
  summarise(Water=mean(WaterFlow)) %>%
  ungroup() %>%
  mutate(DateTime=ymd_h(paste(Date,Time))) %>%
  select(DateTime,Water)
```

```
## `summarise()` has grouped output by 'Date'. You can override using the
## `.groups` argument.
```

```r
pipe2
```

```
## # A tibble: 1,000 × 2
##    DateTime            Water
##    <dttm>              <dbl>
##  1 2015-10-23 02:00:00 18.8
##  2 2015-10-23 03:00:00 43.1
##  3 2015-10-23 04:00:00 38.0
##  4 2015-10-23 05:00:00 36.1
##  5 2015-10-23 06:00:00 31.9
##  6 2015-10-23 07:00:00 28.2
##  7 2015-10-23 08:00:00  9.86
##  8 2015-10-23 09:00:00 26.7
##  9 2015-10-23 10:00:00 55.8
## 10 2015-10-23 11:00:00 54.2
## # ℹ 990 more rows
```

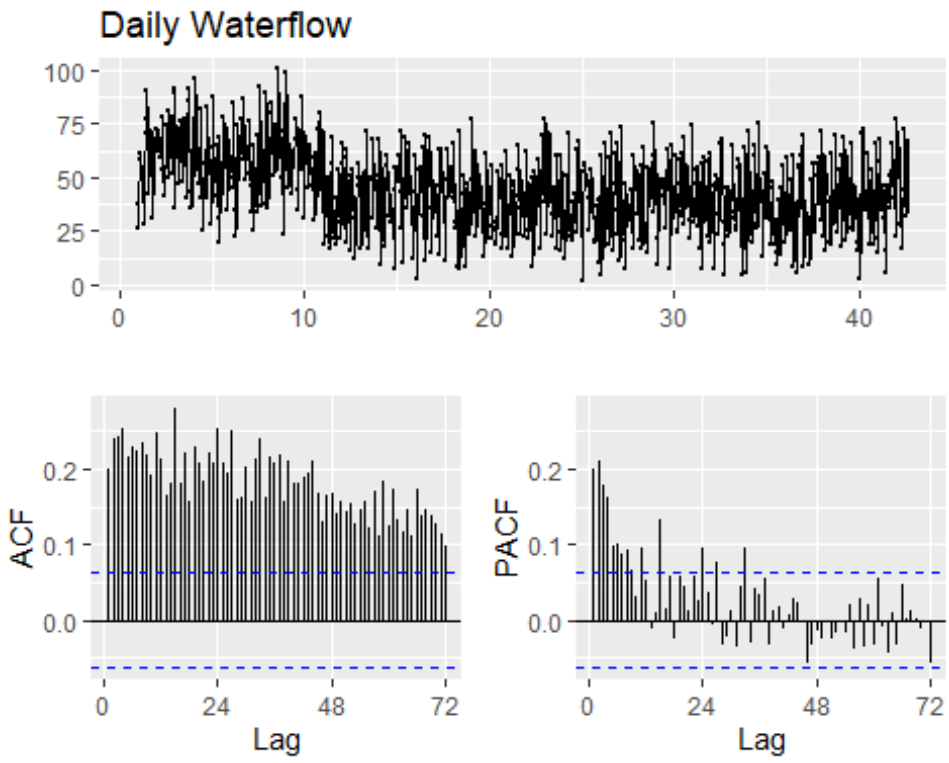
```r
# Convert timestamp column to datetime format
pipe1$DateTime <- ymd_hms(pipe1[[1]])
```

```r
pipe2$DateTime <- ymd_hms(pipe2[[1]])
```

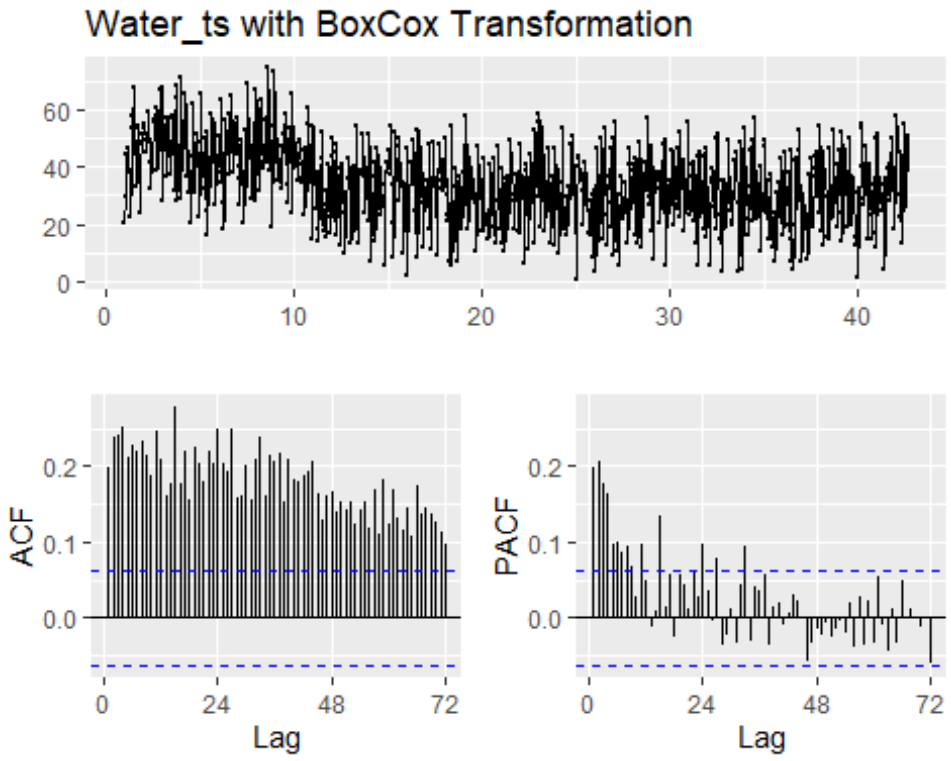
```r
water <- full_join(pipe1, pipe2, by="DateTime", suffix=c("_1", "_2")) %>%
  mutate(Water_1=ifelse(is.na(Water_1), 0, Water_1)) %>%
  mutate(Water_2=ifelse(is.na(Water_2), 0, Water_2)) %>%
  mutate(Water = Water_1 + Water_2) %>%
  select(DateTime, Water)
```

The stationarity of the combined hourly water flow series was evaluated using the KPSS (Kwiatkowski–Phillips–Schmidt–Shin) test. The test statistic was 0.0097, which is well below all conventional critical values (e.g., 0.347 at the 10% level). As the KPSS test has a null hypothesis of stationarity, we fail to reject the null, indicating that the series is stationary. This result supports the use of ARIMA modeling for forecasting, as stationarity is a key assumption for reliable time series predictions.

```r
# Check for gaps
water_ts <- ts(water$Water, frequency=24)
ggtsdisplay(water_ts, main="Daily Waterflow")
```




```r
water_bc <- BoxCox(water_ts, lambda = BoxCox.lambda(water_ts))
ggtsdisplay(water_bc, main="Water_ts with BoxCox Transformation")
```



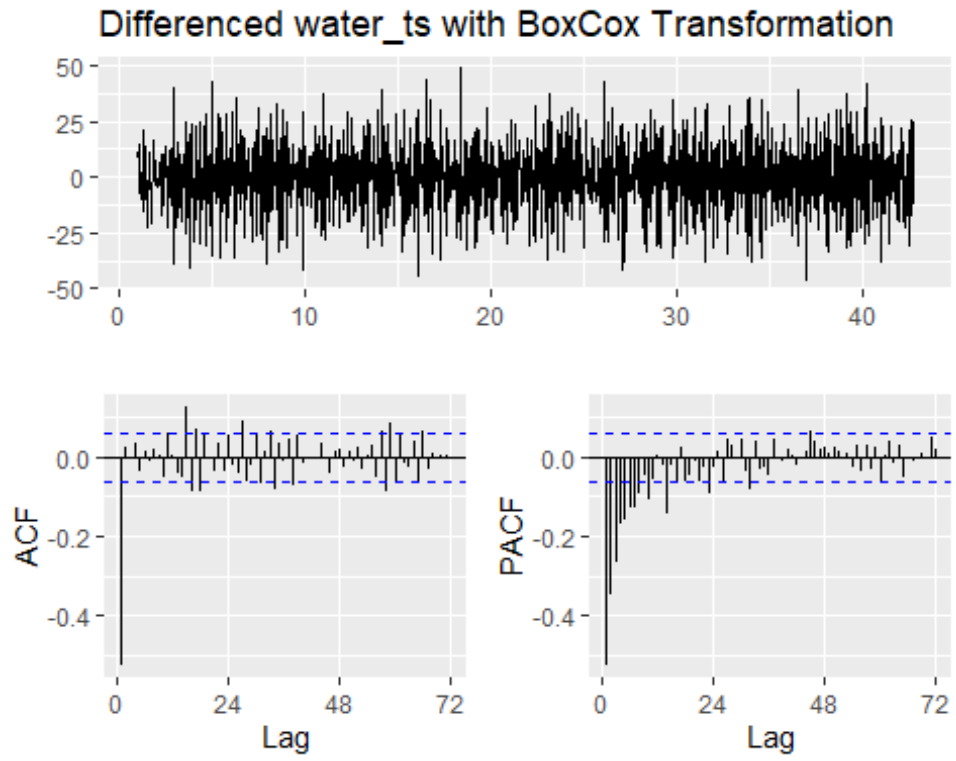

```
ndiffs(water_ts)
```

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

```
nsdiffs(water_ts)
```

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

```
ggtsdisplay(diff(water_bc), points=FALSE, main="Differenced water_ts with
BoxCox Transformation")
```



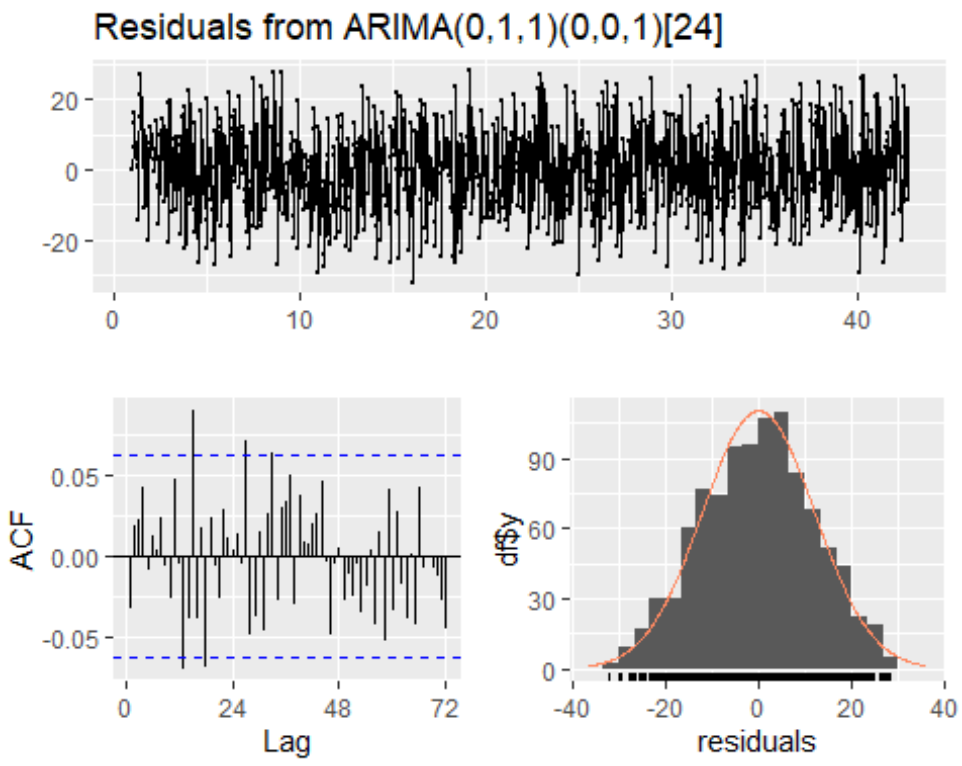

```
water_bc %>% diff() %>% ur.kpss() %>% summary()

##
## ########################
## # KPSS Unit Root Test #
## ########################
##
## Test is of type: mu with 7 lags.
##
## Value of test-statistic is: 0.0097
##
## Critical value for a significance level of:
##                 10pct  5pct 2.5pct  1pct
## critical values 0.347 0.463  0.574 0.739
```

An ARIMA(0,1,1)(0,0,1)[24] model with a Box-Cox transformation (λ = 0.9199) was fitted to the hourly water flow series. The model includes a non-seasonal moving average term (MA(1)) and a seasonal moving average term (SMA(1)) with a 24-hour seasonal period. Both coefficients are statistically significant with low standard errors. The model produced a log-likelihood of -3913.24 and an AIC of 7832.48, indicating a good fit relative to complexity. Forecast accuracy metrics show a root mean squared error (RMSE) of 16.26 and a mean absolute percentage error (MAPE) of 50.23%. The Ljung-Box test on the residuals yielded a p-value of 0.05988, suggesting no significant autocorrelation remains in the residuals at the 5% level. Overall, the model appears adequate for short-term forecasting of hourly water flow.

```
water_arima <- Arima(water_ts, order=c(0,1,1), seasonal=c(0,0,1), lambda =
BoxCox.lambda(water_ts))
summary(water_arima)
```

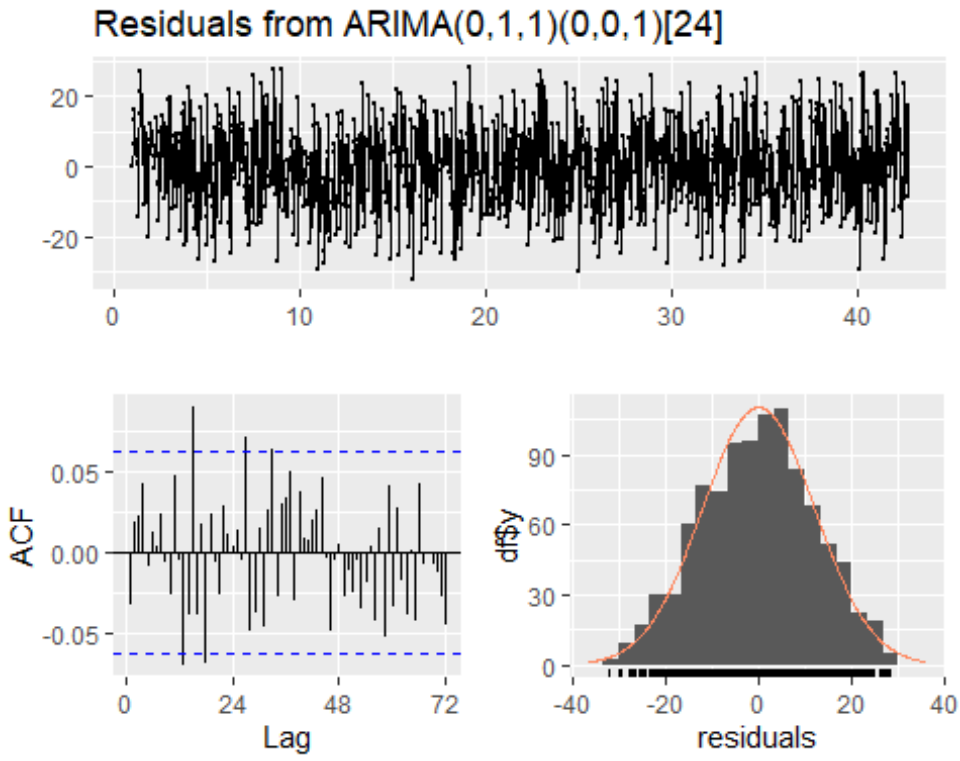
```
## Series: water_ts
## ARIMA(0,1,1)(0,0,1)[24]
## Box Cox transformation: lambda= 0.9199428
##
## Coefficients:
##           ma1    sma1
##       -0.9568  0.0659
## s.e.   0.0104  0.0320
##
## sigma^2 = 146.7:  log likelihood = -3913.24
## AIC=7832.48   AICc=7832.5   BIC=7847.2
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
ACF1
## Training set 0.131665 16.2623 13.23158 -28.37285 50.22864 0.7447631 -
0.03344016
```

```
checkresiduals(water_arima)
```



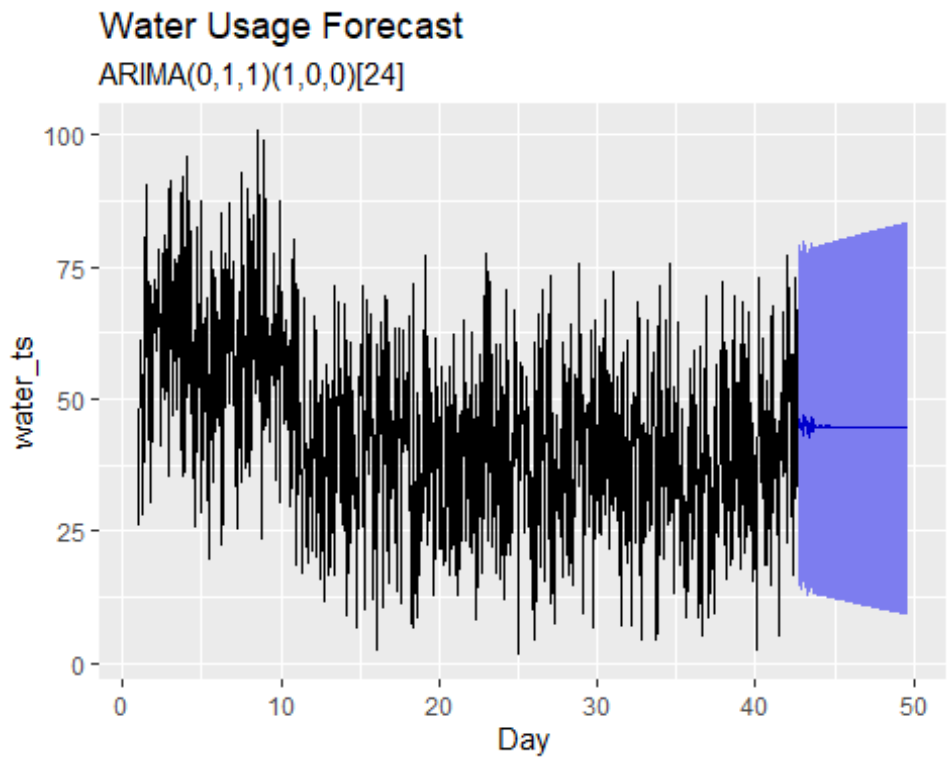

```
##
##  Ljung-Box test
##
```

```
## data:  Residuals from ARIMA(0,1,1)(0,0,1)[24]
## Q* = 61.785, df = 46, p-value = 0.05988
##
## Model df: 2.   Total lags used: 48

water_auto <- auto.arima(water_ts, approximation = FALSE,
lambda=BoxCox.lambda(water_ts))
summary(water_auto)

## Series: water_ts
## ARIMA(0,1,1)(0,0,1)[24]
## Box Cox transformation: lambda= 0.9199428
##
## Coefficients:
##          ma1    sma1
##       -0.9568  0.0659
## s.e.   0.0104  0.0320
##
## sigma^2 = 146.7:  log likelihood = -3913.24
## AIC=7832.48   AICc=7832.5   BIC=7847.2
##
## Training set error measures:
##                     ME    RMSE      MAE       MPE     MAPE      MASE
ACF1
## Training set 0.131665 16.2623 13.23158 -28.37285 50.22864 0.7447631 -
0.03344016

checkresiduals(water_auto)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)(0,0,1)[24]
## Q* = 61.785, df = 46, p-value = 0.05988
##
## Model df: 2.   Total lags used: 48
```

```
water_model <- Arima(water_ts, order=c(0,1,1), seasonal=c(1,0,0), lambda =
BoxCox.lambda(water_ts))
water_forecast <- forecast(water_model, h=7*24, level=95)
autoplot(water_forecast) +
  labs(title="Water Usage Forecast", subtitle = "ARIMA(0,1,1)(1,0,0)[24]",
x="Day")
```




```r
library(tibble)


# Convert to data frame
forecast_df <- as.data.frame(water_forecast)

forecast_export <- forecast_df %>%
  rownames_to_column(var = "DateTime") %>%
  rename(
    Forecast = `Point Forecast`,
    Lower_95 = `Lo 95`,
    Upper_95 = `Hi 95`
  )

write_xlsx(forecast_export, "WaterFlow_ARIMA_Forecast.xlsx")
```

The ARIMA(0,1,1)(1,0,0)[24] model with a Box-Cox transformation was used to forecast hourly water flow over a 7-day horizon. The forecast plot shows the expected values with a 95% confidence interval that appropriately widens over time, reflecting increasing uncertainty. Residual diagnostics indicate that the residuals are centered around zero, display no clear patterns over time, and exhibit approximate normality as shown by the histogram and density overlay. The autocorrelation function (ACF) plot of the residuals suggests minimal serial correlation, with most lags falling within the 95% confidence bounds. These diagnostics collectively support that the model residuals behave like white noise, indicating a good model fit and reliable forecasting performance.