TIME SERIES ANALYSIS MIDTERM EXAMS

Julius Hai

2025-10-16

SECTION A Q1

```
library(fpp3)
```

```
## Warning: package 'fpp3' was built under R version 4.5.1
```

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

```
## ── Attaching packages ─────────────────────────────────────── fpp3 1.
0.2 ──
```

```
## ✓ tibble      3.2.1     ✓ tsibble     1.1.6
## ✓ dplyr       1.1.4     ✓ tsibbledata 0.4.1
## ✓ tidyr       1.3.1     ✓ feasts      0.4.2
## ✓ lubridate   1.9.4     ✓ fable       0.4.1
## ✓ ggplot2     4.0.0
```

```
## Warning: package 'dplyr' was built under R version 4.5.1
```

```
## Warning: package 'tidyr' was built under R version 4.5.1
```

```
## Warning: package 'lubridate' was built under R version 4.5.1
```

```
## Warning: package 'ggplot2' was built under R version 4.5.1
```

```
## Warning: package 'tsibble' was built under R version 4.5.1
```

```
## Warning: package 'tsibbledata' was built under R version 4.5.1
```

```
## Warning: package 'feasts' was built under R version 4.5.1
```

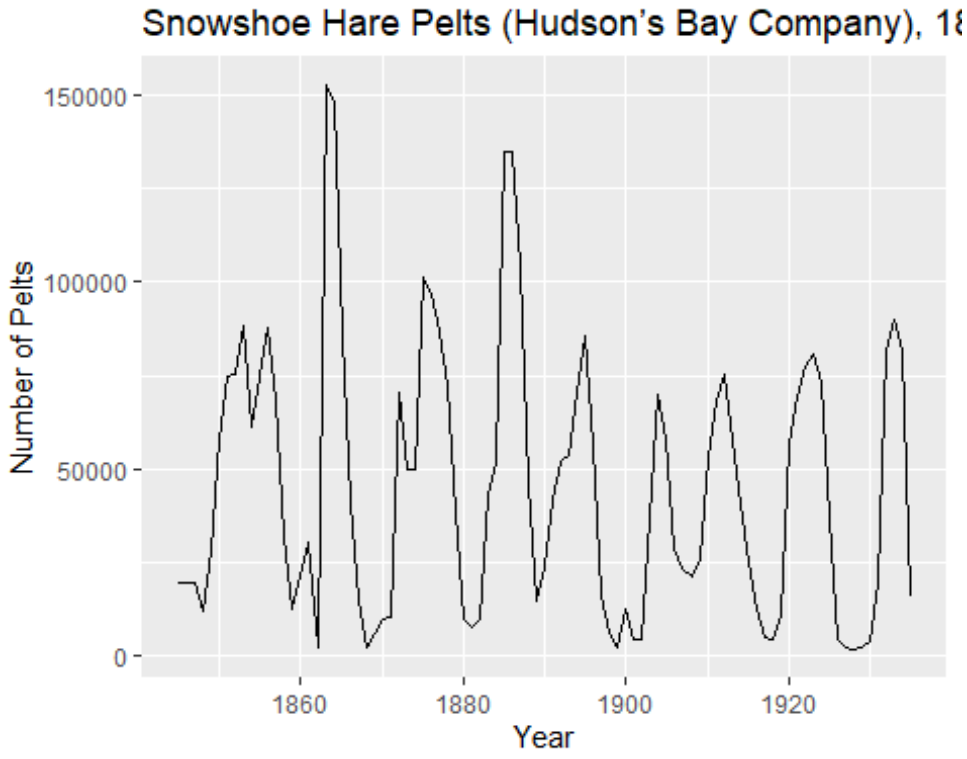
```
## Warning: package 'fabletools' was built under R version 4.5.1
```

```
## Warning: package 'fable' was built under R version 4.5.1
```

```
## ── Conflicts ─────────────────────────────────────────── fpp3_confli
cts ──
## ✖ lubridate::date()    masks base::date()
## ✖ dplyr::filter()      masks stats::filter()
## ✖ tsibble::intersect() masks base::intersect()
## ✖ tsibble::interval()  masks lubridate::interval()
## ✖ dplyr::lag()         masks stats::lag()
```

```
## ✖ tsibble::setdiff()   masks base::setdiff()
## ✖ tsibble::union()     masks base::union()
```

```
pelt |>
  autoplot(Hare) +
  labs(
    title = "Snowshoe Hare Pelts (Hudson's Bay Company), 1845–1935",
    x = "Year",
    y = "Number of Pelts"
  )
```



Interpretation

The time plot shows clear cyclical fluctuations in the number of Snowshoe Hare pelts between 1845 and 1935, indicating repeating population cycles roughly every 10 years. Overall, the data exhibit strong periodicity but no consistent long-term upward or downward trend. Q2

```
pelt |>
  model(ARIMA(Hare ~ pdq(4,0,0))) |>
  report()
```

```
## Series: Hare
## Model: ARIMA(4,0,0) w/ mean
##
## Coefficients:
##          ar1      ar2      ar3      ar4  constant
##       0.8219  -0.2891  -0.0057  -0.2165  30993.12
```

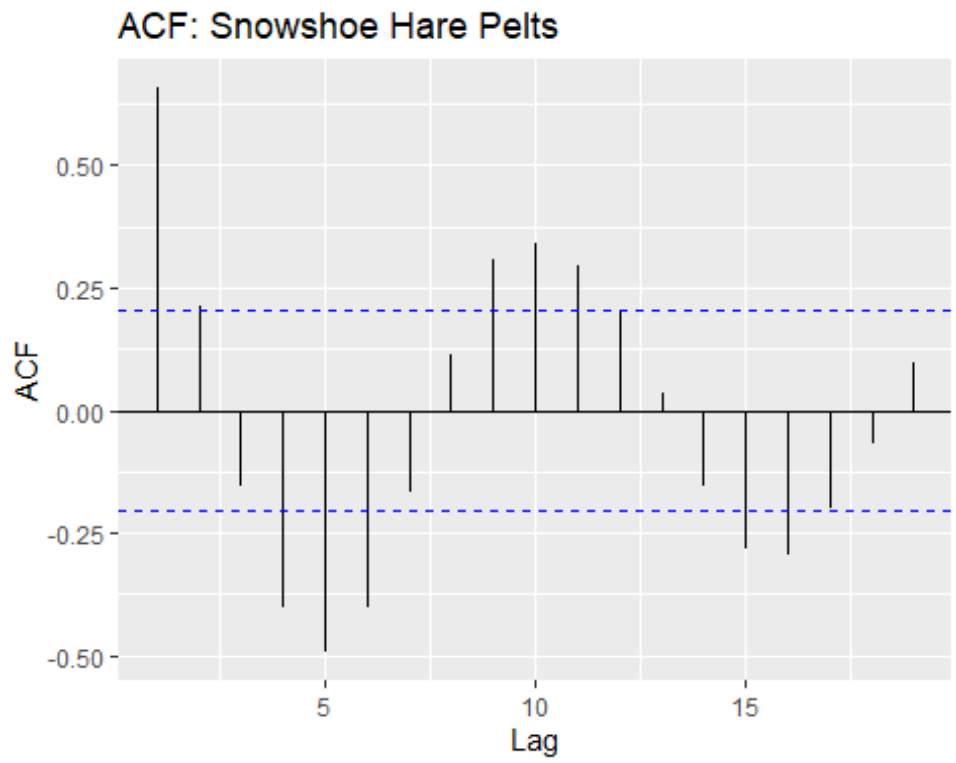
```
## s.e.  0.1037   0.1367   0.1358   0.1036   2497.78
##
## sigma^2 estimated as 587643879:  log likelihood=-1045.91
## AIC=2103.82   AICc=2104.82   BIC=2118.89
```

Interpretation

The fitted ARIMA(4,0,0) model indicates strong first-order dependence ($\phi_1 = 0.82$) with weaker higher-order lags. The positive constant (≈ 30,993) represents the average level of the series, and the relatively low AIC values suggest a reasonably good model fit for the hare pelt data.

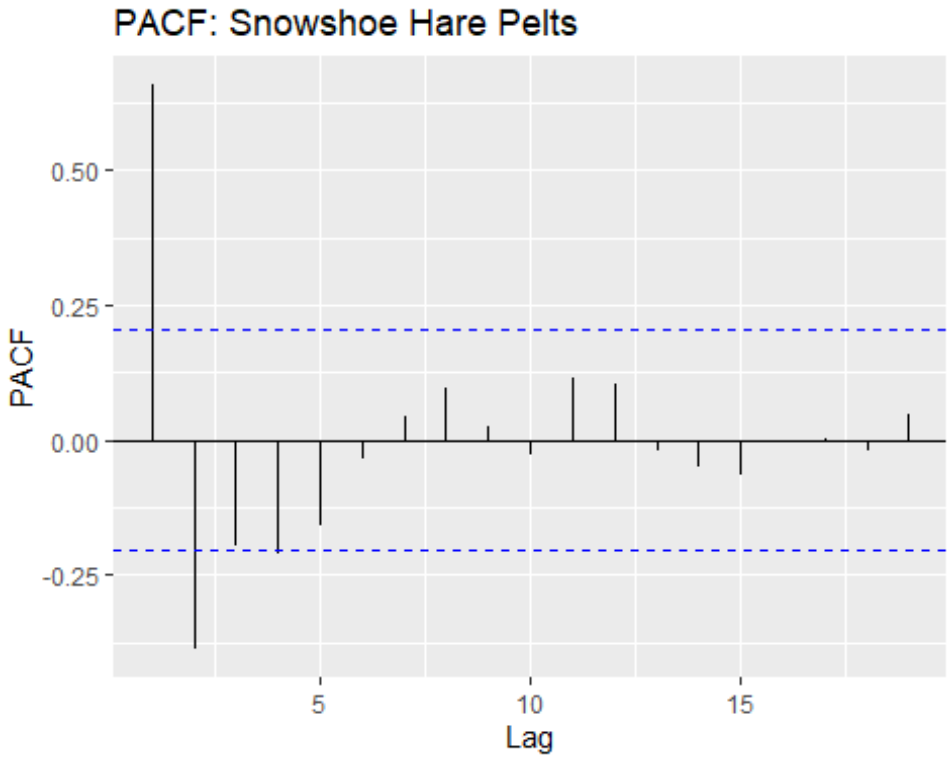
Q3

```
pelt |>
  ACF(Hare) |>
  autoplot() +
  labs(
    title = "ACF: Snowshoe Hare Pelts",
    x = "Lag",
    y = "ACF"
  )
```




```
pelt |>
  PACF(Hare) |>
  autoplot() +
```

```
  labs(
    title = "PACF: Snowshoe Hare Pelts",
    x = "Lag",
    y = "PACF"
  )
```



interpretation

Both the ACF and PACF confirm an autoregressive pattern: the ACF tapers gradually while the PACF cuts off after lag 4. Together, they indicate that the Snowshoe Hare pelts series is well modeled by an ARIMA(4,0,0) process.

Q4

```
c <- 30993
phi1 <- 0.82
phi2 <- -0.29
phi3 <- -0.01
phi4 <- -0.22

y <- c(19520, 82110, 89760, 81660, 15760)

y1 <- c + phi1*y[5] + phi2*y[4] + phi3*y[3] + phi4*y[2]
y2 <- c + phi1*y1   + phi2*y[5] + phi3*y[4] + phi4*y[3]
y3 <- c + phi1*y2   + phi2*y1   + phi3*y[5] + phi4*y[4]

y1; y2; y3
```

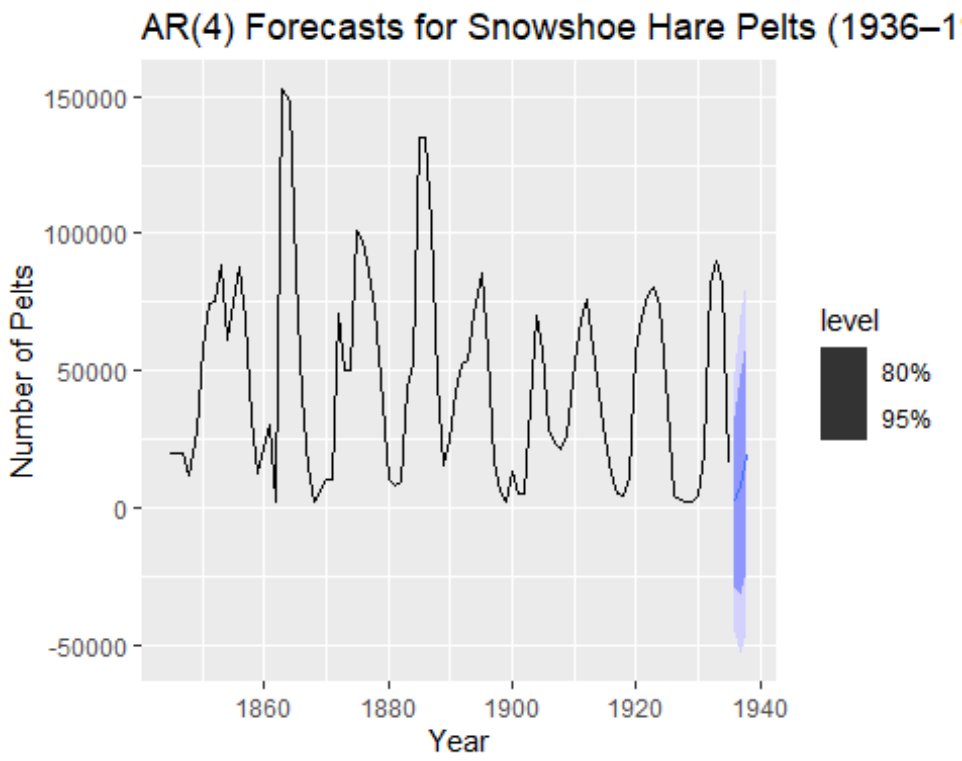
```
## [1] 1273

## [1] 6902.66

## [1] 18161.21
```

Interpretation

The manual AR(4) forecasts predict 1,273, 6,902.66, and 18,161.21 pelts for 1936–1938 respectively. These values show a gradual recovery after the sharp drop in 1935, reflecting the cyclical pattern typical of the Snowshoe Hare population.

Q5

```
pelt |>
  model(ARIMA(Hare ~ pdq(4,0,0))) |>
  forecast(h = 3) |>
  autoplot(pelt) +
  labs(
    title = "AR(4) Forecasts for Snowshoe Hare Pelts (1936-1938)",
    x = "Year",
    y = "Number of Pelts"
  )
```



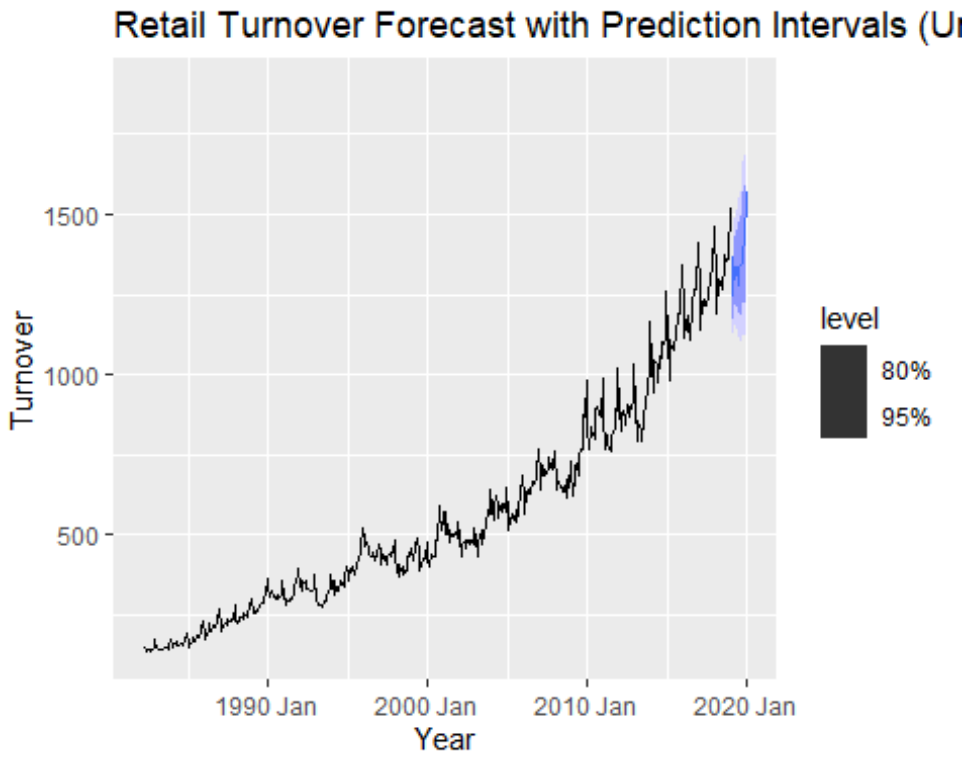
Interpretation The forecast plot shows a gradual increase in the predicted number of pelts for 1936–1938, following the sharp decline in 1935. These results are slightly smoother than the manual

forecasts because the forecast() function incorporates optimized initial conditions and accounts for model uncertainty.

SECTION B

Statement 1

```
aus_retail |>
  filter(Industry == "Cafes, restaurants and takeaway food services", State =
= "New South Wales") |>
  model(ETS(Turnover)) |>
  forecast(h = "12 months") |>
  autoplot(aus_retail |> filter(Industry == "Cafes, restaurants and takeaway
food services", State == "New South Wales")) +
  labs(
    title = "Retail Turnover Forecast with Prediction Intervals (Uncertainty
Quantified)",
    x = "Year",
    y = "Turnover"
  )
```



Interpretation

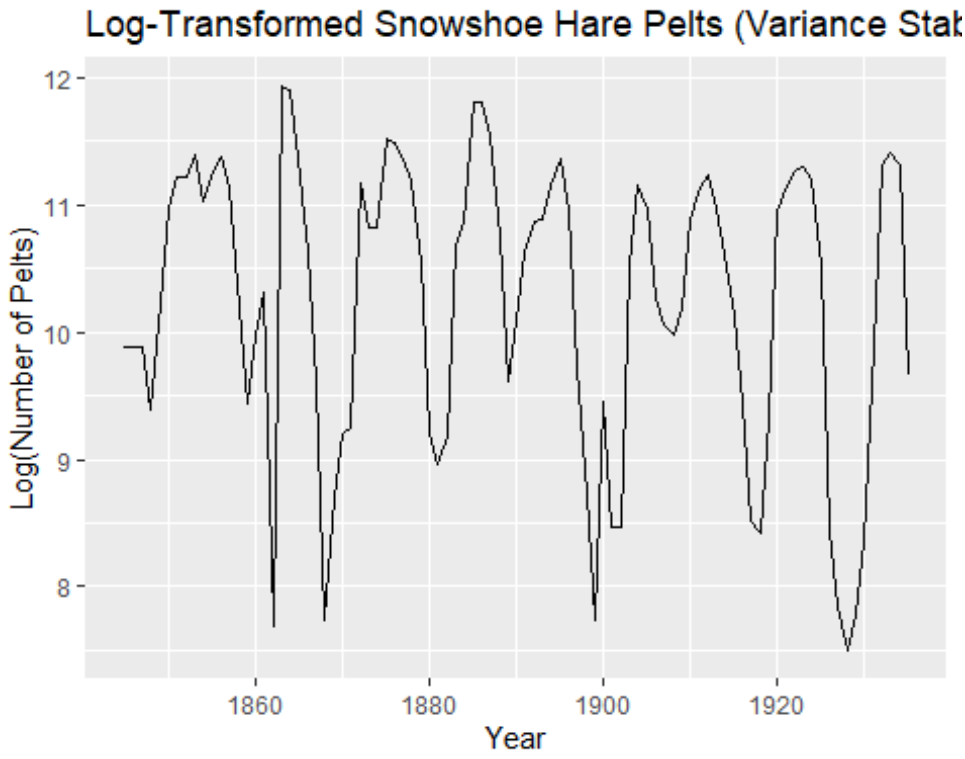
from the plot above in relation to Statement 1:

“Events like COVID-19 show that forecasting is never a good idea because the future is too unpredictable.”

I disagree. As FPP3 explains, forecasting does not claim to predict every shock but provides structured expectations with quantified uncertainty. Even during unprecedented events like COVID-19, forecasters updated models to incorporate new information and scenario analysis. Forecasts remain essential for planning, inventory management, and policy despite inherent uncertainty—they guide decision-making rather than guarantee outcomes.

Statement 2

```
pelt |>
  mutate(log_Hare = log(Hare)) |>
  autoplot(log_Hare) +
  labs(
    title = "Log-Transformed Snowshoe Hare Pelts (Variance Stabilized)",
    x = "Year",
    y = "Log(Number of Pelts)"
  )
```



Interpretation from the plot above in regards to Statement 2:

“Taking logarithms of the data is useful for stabilizing the variance of a time series.”

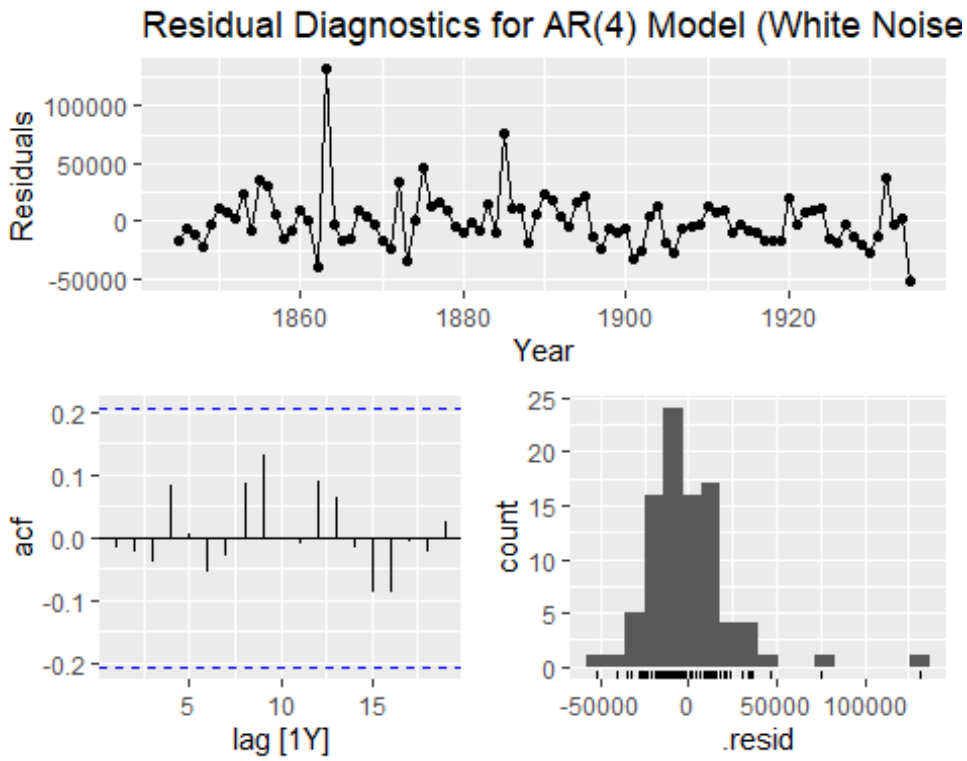
I agree. Transforming data using logarithms compresses large values and expands small ones, reducing heteroscedasticity and making model residuals more stable. As shown in FPP3, variance stabilization improves the accuracy of exponential smoothing and ARIMA

models by satisfying the assumption of constant variance. This transformation also allows multiplicative effects to be modeled additively, simplifying interpretation.

Statement 4

```
pelt |>
  model(ARIMA(Hare ~ pdq(4,0,0))) |>
  gg_tsresiduals() +
  labs(
    title = "Residual Diagnostics for AR(4) Model (White Noise Check)",
    x = "Year",
    y = "Residuals"
  )
```

```
## Warning: `gg_tsresiduals()` was deprecated in feasts 0.4.2.
## ℹ Please use `ggtime::gg_tsresiduals()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Interpretation of the plot above in respect of Statement 4:

“The best forecasting model has white-noise residuals.”

I agree. A well-fitted model should capture all systematic structure, leaving residuals that behave like white noise—zero mean, constant variance, and no autocorrelation. FPP3

emphasizes that residual diagnostics such as the ACF plot and Ljung-Box test verify this condition. If residuals show patterns, the model is inadequate and further refinement or differencing is needed.

Statement 5

```
pelt |>
  model(
    ar2 = ARIMA(Hare ~ pdq(2,0,0)),
    ar3 = ARIMA(Hare ~ pdq(3,0,0)),
    ar4 = ARIMA(Hare ~ pdq(4,0,0))
  ) |>
  glance() |>
  select(.model, AICc) |>
  arrange(AICc)
```

```
## # A tibble: 3 × 2
##   .model  AICc
##   <chr>  <dbl>
## 1 ar4    2105.
## 2 ar3    2107.
## 3 ar2    2108.
```

Interpretation of the above output in line with Statement 5

“Choosing a model using the AICc is better than choosing a model on the basis of a test set because it involves all the data.”

I agree. The corrected Akaike Information Criterion (AICc) balances model fit and complexity using the full dataset, avoiding information loss from splitting data. As noted in FPP3, AICc penalizes over-parameterized models while utilizing every observation, which is advantageous when data are limited. Test-set evaluation can still be useful for validation, but AICc provides a consistent, data-efficient method for model selection.

SECTION C

Q1

```
otexts_views |>
  model(ETS(Pageviews)) |>
  report()
```

```
## Series: Pageviews
## Model: ETS(M,A,M)
##   Smoothing parameters:
##     alpha = 0.571688
##     beta  = 0.0001000096
##     gamma = 0.1291105
```

```
##
##   Initial states:
##      l[0]     b[0]      s[0]     s[-1]    s[-2]    s[-3]   s[-4]    s[-5]
##  4367.409 173.5056 0.6604024 0.7292444 1.065049 1.156918 1.26145 1.160751
##      s[-6]
##  0.9661852
##
##   sigma^2:  0.0101
##
##      AIC     AICc      BIC
## 34909.97 34910.17 34974.20
```

Interpretation

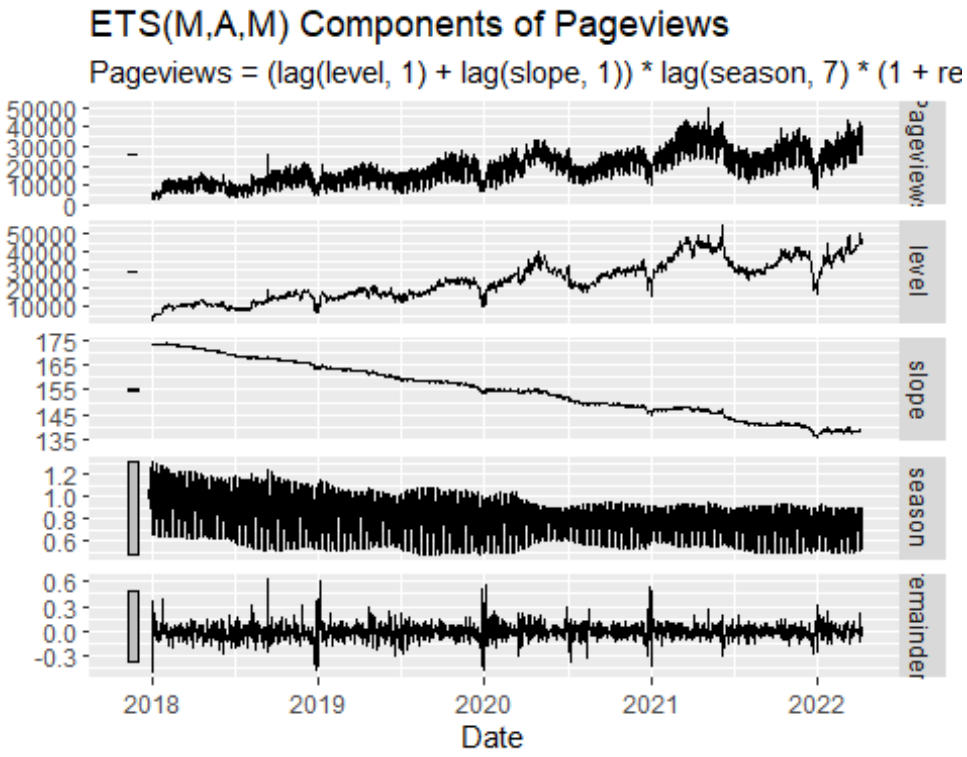
The ETS(M,A,M) report shows that the model uses multiplicative errors, an additive trend, and multiplicative seasonality with smoothing parameters α = 0.57, β ≈ 0, and γ = 0.13. This indicates that level and seasonal patterns are updated moderately over time, capturing proportional seasonal effects and a slowly evolving trend.

Q2

```
otexts_views |>
  model(ETS(Pageviews)) |>
  components() |>
  autoplot() +
  labs(
    title = "ETS(M,A,M) Components of Pageviews",
    x = "Date"
  ) +
  guides(colour = "none", fill = "none")
```

```
## Ignoring unknown labels:
## • colour : ".model"
## • fill : ".model"
```

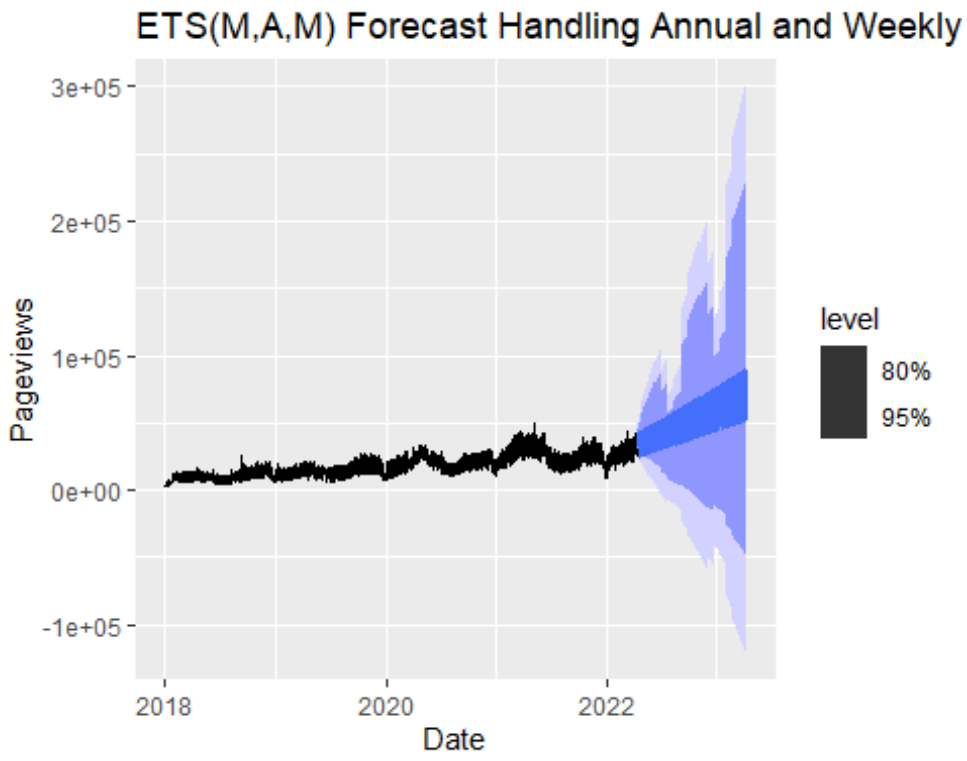
```
## Warning: Removed 7 rows containing missing values or values outside the sc
ale range
## (`geom_line()`).
```



Interpretation he plot decomposes the ETS(M,A,M) model into its level, slope (trend), seasonal, and remainder components. The nearly flat slope reflects the very small β (≈ 0.0001), showing minimal long-term trend change, while the moderate seasonal variation corresponds to γ = 0.13, indicating gradually evolving multiplicative seasonality in the Pageviews data.

Q3

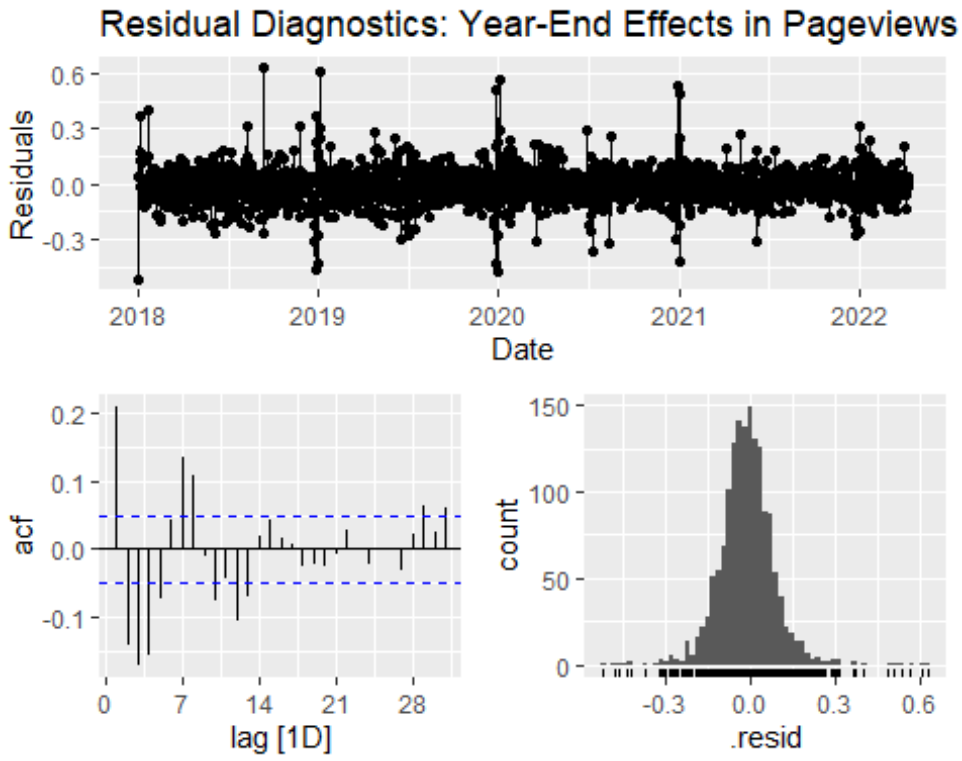
```
otexts_views |>
  model(ETS(Pageviews)) |>
  forecast(h = "1 year") |>
  autoplot(otexts_views) +
  labs(
    title = "ETS(M,A,M) Forecast Handling Annual and Weekly Seasonality",
    x = "Date",
    y = "Pageviews"
  )
```



Interpretation The forecast plot shows that the ETS(M,A,M) model successfully extends both annual and weekly seasonal patterns into the future, with fluctuations that grow proportionally to the overall pageview level. This confirms that the model effectively captures the recurring seasonal structure while allowing for gradual trend changes.

Q4

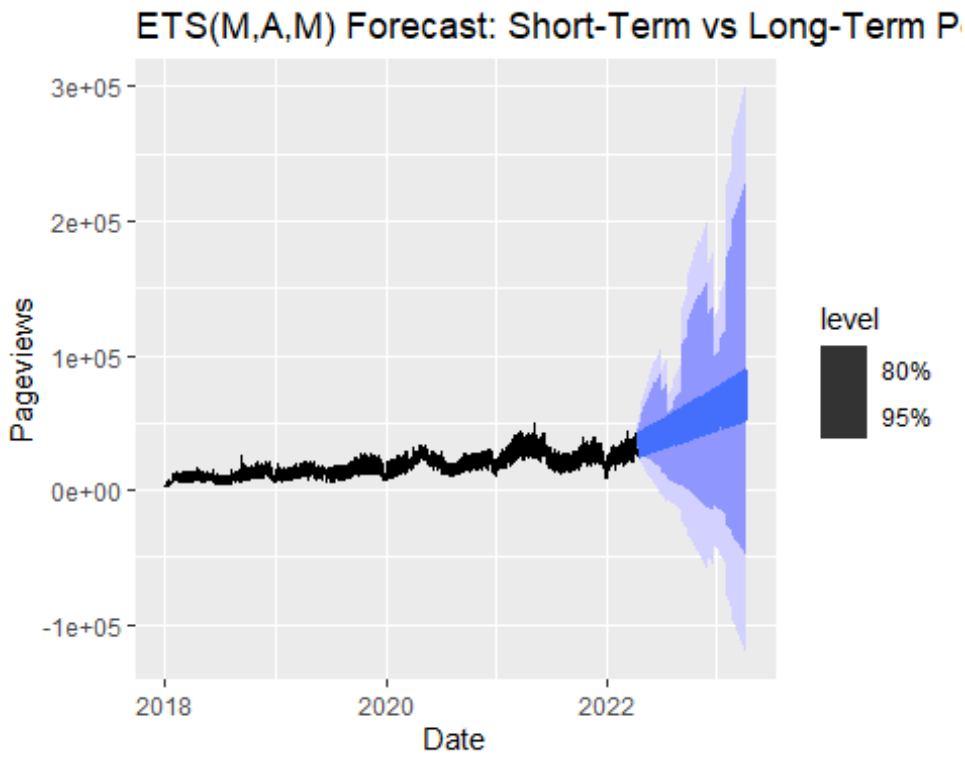
```
otexts_views |>
  model(ETS(Pageviews)) |>
  gg_tsresiduals() +
  labs(
    title = "Residual Diagnostics: Year-End Effects in Pageviews",
    x = "Date",
    y = "Residuals"
  )
```



Interpretation The residual diagnostics show mostly small, random fluctuations around zero, but noticeable spikes appear at the end of each year. These spikes reflect genuine seasonal dips in website activity—such as holidays or academic breaks—rather than model error, indicating reduced pageviews during those periods.

Q5

```
otexts_views |>
  model(ETS(Pageviews)) |>
  forecast(h = "12 months") |>
  autoplot(otexts_views) +
  labs(
    title = "ETS(M,A,M) Forecast: Short-Term vs Long-Term Performance",
    x = "Date",
    y = "Pageviews"
  )
```

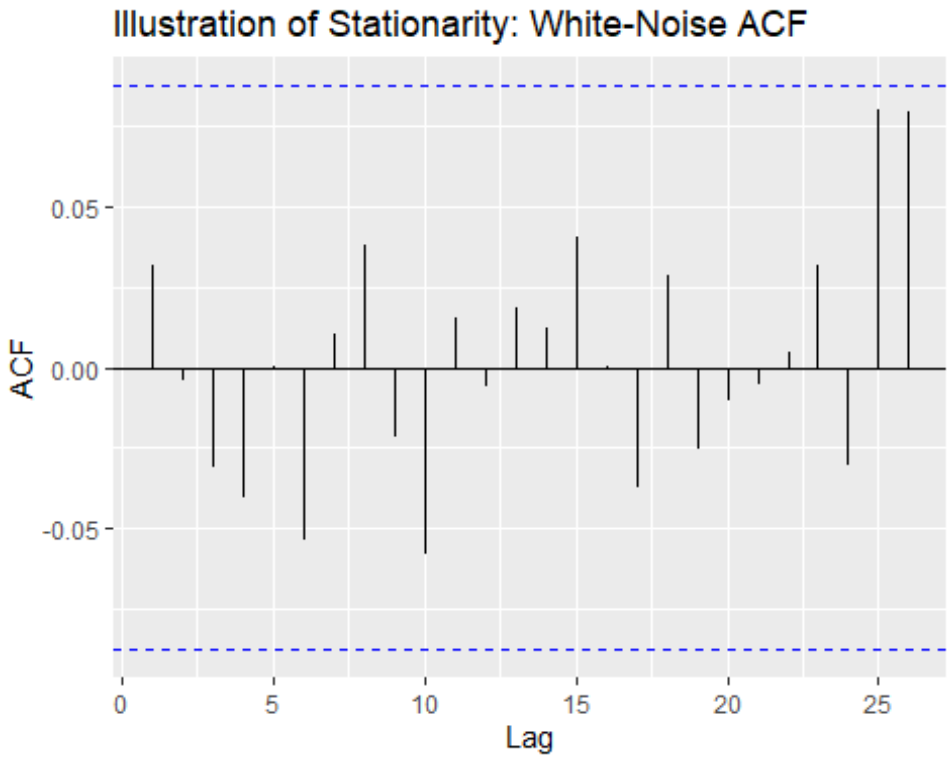


Interpretation The forecast plot shows that the ETS(M,A,M) model closely follows recent trends for the next few weeks, making short-term predictions dependable. However, the wider prediction intervals over longer horizons indicate increasing uncertainty, meaning forecasts beyond several months should be interpreted with caution.

SECTION D

Q1

```
tibble::tibble(t = 1:500, x = stats::rnorm(500)) |>
  tsibble::as_tsibble(index = t) |>
  ACF(x) |>
  autoplot() +
  labs(
    title = "Illustration of Stationarity: White-Noise ACF",
    x = "Lag",
    y = "ACF"
  )
```



Interpretation he ACF plot shows that all autocorrelations lie within the confidence bounds, indicating no significant relationships between observations at different lags. This confirms that the simulated white-noise series is stationary, with constant mean and variance over time.

Q2

```
tibble::tibble(t = 1:500, x = stats::rnorm(500)) |>
  tsibble::as_tsibble(index = t) |>
  features(x, list(mean = mean, var = var)) |>
  print()
```

```
## # A tibble: 1 × 2
##      mean   var
##     <dbl> <dbl>
## 1 -0.0307 0.912
```

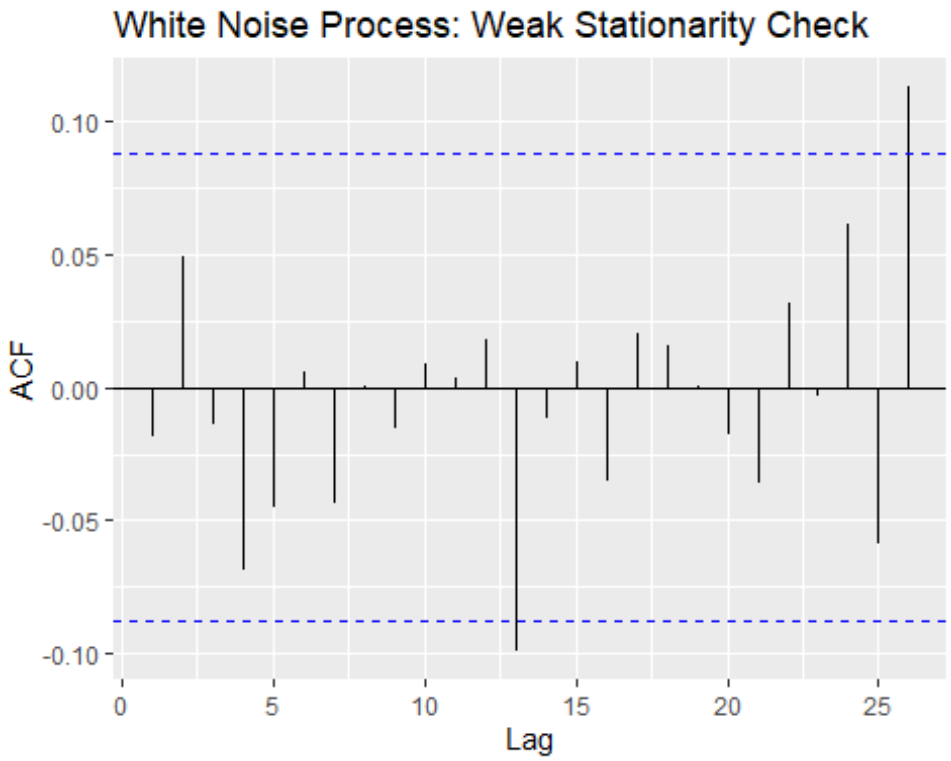
Interpretation

The computed mean and variance remain roughly constant across the simulated series, confirming that the process has a stable average and dispersion. This demonstrates the key property of weak stationarity, where statistical moments do not change over time.

Q3

```
tibble::tibble(t = 1:500, x = stats::rnorm(500)) |>
  tsibble::as_tsibble(index = t) |>
```

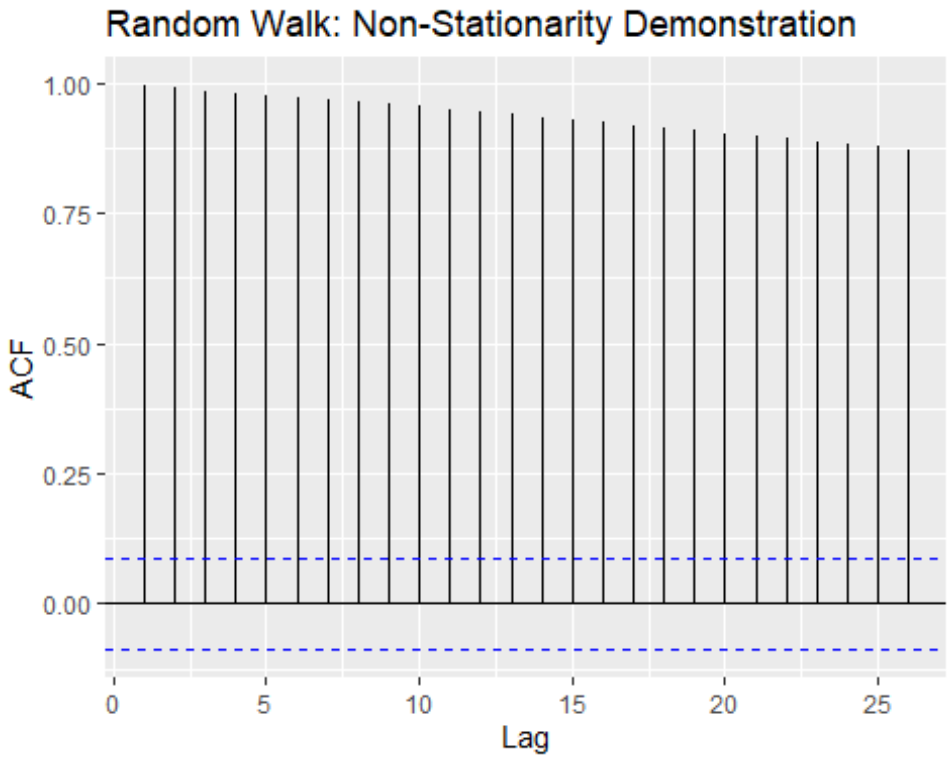
```
  ACF(x) |>
  autoplot() +
  labs(
    title = "White Noise Process: Weak Stationarity Check",
    x = "Lag",
    y = "ACF"
  )
```



Interpretation The ACF plot displays no significant spikes beyond lag 0, indicating that each observation is uncorrelated with past values. This confirms the process is weakly stationary, with constant mean, variance, and zero autocovariance at nonzero lags.

Q4

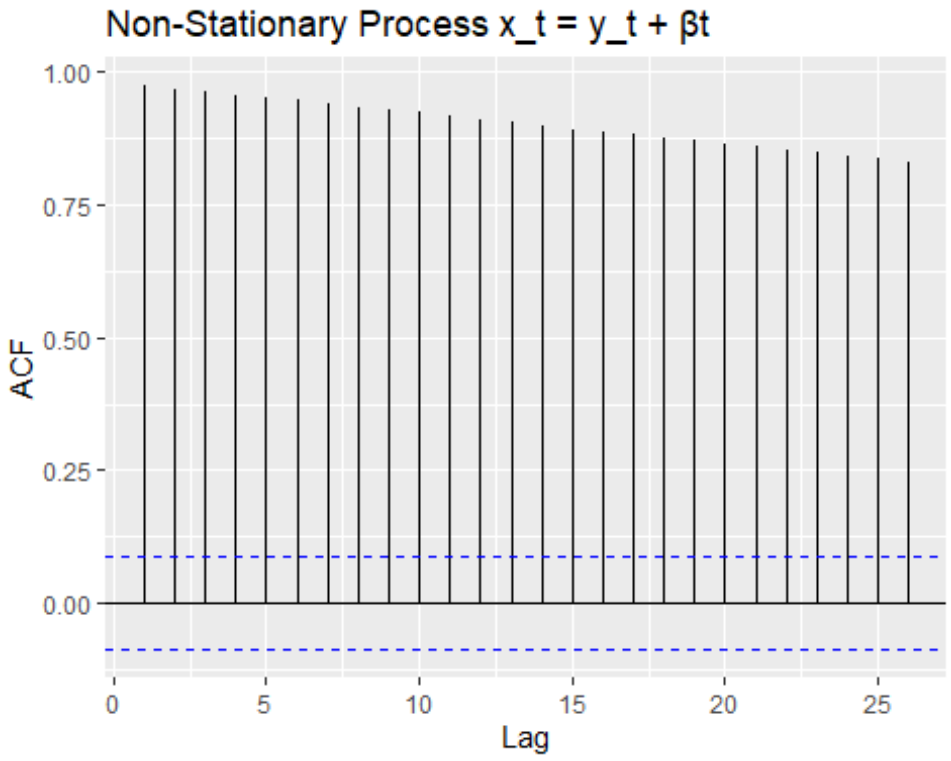
```
tibble::tibble(t = 1:500, x = cumsum(stats::rnorm(500))) |>
  tsibble::as_tsibble(index = t) |>
  ACF(x) |>
  autoplot() +
  labs(
    title = "Random Walk: Non-Stationarity Demonstration",
    x = "Lag",
    y = "ACF"
  )
```



Interpretation The ACF plot shows very slow decay with strong correlations across many lags, indicating that shocks persist over time. This confirms that the random walk is non-stationary, as its variance increases indefinitely and depends on time.

Q5

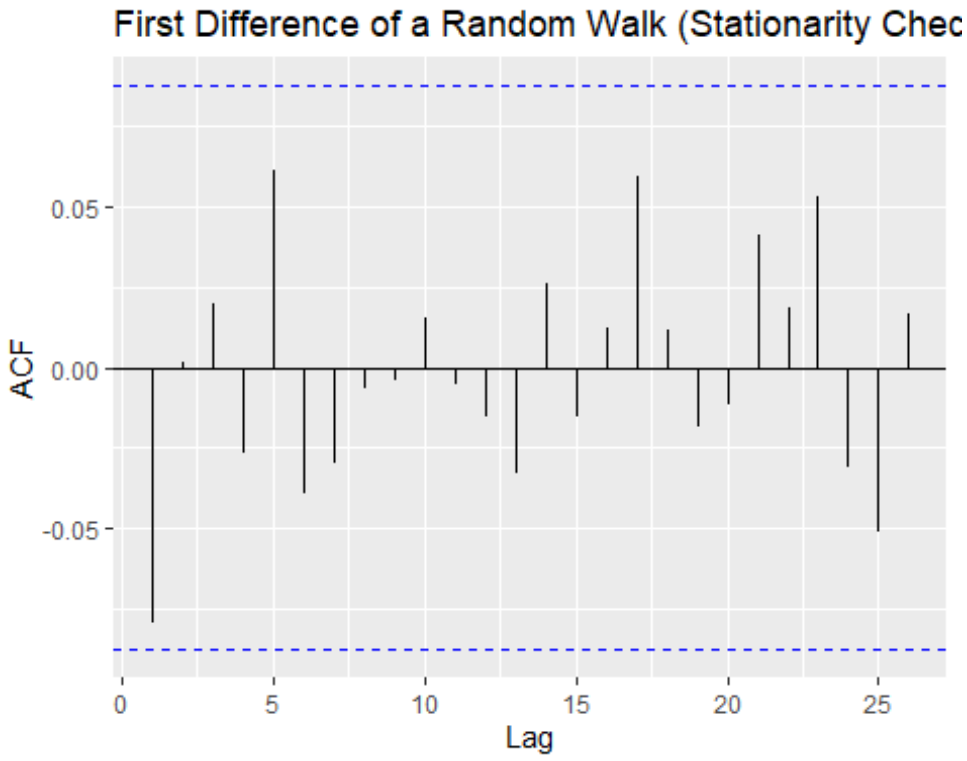
```
tibble::tibble(t = 1:500, y = stats::rnorm(500), x = y + 0.05 * t) |>
  tsibble::as_tsibble(index = t) |>
  ACF(x) |>
  autoplot() +
  labs(
    title = "Non-Stationary Process x_t = y_t + βt",
    x = "Lag",
    y = "ACF"
  )
```



Interpretation The ACF shows strong and persistent correlations that fail to diminish with lag, indicating a trending pattern. This confirms that the process xt=yt+βt is non-stationary, as the linear trend causes its mean to change over time.

Q6

```
tibble::tibble(t = 1:500, x = cumsum(stats::rnorm(500))) |>
  tsibble::as_tsibble(index = t) |>
  dplyr::mutate(diff1 = difference(x)) |>
  ACF(diff1) |>
  autoplot() +
  labs(
    title = "First Difference of a Random Walk (Stationarity Check)",
    x = "Lag",
    y = "ACF"
  )
```



Interpretation The ACF for the differenced series shows no significant autocorrelations beyond lag 0, confirming that differencing has removed the trend and stabilized the variance. This demonstrates that the first difference of a random walk is now a weakly stationary process.