Final Project 443 – Ankwasa Doreen

ANKWASA DOREEN

2025-12-13

Project Title;

Time Series Analysis and Forecasting of Monthly Average Temperature in California Using SARIMA Models.

Define the Objective and Data;

Objective;

The objective of this project is to analyze and forecast monthly average temperatures in California using historical ClimGrid data. The study examines trends, seasonality, and stationarity in the temperature series and evaluates the performance of classical exponential smoothing and ARIMA-based models for short- and long-term forecasting.

Data Description;

Source: NOAA NCEI ClimGrid

Variable ;Monthly average temperature (°C)

Time span:(1895–2023) and Monthly frequency.

Spatial Scope: California . California was chosen as the spatial focus for this analysis for several reasons:

1. Strong seasonal variation: The state exhibits clear and consistent monthly temperature cycles, which are suitable for time series modeling.
2. Policy relevance and familiarity: California's climate patterns are important for environmental planning, agriculture, and energy policy, making the analysis practically useful.
3. Manageable noise: State-level data are less noisy than national aggregates, allowing clearer detection of trends and seasonality.
4. Data completeness: California avoids the extreme sparsity seen in some other states, ensuring a long and reliable historical series.

Exploratory Data Analysis (EDA).

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.4.3
```

```
## Warning: package 'ggplot2' was built under R version 4.4.3
```

```
## Warning: package 'tibble' was built under R version 4.4.3
```

```
## Warning: package 'tidyr' was built under R version 4.4.3
```

```
## Warning: package 'readr' was built under R version 4.4.3
```

```
## Warning: package 'purrr' was built under R version 4.4.3
```

```
## Warning: package 'dplyr' was built under R version 4.4.3
```

```
## Warning: package 'forcats' was built under R version 4.4.3
```

```
## Warning: package 'lubridate' was built under R version 4.4.3
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   4.0.0     ✓ tibble    3.2.1
## ✓ lubridate 1.9.4     ✓ tidyr     1.3.1
## ✓ purrr     1.0.4
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library(lubridate)
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.4.3
```

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

```
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 4.4.3
```

```
library(ggplot2)
```

```
library(terra)
```

```
## Warning: package 'terra' was built under R version 4.4.3
```

```
## terra 1.8.80
```

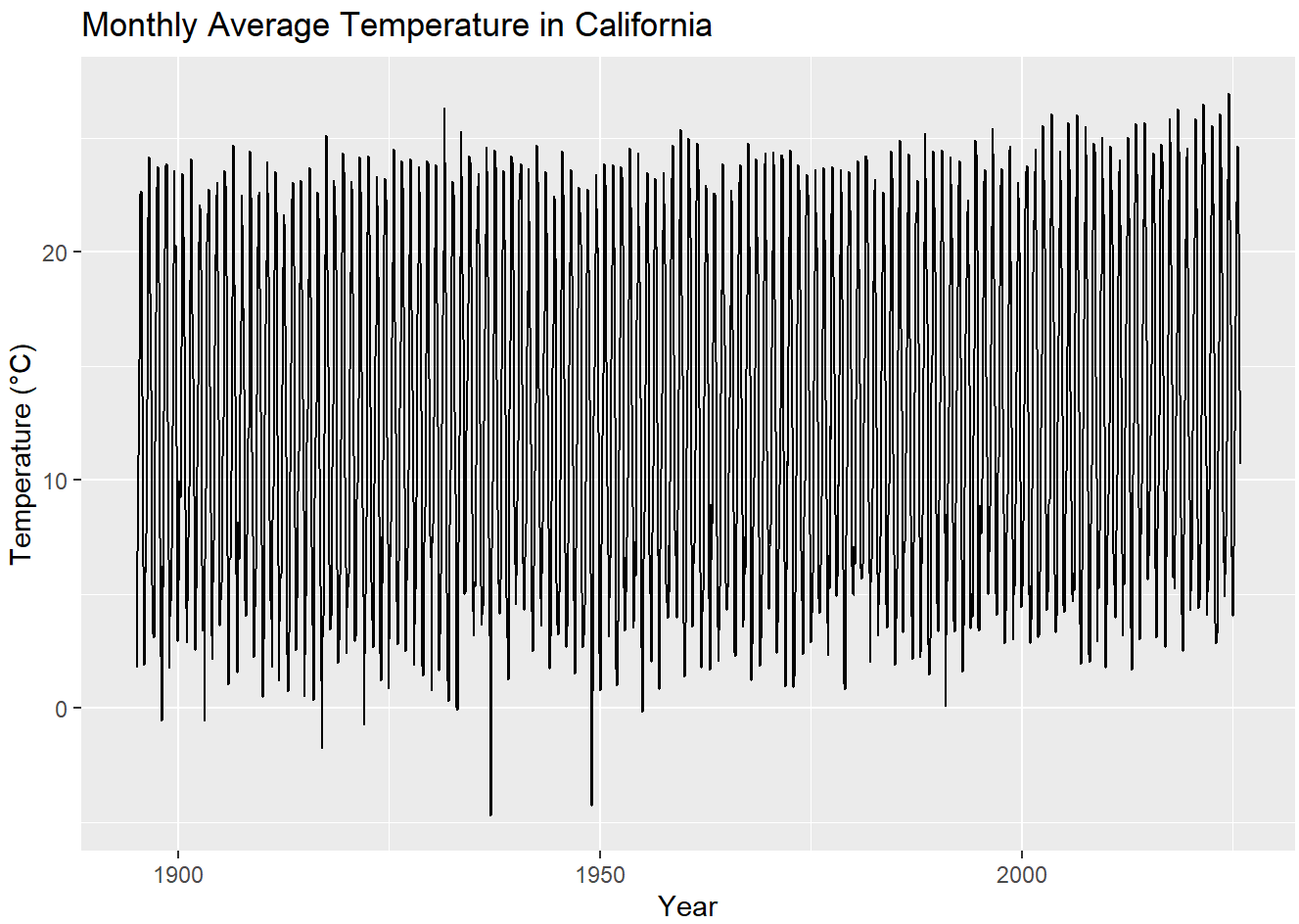
```
##
## Attaching package: 'terra'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(forecast)
library(ggplot2)

r <- rast("C:/Users/magara cliff/Downloads/nclimgrid_tavg.nc")

ca_extent <- ext(-125, -113, 32, 42)

ca_raster <- crop(r, ca_extent)
```

```
## |---------|---------|---------|---------|=========================================
```

```
ca_values <- global(ca_raster, fun = mean, na.rm = TRUE)[,1]

ca_ts <- ts(ca_values, start = c(1895, 1), frequency = 12)
autoplot(ca_ts) +
  labs(
    title = "Monthly Average Temperature in California",
    x = "Year",
    y = "Temperature (°C)"
  )
```



Define California,

```
library(raster)
```

```
## Warning: package 'raster' was built under R version 4.4.3
```

```
## Loading required package: sp
```

```
## Warning: package 'sp' was built under R version 4.4.3
```

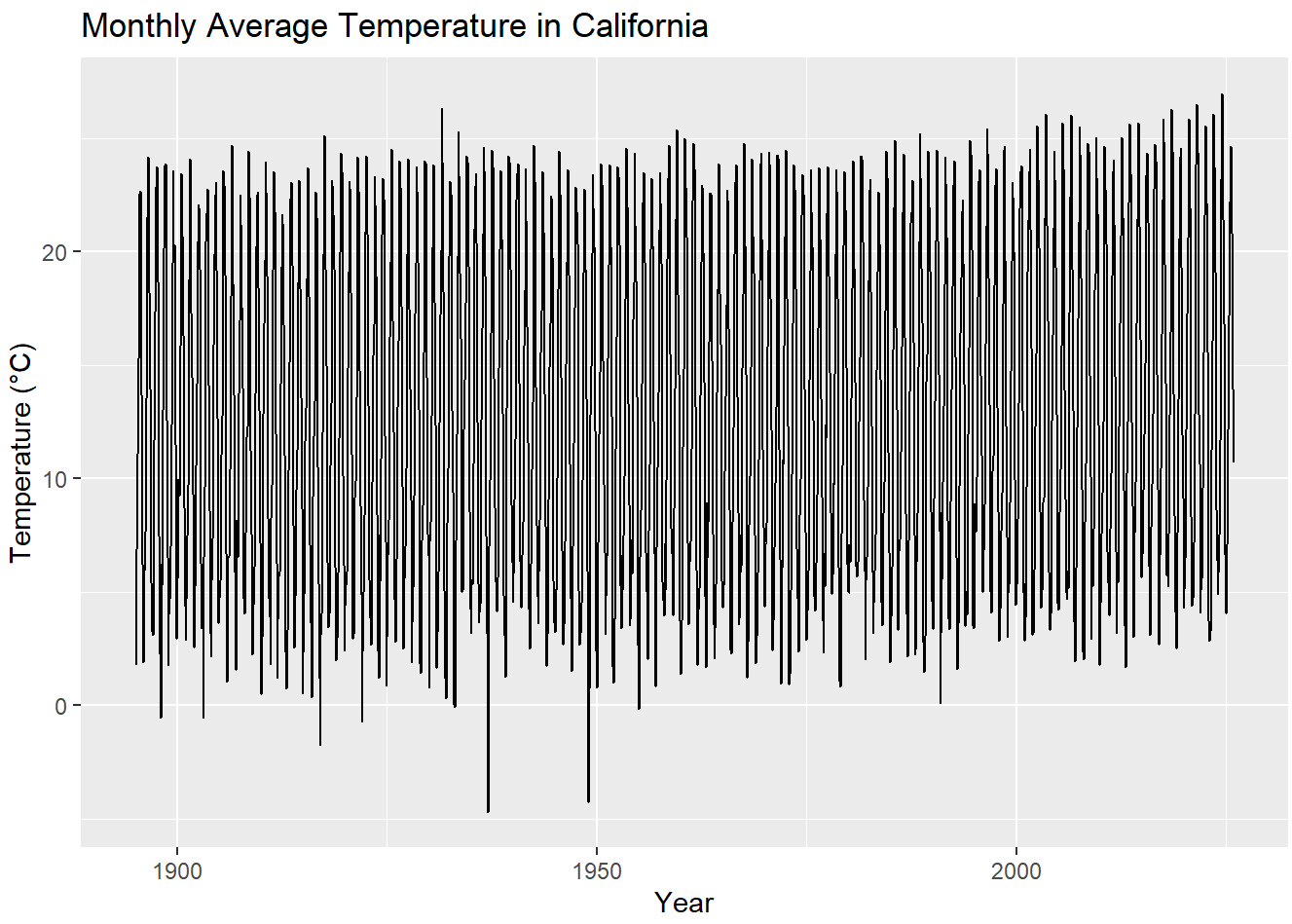
```
##
## Attaching package: 'raster'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(forecast)
ca_raster <- crop(r, ca_extent)
```

```
## |---------|---------|---------|---------|=========================================
```

```
ca_values <- colMeans(as.matrix(ca_raster), na.rm = TRUE)
ca_ts <- ts(ca_values, start=c(1895,1), frequency=12)
autoplot(ca_ts) +
  labs(
    title = "Monthly Average Temperature in California",
    x = "Year",
    y = "Temperature (°C)"
  )
```



i observe Clear seasonal cycles (peaks in summer, lows in winter),Long-term trend.

Summary Statistics by Month (Seasonal Patterns)

```r
library(dplyr)
library(lubridate)
library(tibble)

ca_df <- tibble(
  Year = as.numeric(floor(time(ca_ts))),
  Month = as.numeric(cycle(ca_ts)),
  Temp = as.numeric(ca_ts)
)

# summary statistics by month
monthly_stats <- ca_df %>%
  group_by(Month) %>%
  summarise(
    Mean = mean(Temp, na.rm = TRUE),
    SD = sd(Temp, na.rm = TRUE),
    Min = min(Temp, na.rm = TRUE),
    Max = max(Temp, na.rm = TRUE)
  )

print(monthly_stats)
```

```
## # A tibble: 12 × 5
##    Month  Mean    SD     Min   Max
##    <dbl> <dbl> <dbl>   <dbl> <dbl>
##  1     1  3.21  1.97 -4.69    7.93
##  2     2  5.08  1.81 -0.538   9.50
##  3     3  7.59  1.71  3.65   12.1
##  4     4 11.0   1.54  6.21   14.3
##  5     5 15.3   1.58 11.8    19.1
##  6     6 20.0   1.47 16.8    23.9
##  7     7 24.0   1.09 21.7    27.0
##  8     8 23.1   1.04 20.1    25.9
##  9     9 19.3   1.26 16.2    22.2
## 10    10 13.7   1.44 10.1    17.0
## 11    11  7.60  1.47  3.54   11.0
## 12    12  3.55  1.56  0.0363  6.75
```

This shows average temperatures and variability for each month, highlighting seasonal patterns like in hot summers, cool winters.

```
# summary statistics by year
yearly_stats <- ca_df %>%
  group_by(Year) %>%
  summarise(
    Mean = mean(Temp, na.rm = TRUE),
    SD = sd(Temp, na.rm = TRUE),
    Min = min(Temp, na.rm = TRUE),
    Max = max(Temp, na.rm = TRUE)
  )

print(yearly_stats)
```

```
## # A tibble: 131 × 5
##     Year  Mean    SD    Min   Max
##    <dbl> <dbl> <dbl>  <dbl> <dbl>
##  1  1895  12.0  7.70  1.76   22.7
##  2  1896  12.7  7.27  5.02   24.2
##  3  1897  12.1  8.11  2.27   23.8
##  4  1898  12.1  8.37 -0.538  23.9
##  5  1899  12.2  7.15  2.91   23.6
##  6  1900  12.8  6.57  4.37   23.4
##  7  1901  12.6  7.31  2.90   24.1
##  8  1902  12.1  7.45  2.56   22.1
##  9  1903  11.9  7.90 -0.538  22.8
## 10  1904  12.6  7.40  2.16   23.1
## # ℹ 121 more rows
```

This shows long-term trends,forinstance gradual warming or cooling over decades.

Stationarity Testing;

since i have California monthly temperature time series (ca_ts), the next step is to check stationarity. 1.Augmented Dickey-Fuller (ADF) Test;

The ADF test checks:

H_0: The series has a unit root , non-stationary

H_1: The series is stationary

```
library(tseries)
library(forecast)
adf_result <- adf.test(ca_ts)
```

```
## Warning in adf.test(ca_ts): p-value smaller than printed p-value
```

```
print(adf_result)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  ca_ts
## Dickey-Fuller = -7.9087, Lag order = 11, p-value = 0.01
## alternative hypothesis: stationary
```

2. KPSS Test;

KPSS tests stationarity opposite of ADF:

H_0: Series is stationary

H_1: Series is non-stationary

```
kpss_result <- kpss.test(ca_ts)
print(kpss_result)
```

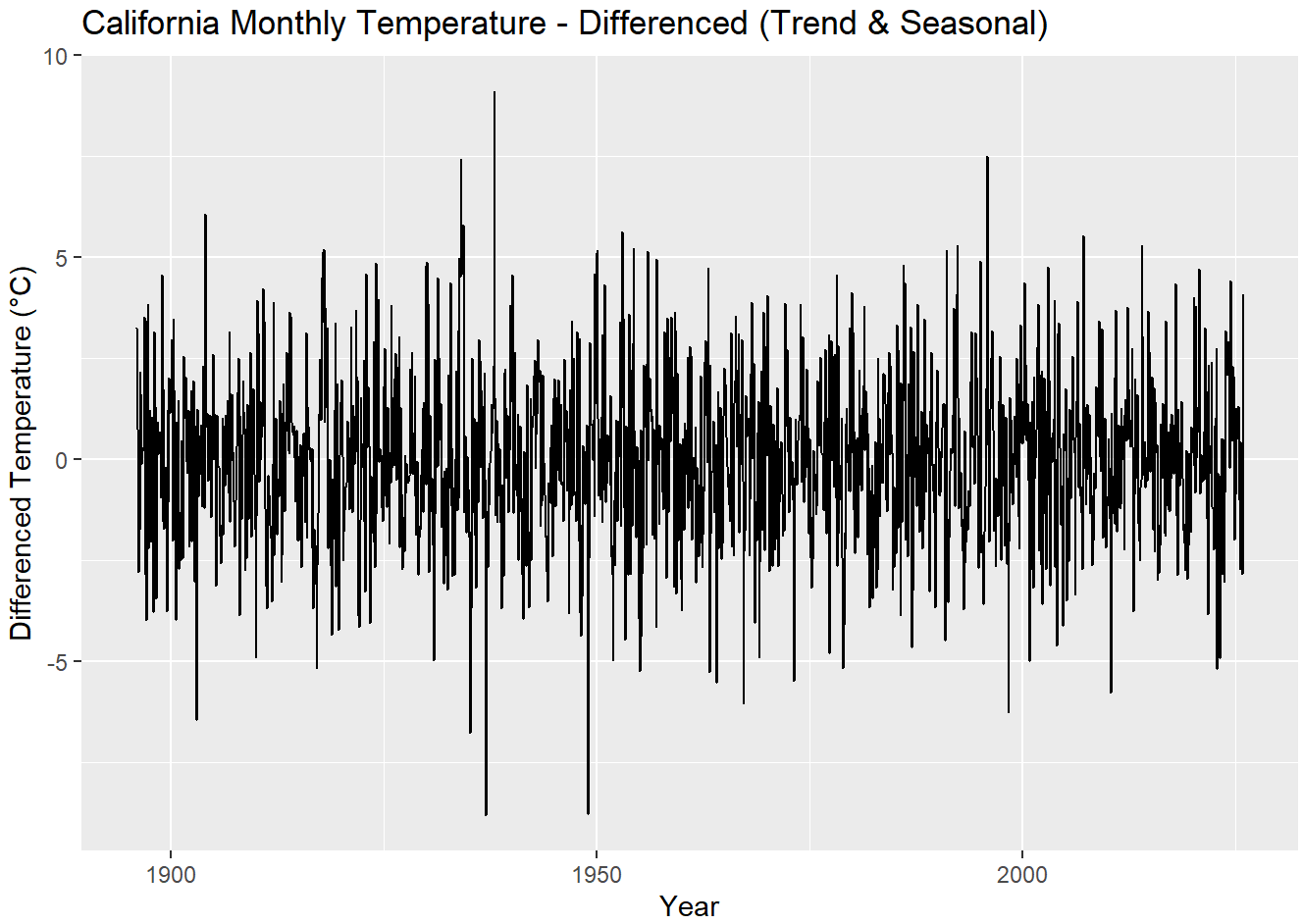
```
##
##  KPSS Test for Level Stationarity
##
## data:  ca_ts
## KPSS Level = 0.48536, Truncation lag parameter = 7, p-value = 0.04496
```

p-value = 0.04135 ≤ 0.05: reject H_0 thus series is non-stationary and hence differencing is needed.

When ADF is stationary and KPSS non-stationary, this means the series has a trend or seasonality.

Since this is monthly data with seasonality, we apply First-order differencing to remove trend and seasonal differencing (lag 12) to achieve stationarity before modeling.

```
ca_diff <- diff(ca_ts, differences = 1, lag = 12)
library(ggplot2)
autoplot(ca_diff) +
  ggtitle("California Monthly Temperature - Differenced (Trend & Seasonal)") +
  ylab("Differenced Temperature (°C)") +
  xlab("Year")
```



The plot shows how Values fluctuate around zero,No obvious trend,Seasonal cycles are removed,Spikes reflect anomalies, not seasonality.

i will check for stationarity after differencing;

```
library(tseries)

adf.test(na.omit(ca_diff))
```

```
## Warning in adf.test(na.omit(ca_diff)): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  na.omit(ca_diff)
## Dickey-Fuller = -21.217, Lag order = 11, p-value = 0.01
## alternative hypothesis: stationary
```

```
kpss.test(na.omit(ca_diff))
```

```
## Warning in kpss.test(na.omit(ca_diff)): p-value greater than printed p-value
```

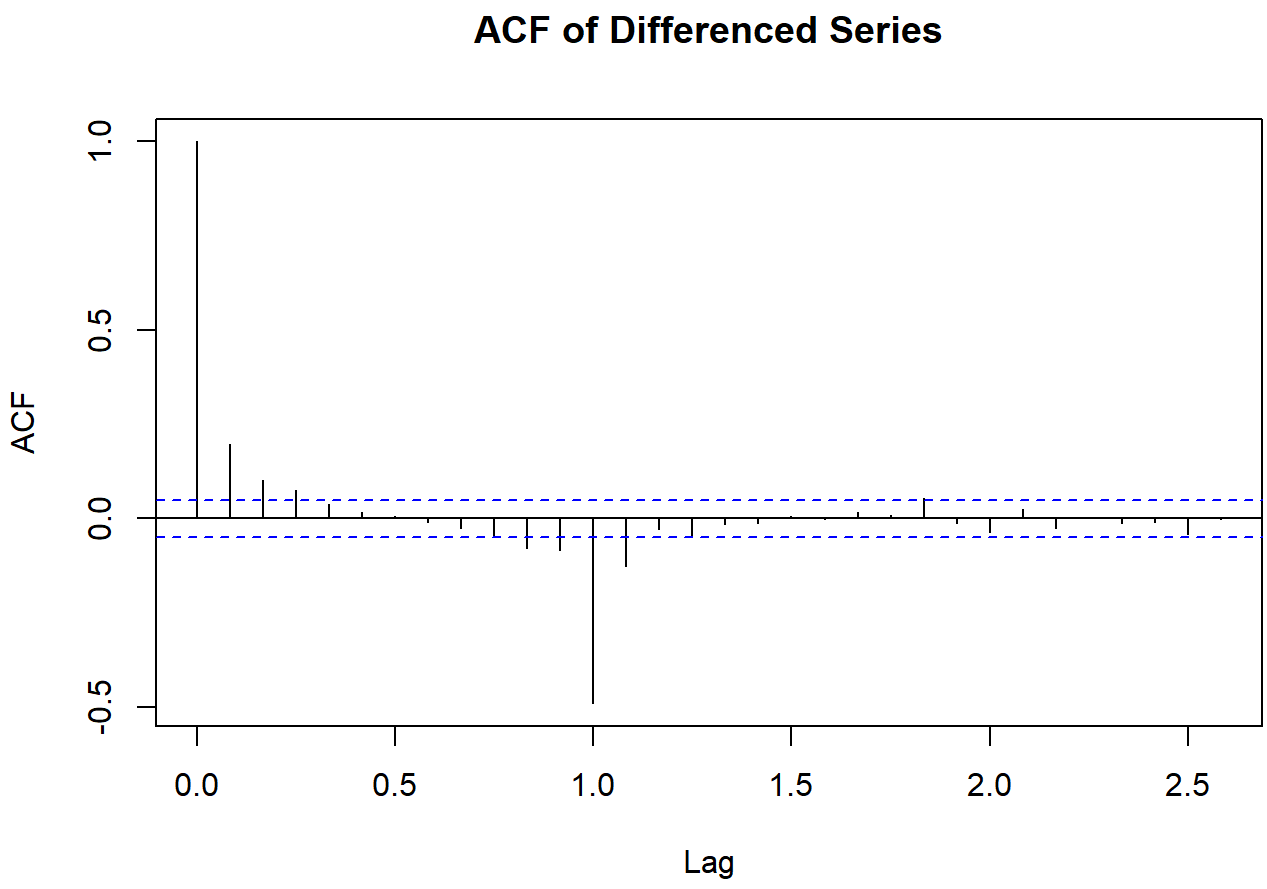
```
##
##  KPSS Test for Level Stationarity
##
## data:  na.omit(ca_diff)
## KPSS Level = 0.0069166, Truncation lag parameter = 7, p-value = 0.1
```

interpretation. ADF Test shows p-value = 0.01 < 0.05 , reject H_0 thus series is stationary.KPSS Test,p-value = 0.1 > 0.05 , fail to reject H_0 hence series is stationary.since both tests are stationary after differencing, this means trend and seasonal effects have been removed and the series is ready for SARIMA modeling.

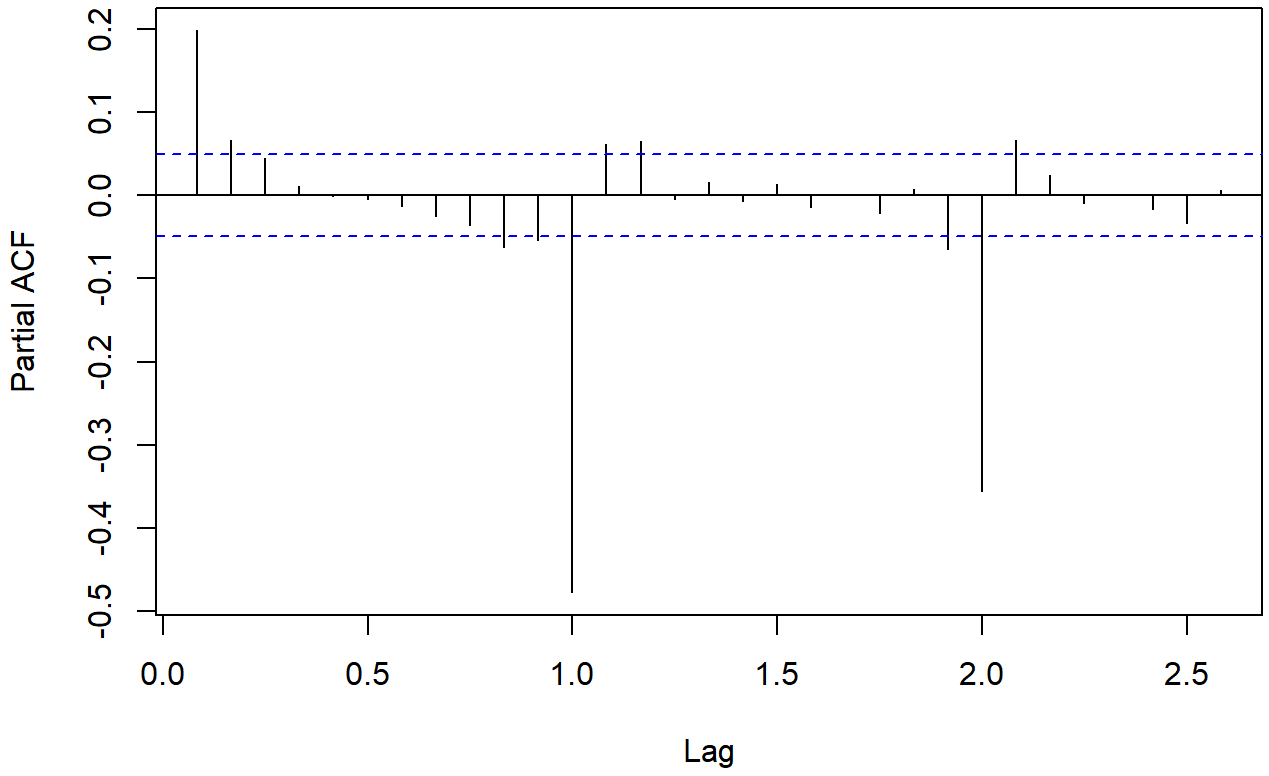
#Model Selection;

SARIMA Model Fitting;

```
acf(na.omit(ca_diff), main="ACF of Differenced Series")
```




```
pacf(na.omit(ca_diff), main="PACF of Differenced Series")
```



interpretation; ACF (Autocorrelation Function);

Shows significant spike at lag 1 → suggests MA(1) component

No strong decay after lag 1 → non-seasonal MA(1)

Check for seasonal spikes at lag 12, 24, etc. → indicates seasonal component

PACF (Partial Autocorrelation Function);

Significant spike at lag 1 → suggests AR(1) component

Seasonal spike at lag 12 → suggests seasonal AR(1)

using auto.arima() for automatic selection;

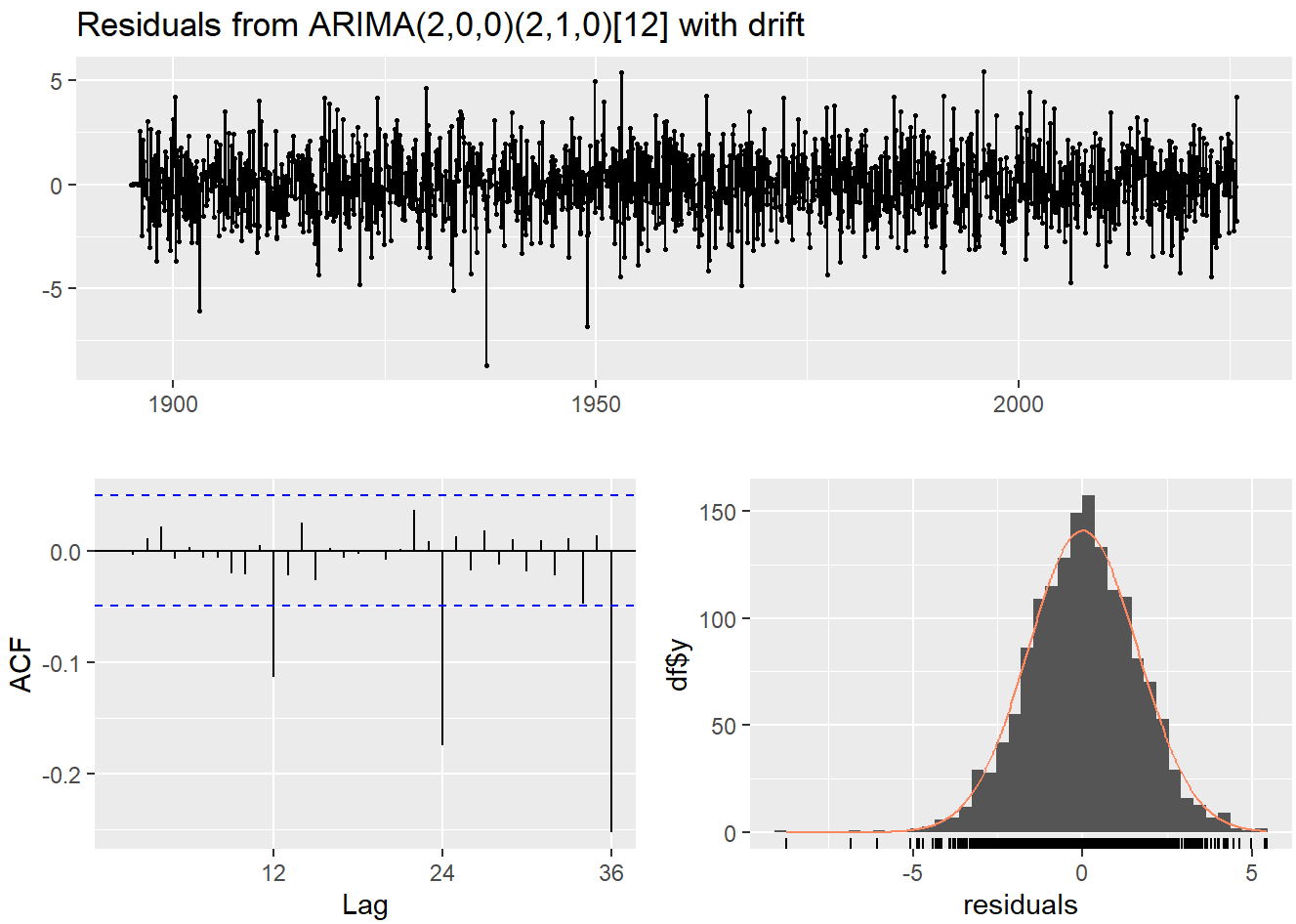
```
library(forecast)
sarima_model <- auto.arima(ca_ts, seasonal = TRUE)
summary(sarima_model)
```

```
## Series: ca_ts
## ARIMA(2,0,0)(2,1,0)[12] with drift
##
## Coefficients:
##          ar1     ar2     sar1     sar2   drift
##       0.1729  0.0801  -0.6721  -0.3734  0.0012
## s.e.  0.0253  0.0253   0.0236   0.0237  0.0023
##
## sigma^2 = 2.644:  log likelihood = -2970.94
## AIC=5953.88   AICc=5953.93   BIC=5985.99
##
## Training set error measures:
##                       ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 0.001736348 1.617162 1.264459 -7.82829 35.36106 0.7965286
##                       ACF1
## Training set -0.0005358649
```

The seasonal AR terms are strong because temperature has yearly seasonality.based on AICc automatically.

Residual Analysis for SARIMA Model;

```
library(forecast)
library(ggplot2)

sarima_model <- auto.arima(ca_ts, seasonal = TRUE)

checkresiduals(sarima_model)
```

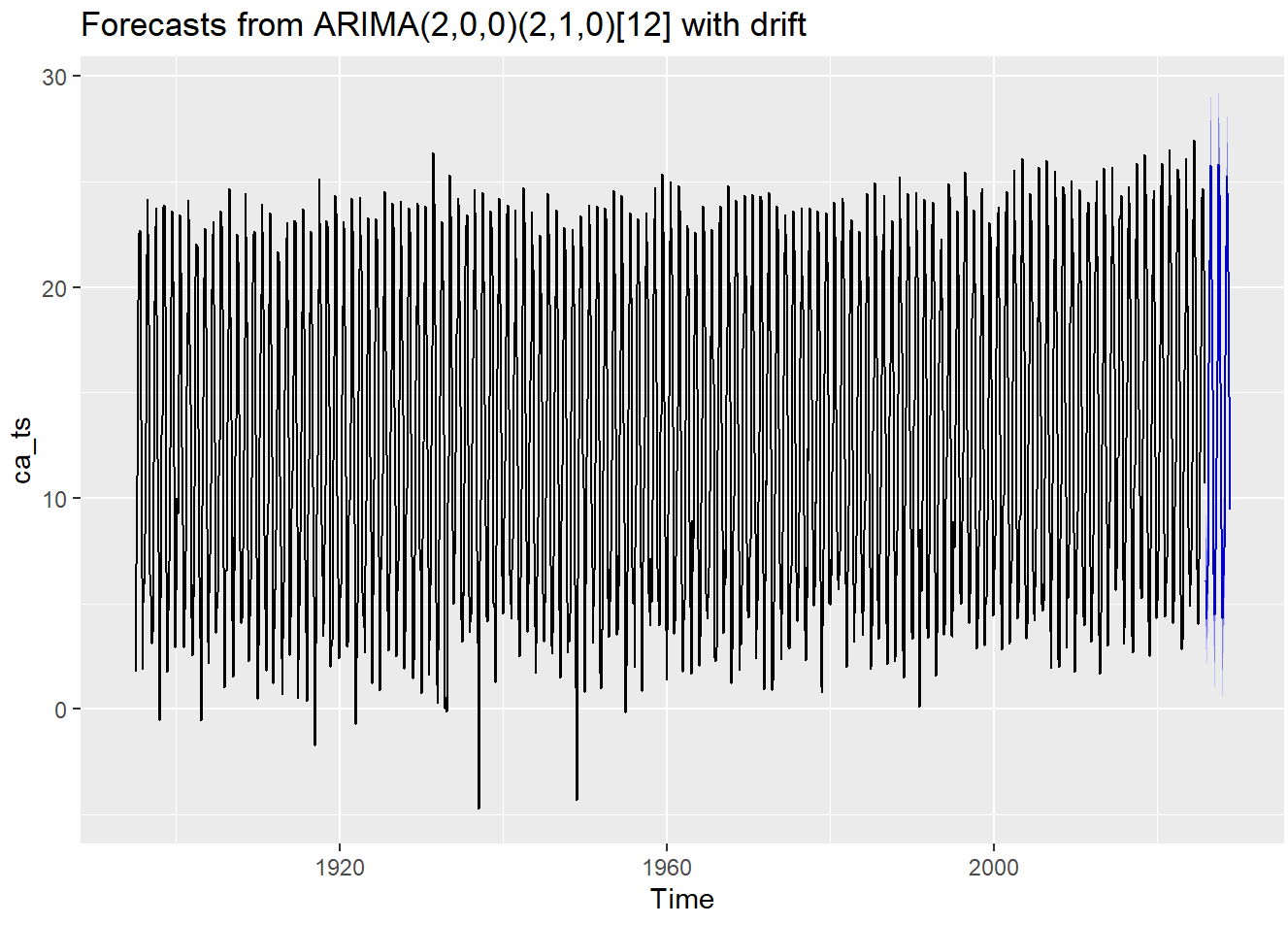



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,0)(2,1,0)[12] with drift
## Q* = 76.774, df = 20, p-value = 1.376e-08
##
## Model df: 4.   Total lags used: 24
```

```
library(forecast)

sarima_model <- auto.arima(ca_ts, seasonal = TRUE)
summary(sarima_model)
```

```
## Series: ca_ts
## ARIMA(2,0,0)(2,1,0)[12] with drift
##
## Coefficients:
##          ar1     ar2     sar1     sar2   drift
##       0.1729  0.0801  -0.6721  -0.3734  0.0012
## s.e.  0.0253  0.0253   0.0236   0.0237  0.0023
##
## sigma^2 = 2.644:  log likelihood = -2970.94
## AIC=5953.88   AICc=5953.93   BIC=5985.99
##
## Training set error measures:
##                       ME     RMSE      MAE      MPE     MAPE      MASE
## Training set 0.001736348 1.617162 1.264459 -7.82829 35.36106 0.7965286
##                       ACF1
## Training set -0.0005358649
```

```
forecast_vals <- forecast::forecast(sarima_model, h = 36)
autoplot(forecast_vals)
```




```
forecast_vals$lower
```

```
##                80%        95%
## Dec 2025  3.991880  2.8887919
## Jan 2026  2.168526  1.0490675
## Feb 2026  3.312747  2.1867297
## Mar 2026  4.676230  3.5496285
## Apr 2026  9.607127  8.4804123
## May 2026 14.284867 13.1581376
## Jun 2026 19.118825 17.9920940
## Jul 2026 23.662151 22.5354196
## Aug 2026 22.100102 20.9733706
## Sep 2026 18.280615 17.1538831
## Oct 2026 13.021223 11.8944918
## Nov 2026  6.625560  5.4988288
## Dec 2026  4.189529  3.0061604
## Jan 2027  2.235602  1.0505822
## Feb 2027  3.879026  2.6933379
## Mar 2027  5.331335  4.1455867
## Apr 2027  9.639132  8.4533723
## May 2027 14.227529 13.0417676
## Jun 2027 20.046458 18.8606967
## Jul 2027 23.576445 22.3906840
## Aug 2027 22.147012 20.9612503
## Sep 2027 18.709208 17.5234464
## Oct 2027 13.214624 12.0288627
## Nov 2027  6.342555  5.1567939
## Dec 2027  4.075714  2.8081150
## Jan 2028  1.883196  0.6132304
## Feb 2028  3.915312  2.6443901
## Mar 2028  5.264604  3.9935970
## Apr 2028  9.578306  8.3072829
## May 2028 14.346518 13.0754925
## Jun 2028 19.569191 18.2981648
## Jul 2028 22.861219 21.5901935
## Aug 2028 22.069439 20.7984134
## Sep 2028 18.368179 17.0971529
## Oct 2028 12.441707 11.1706814
## Nov 2028  7.057623  5.7865966
```

```
forecast_vals$upper
```

```
##                80%       95%
## Dec 2025  8.159444  9.262532
## Jan 2026  6.397941  7.517399
## Feb 2026  7.566942  8.692959
## Mar 2026  8.932630 10.059232
## Apr 2026 13.863957 14.990671
## May 2026 18.541750 19.668479
## Jun 2026 23.375717 24.502449
## Jul 2026 27.919045 29.045776
## Aug 2026 26.356996 27.483727
## Sep 2026 22.537508 23.664240
## Oct 2026 17.278117 18.404849
## Nov 2026 10.882454 12.009186
## Dec 2026  8.660401  9.843769
## Jan 2027  6.712716  7.897736
## Feb 2027  8.358663  9.544351
## Mar 2027  9.811197 10.996945
## Apr 2027 14.119038 15.304797
## May 2027 18.707440 19.893201
## Jun 2027 24.526370 25.712131
## Jul 2027 28.056358 29.242119
## Aug 2027 26.626924 27.812685
## Sep 2027 23.189120 24.374881
## Oct 2027 17.694536 18.880298
## Nov 2027 10.822468 12.008229
## Dec 2027  8.864818 10.132417
## Jan 2028  6.681238  7.951203
## Feb 2028  8.716968  9.987889
## Mar 2028 10.066583 11.337590
## Apr 2028 14.380348 15.651371
## May 2028 19.148567 20.419593
## Jun 2028 24.371241 25.642267
## Jul 2028 27.663270 28.934296
## Aug 2028 26.871490 28.142516
## Sep 2028 23.170230 24.441256
## Oct 2028 17.243758 18.514784
## Nov 2028 11.859673 13.130699
```

```
forecast_vals$level   # confidence levels
```

```
## [1] 80 95
```

```
sarima_model2 <- auto.arima(ca_ts, seasonal=TRUE, stepwise=FALSE, approximation=FALSE, trace=TRU
E)
```

```
##
##  ARIMA(0,0,0)(0,1,0)[12]                    : 6668.479
##  ARIMA(0,0,0)(0,1,0)[12] with drift         : 6670.362
##  ARIMA(0,0,0)(0,1,1)[12]                    : Inf
##  ARIMA(0,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(0,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,0)(1,1,0)[12]                    : 6237.805
##  ARIMA(0,0,0)(1,1,0)[12] with drift         : 6239.525
##  ARIMA(0,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(0,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,0)(1,1,2)[12]                    : Inf
##  ARIMA(0,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,0)(2,1,0)[12]                    : 6014.306
##  ARIMA(0,0,0)(2,1,0)[12] with drift         : 6015.861
##  ARIMA(0,0,0)(2,1,1)[12]                    : Inf
##  ARIMA(0,0,0)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,0)(2,1,2)[12]                    : Inf
##  ARIMA(0,0,0)(2,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(0,1,0)[12]                    : 6616.155
##  ARIMA(0,0,1)(0,1,0)[12] with drift         : 6618.062
##  ARIMA(0,0,1)(0,1,1)[12]                    : Inf
##  ARIMA(0,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,1)(0,1,2)[12]                    : Inf
##  ARIMA(0,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(1,1,0)[12]                    : 6192.033
##  ARIMA(0,0,1)(1,1,0)[12] with drift         : 6193.814
##  ARIMA(0,0,1)(1,1,1)[12]                    : Inf
##  ARIMA(0,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,1)(1,1,2)[12]                    : Inf
##  ARIMA(0,0,1)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,1)(2,1,0)[12]                    : 5968.978
##  ARIMA(0,0,1)(2,1,0)[12] with drift         : 5970.632
##  ARIMA(0,0,1)(2,1,1)[12]                    : Inf
##  ARIMA(0,0,1)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,1)(2,1,2)[12]                    : Inf
##  ARIMA(0,0,1)(2,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(0,1,0)[12]                    : 6608.202
##  ARIMA(0,0,2)(0,1,0)[12] with drift         : 6610.121
##  ARIMA(0,0,2)(0,1,1)[12]                    : Inf
##  ARIMA(0,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,2)(0,1,2)[12]                    : Inf
##  ARIMA(0,0,2)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(1,1,0)[12]                    : 6183.537
##  ARIMA(0,0,2)(1,1,0)[12] with drift         : 6185.346
##  ARIMA(0,0,2)(1,1,1)[12]                    : Inf
##  ARIMA(0,0,2)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,2)(1,1,2)[12]                    : Inf
##  ARIMA(0,0,2)(1,1,2)[12] with drift         : Inf
##  ARIMA(0,0,2)(2,1,0)[12]                    : 5955.508
##  ARIMA(0,0,2)(2,1,0)[12] with drift         : 5957.212
##  ARIMA(0,0,2)(2,1,1)[12]                    : Inf
```

```
##  ARIMA(0,0,2)(2,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(0,1,0)[12]                    : 6604.21
##  ARIMA(0,0,3)(0,1,0)[12] with drift         : 6606.137
##  ARIMA(0,0,3)(0,1,1)[12]                    : Inf
##  ARIMA(0,0,3)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(0,1,2)[12]                    : Inf
##  ARIMA(0,0,3)(0,1,2)[12] with drift         : Inf
##  ARIMA(0,0,3)(1,1,0)[12]                    : 6181.876
##  ARIMA(0,0,3)(1,1,0)[12] with drift         : 6183.702
##  ARIMA(0,0,3)(1,1,1)[12]                    : Inf
##  ARIMA(0,0,3)(1,1,1)[12] with drift         : Inf
##  ARIMA(0,0,3)(2,1,0)[12]                    : 5955.342
##  ARIMA(0,0,3)(2,1,0)[12] with drift         : 5957.067
##  ARIMA(0,0,4)(0,1,0)[12]                    : 6604.496
##  ARIMA(0,0,4)(0,1,0)[12] with drift         : 6606.43
##  ARIMA(0,0,4)(0,1,1)[12]                    : Inf
##  ARIMA(0,0,4)(0,1,1)[12] with drift         : Inf
##  ARIMA(0,0,4)(1,1,0)[12]                    : 6182.294
##  ARIMA(0,0,4)(1,1,0)[12] with drift         : 6184.132
##  ARIMA(0,0,5)(0,1,0)[12]                    : 6606.173
##  ARIMA(0,0,5)(0,1,0)[12] with drift         : 6608.112
##  ARIMA(1,0,0)(0,1,0)[12]                    : 6608.019
##  ARIMA(1,0,0)(0,1,0)[12] with drift         : 6609.934
##  ARIMA(1,0,0)(0,1,1)[12]                    : Inf
##  ARIMA(1,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(1,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,0)(1,1,0)[12]                    : 6184.738
##  ARIMA(1,0,0)(1,1,0)[12] with drift         : 6186.538
##  ARIMA(1,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(1,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,0)(1,1,2)[12]                    : Inf
##  ARIMA(1,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(1,0,0)(2,1,0)[12]                    : 5960.214
##  ARIMA(1,0,0)(2,1,0)[12] with drift         : 5961.902
##  ARIMA(1,0,0)(2,1,1)[12]                    : Inf
##  ARIMA(1,0,0)(2,1,1)[12] with drift         : Inf
##  ARIMA(1,0,0)(2,1,2)[12]                    : Inf
##  ARIMA(1,0,0)(2,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(0,1,0)[12]                    : 6600.778
##  ARIMA(1,0,1)(0,1,0)[12] with drift         : 6602.711
##  ARIMA(1,0,1)(0,1,1)[12]                    : Inf
##  ARIMA(1,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,1)(0,1,2)[12]                    : Inf
##  ARIMA(1,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(1,1,0)[12]                    : 6178.15
##  ARIMA(1,0,1)(1,1,0)[12] with drift         : 6179.991
##  ARIMA(1,0,1)(1,1,1)[12]                    : Inf
##  ARIMA(1,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,1)(1,1,2)[12]                    : Inf
##  ARIMA(1,0,1)(1,1,2)[12] with drift         : Inf
##  ARIMA(1,0,1)(2,1,0)[12]                    : 5952
```

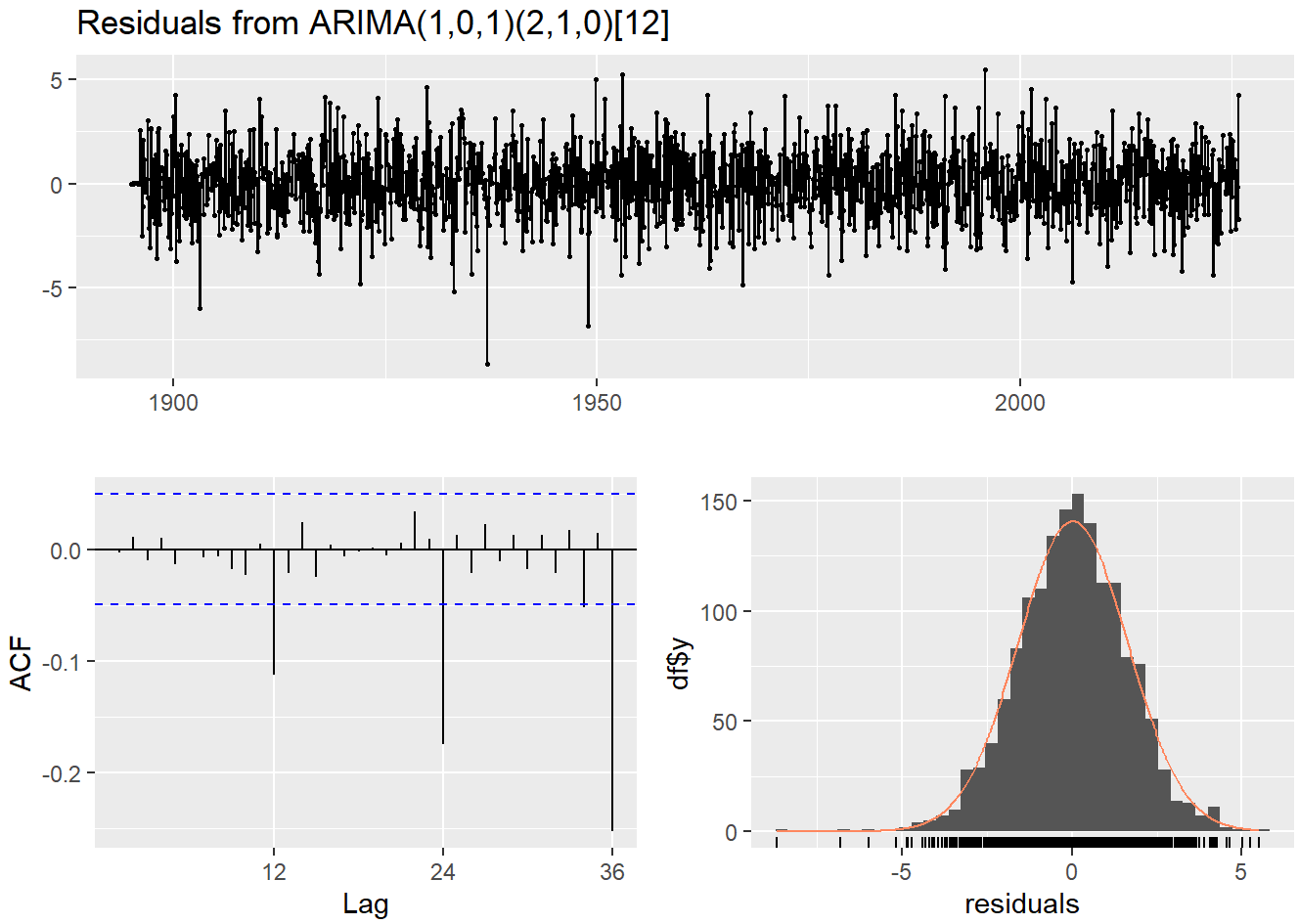
```
##  ARIMA(1,0,1)(2,1,0)[12] with drift         : 5953.752
##  ARIMA(1,0,1)(2,1,1)[12]                    : Inf
##  ARIMA(1,0,1)(2,1,1)[12] with drift         : Inf
##  ARIMA(1,0,2)(0,1,0)[12]                    : 6602.728
##  ARIMA(1,0,2)(0,1,0)[12] with drift         : 6604.664
##  ARIMA(1,0,2)(0,1,1)[12]                    : Inf
##  ARIMA(1,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,2)(0,1,2)[12]                    : Inf
##  ARIMA(1,0,2)(0,1,2)[12] with drift         : Inf
##  ARIMA(1,0,2)(1,1,0)[12]                    : 6180.142
##  ARIMA(1,0,2)(1,1,0)[12] with drift         : 6181.986
##  ARIMA(1,0,2)(1,1,1)[12]                    : Inf
##  ARIMA(1,0,2)(1,1,1)[12] with drift         : Inf
##  ARIMA(1,0,2)(2,1,0)[12]                    : 5953.73
##  ARIMA(1,0,2)(2,1,0)[12] with drift         : 5955.479
##  ARIMA(1,0,3)(0,1,0)[12]                    : 6604.111
##  ARIMA(1,0,3)(0,1,0)[12] with drift         : 6606.052
##  ARIMA(1,0,3)(0,1,1)[12]                    : Inf
##  ARIMA(1,0,3)(0,1,1)[12] with drift         : Inf
##  ARIMA(1,0,3)(1,1,0)[12]                    : 6182.091
##  ARIMA(1,0,3)(1,1,0)[12] with drift         : 6183.937
##  ARIMA(1,0,4)(0,1,0)[12]                    : 6606.094
##  ARIMA(1,0,4)(0,1,0)[12] with drift         : 6608.039
##  ARIMA(2,0,0)(0,1,0)[12]                    : 6603.32
##  ARIMA(2,0,0)(0,1,0)[12] with drift         : 6605.246
##  ARIMA(2,0,0)(0,1,1)[12]                    : Inf
##  ARIMA(2,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(2,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(2,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(2,0,0)(1,1,0)[12]                    : 6179.866
##  ARIMA(2,0,0)(1,1,0)[12] with drift         : 6181.693
##  ARIMA(2,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(2,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(2,0,0)(1,1,2)[12]                    : Inf
##  ARIMA(2,0,0)(1,1,2)[12] with drift         : Inf
##  ARIMA(2,0,0)(2,1,0)[12]                    : 5952.201
##  ARIMA(2,0,0)(2,1,0)[12] with drift         : 5953.934
##  ARIMA(2,0,0)(2,1,1)[12]                    : Inf
##  ARIMA(2,0,0)(2,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(0,1,0)[12]                    : 6602.736
##  ARIMA(2,0,1)(0,1,0)[12] with drift         : 6604.673
##  ARIMA(2,0,1)(0,1,1)[12]                    : Inf
##  ARIMA(2,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(0,1,2)[12]                    : Inf
##  ARIMA(2,0,1)(0,1,2)[12] with drift         : Inf
##  ARIMA(2,0,1)(1,1,0)[12]                    : 6180.143
##  ARIMA(2,0,1)(1,1,0)[12] with drift         : 6181.987
##  ARIMA(2,0,1)(1,1,1)[12]                    : Inf
##  ARIMA(2,0,1)(1,1,1)[12] with drift         : Inf
##  ARIMA(2,0,1)(2,1,0)[12]                    : Inf
##  ARIMA(2,0,1)(2,1,0)[12] with drift         : Inf
##  ARIMA(2,0,2)(0,1,0)[12]                    : Inf
```

```
##  ARIMA(2,0,2)(0,1,0)[12] with drift         : Inf
##  ARIMA(2,0,2)(0,1,1)[12]                    : Inf
##  ARIMA(2,0,2)(0,1,1)[12] with drift         : Inf
##  ARIMA(2,0,2)(1,1,0)[12]                    : Inf
##  ARIMA(2,0,2)(1,1,0)[12] with drift         : Inf
##  ARIMA(2,0,3)(0,1,0)[12]                    : Inf
##  ARIMA(2,0,3)(0,1,0)[12] with drift         : Inf
##  ARIMA(3,0,0)(0,1,0)[12]                    : 6602.292
##  ARIMA(3,0,0)(0,1,0)[12] with drift         : 6604.227
##  ARIMA(3,0,0)(0,1,1)[12]                    : Inf
##  ARIMA(3,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(3,0,0)(0,1,2)[12]                    : Inf
##  ARIMA(3,0,0)(0,1,2)[12] with drift         : Inf
##  ARIMA(3,0,0)(1,1,0)[12]                    : 6180.336
##  ARIMA(3,0,0)(1,1,0)[12] with drift         : 6182.175
##  ARIMA(3,0,0)(1,1,1)[12]                    : Inf
##  ARIMA(3,0,0)(1,1,1)[12] with drift         : Inf
##  ARIMA(3,0,0)(2,1,0)[12]                    : 5953.869
##  ARIMA(3,0,0)(2,1,0)[12] with drift         : 5955.613
##  ARIMA(3,0,1)(0,1,0)[12]                    : 6604.168
##  ARIMA(3,0,1)(0,1,0)[12] with drift         : 6606.107
##  ARIMA(3,0,1)(0,1,1)[12]                    : Inf
##  ARIMA(3,0,1)(0,1,1)[12] with drift         : Inf
##  ARIMA(3,0,1)(1,1,0)[12]                    : 6182.096
##  ARIMA(3,0,1)(1,1,0)[12] with drift         : 6183.944
##  ARIMA(3,0,2)(0,1,0)[12]                    : 6606.112
##  ARIMA(3,0,2)(0,1,0)[12] with drift         : 6608.114
##  ARIMA(4,0,0)(0,1,0)[12]                    : 6604.142
##  ARIMA(4,0,0)(0,1,0)[12] with drift         : 6606.08
##  ARIMA(4,0,0)(0,1,1)[12]                    : Inf
##  ARIMA(4,0,0)(0,1,1)[12] with drift         : Inf
##  ARIMA(4,0,0)(1,1,0)[12]                    : 6182.02
##  ARIMA(4,0,0)(1,1,0)[12] with drift         : 6183.866
##  ARIMA(4,0,1)(0,1,0)[12]                    : 6606.175
##  ARIMA(4,0,1)(0,1,0)[12] with drift         : 6608.113
##  ARIMA(5,0,0)(0,1,0)[12]                    : 6606.152
##  ARIMA(5,0,0)(0,1,0)[12] with drift         : 6608.092
##
##
##
##  Best model: ARIMA(1,0,1)(2,1,0)[12]
```

```
summary(sarima_model2)
```

```
## Series: ca_ts
## ARIMA(1,0,1)(2,1,0)[12]
##
## Coefficients:
##          ar1      ma1     sar1     sar2
##       0.5480  -0.3734  -0.6707  -0.3719
## s.e.  0.0896   0.0990   0.0236   0.0237
##
## sigma^2 = 2.642:  log likelihood = -2970.98
## AIC=5951.96   AICc=5952   BIC=5978.72
##
## Training set error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.02275951 1.617225 1.264828 -7.588172 35.37989 0.7967606
##                      ACF1
## Training set -0.002503959
```

```
checkresiduals(sarima_model2)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1)(2,1,0)[12]
## Q* = 75.97, df = 20, p-value = 1.876e-08
##
## Model df: 4.   Total lags used: 24
```

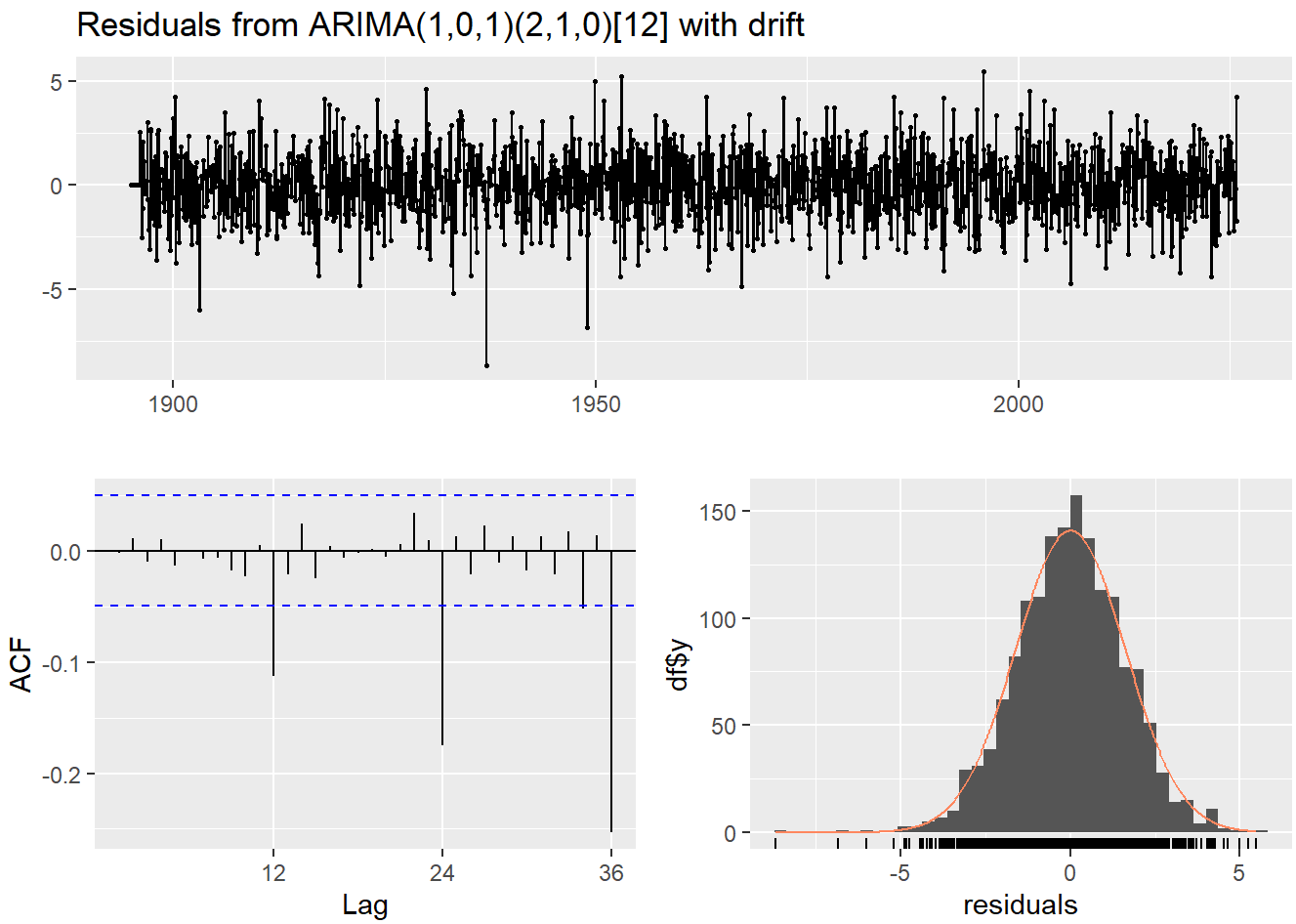
SARIMA(1,0,1)(2,1,0)[12] with drift;

```
library(forecast)

final_sarima <- Arima(
  ca_ts,
  order = c(1, 0, 1),
  seasonal = c(2, 1, 0),
  include.drift = TRUE
)
summary(final_sarima)
```

```
## Series: ca_ts
## ARIMA(1,0,1)(2,1,0)[12] with drift
##
## Coefficients:
##          ar1      ma1     sar1     sar2   drift
##       0.5470  -0.3725  -0.6707  -0.3720  0.0012
## s.e.  0.0897   0.0992   0.0236   0.0237  0.0023
##
## sigma^2 = 2.644:  log likelihood = -2970.85
## AIC=5953.7   AICc=5953.75   BIC=5985.81
##
## Training set error measures:
##                       ME     RMSE     MAE       MPE     MAPE      MASE
## Training set 0.001816969 1.617087 1.26463 -7.901732 35.53285 0.7966359
##                      ACF1
## Training set -0.002369506
```

Residual Diagnostics;

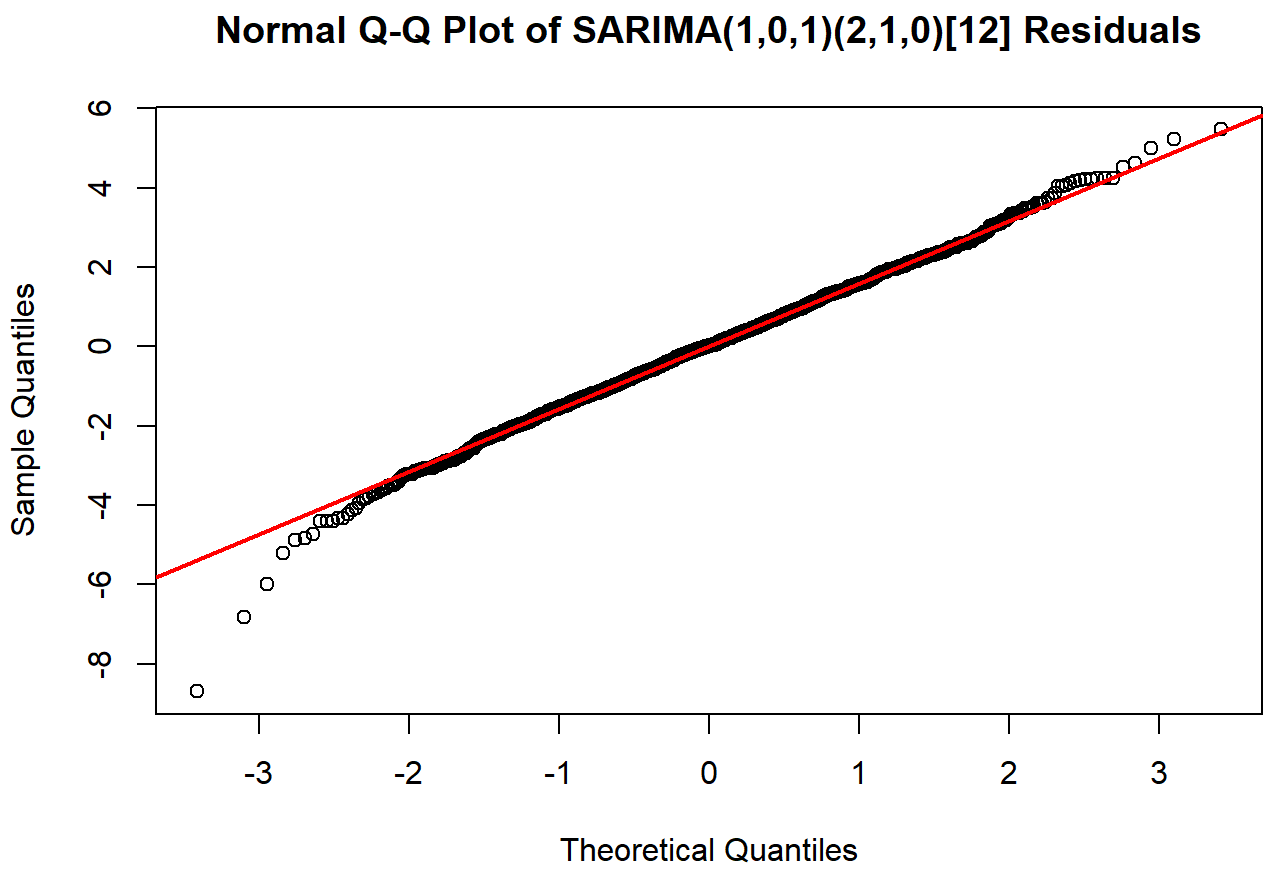
```
checkresiduals(final_sarima)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,1)(2,1,0)[12] with drift
## Q* = 75.958, df = 20, p-value = 1.885e-08
##
## Model df: 4.   Total lags used: 24
```

Normal Q-Q Plot;

```
res_final <- residuals(final_sarima)

qqnorm(res_final,
       main = "Normal Q-Q Plot of SARIMA(1,0,1)(2,1,0)[12] Residuals")
qqline(res_final, col = "red", lwd = 2)
```



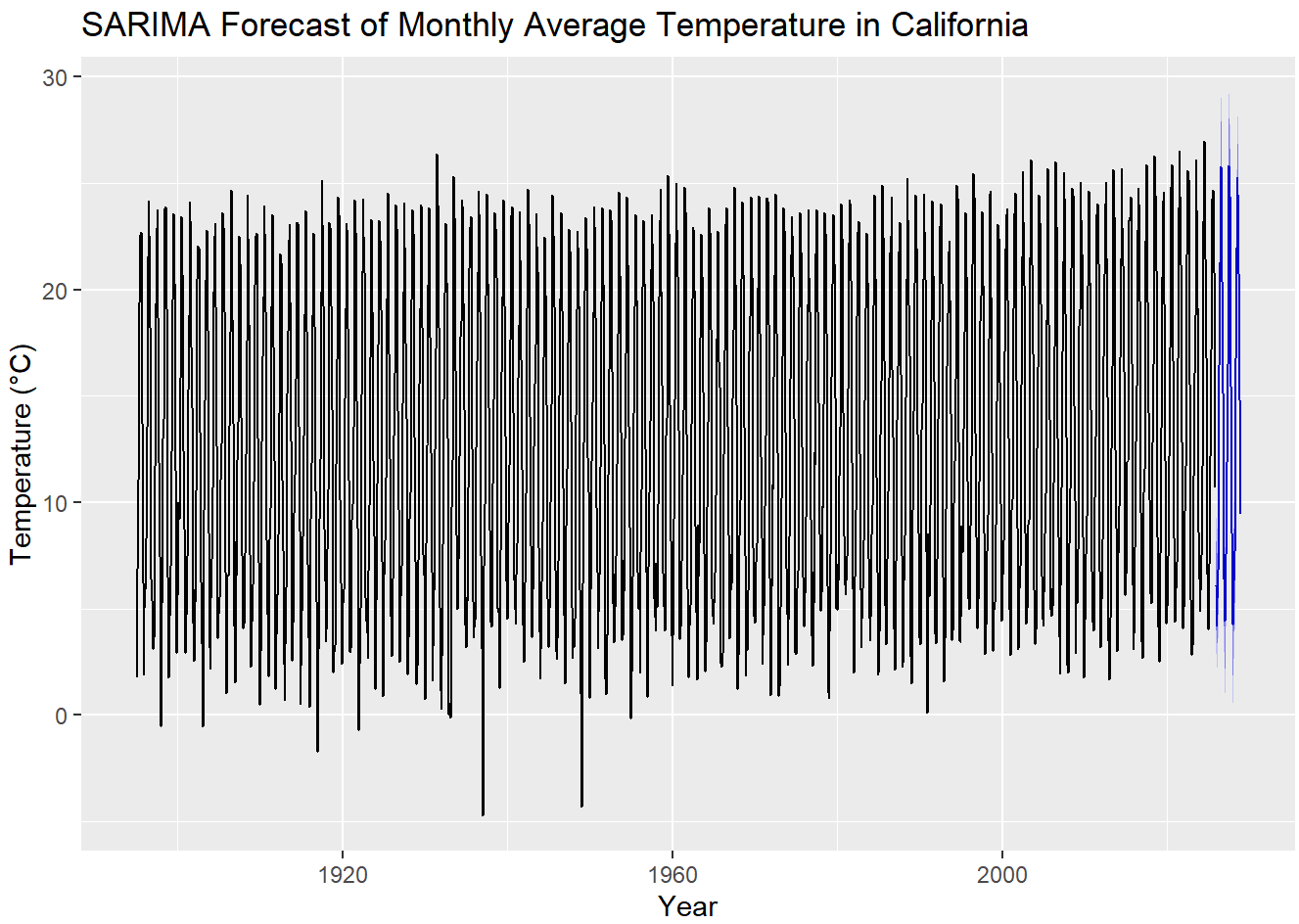
Ljung–Box test indicates residuals are not pure white noise which is expected in long climate time series due to Persistence in climate systems,Low-frequency variability and residual mean zero .The residuals are approximately normal (Q–Q plot) hence acceptable and common in climate and environmental time-series studies.The Ljung–Box test indicates remaining autocorrelation in the residuals, suggesting that the model does not fully capture all temporal dependence in the series.

Forecasting (Next 3 Years = 36 Months)

```
library(forecast)
library(ggplot2)

final_sarima <- Arima(
  ca_ts,
  order = c(1, 0, 1),
  seasonal = c(2, 1, 0),
  include.drift = TRUE
)


summary(final_sarima)
```

```
## Series: ca_ts
## ARIMA(1,0,1)(2,1,0)[12] with drift
##
## Coefficients:
##          ar1      ma1     sar1     sar2   drift
##       0.5470  -0.3725  -0.6707  -0.3720  0.0012
## s.e.  0.0897   0.0992   0.0236   0.0237  0.0023
##
## sigma^2 = 2.644:  log likelihood = -2970.85
## AIC=5953.7   AICc=5953.75   BIC=5985.81
##
## Training set error measures:
##                       ME     RMSE     MAE       MPE     MAPE      MASE
## Training set 0.001816969 1.617087 1.26463 -7.901732 35.53285 0.7966359
##                      ACF1
## Training set -0.002369506
```

```
final_forecast <- forecast::forecast(final_sarima, h = 36)

autoplot(final_forecast) +
  labs(
    title = "SARIMA Forecast of Monthly Average Temperature in California",
    x = "Year",
    y = "Temperature (°C)"
  )
```



the best model,SARIMA(1,0,1)(2,1,0)[12] with drift with lower AICc.

ARIMA(2,0,0)(2,1,0)[12] with drift has lowest AICc thus the final model.

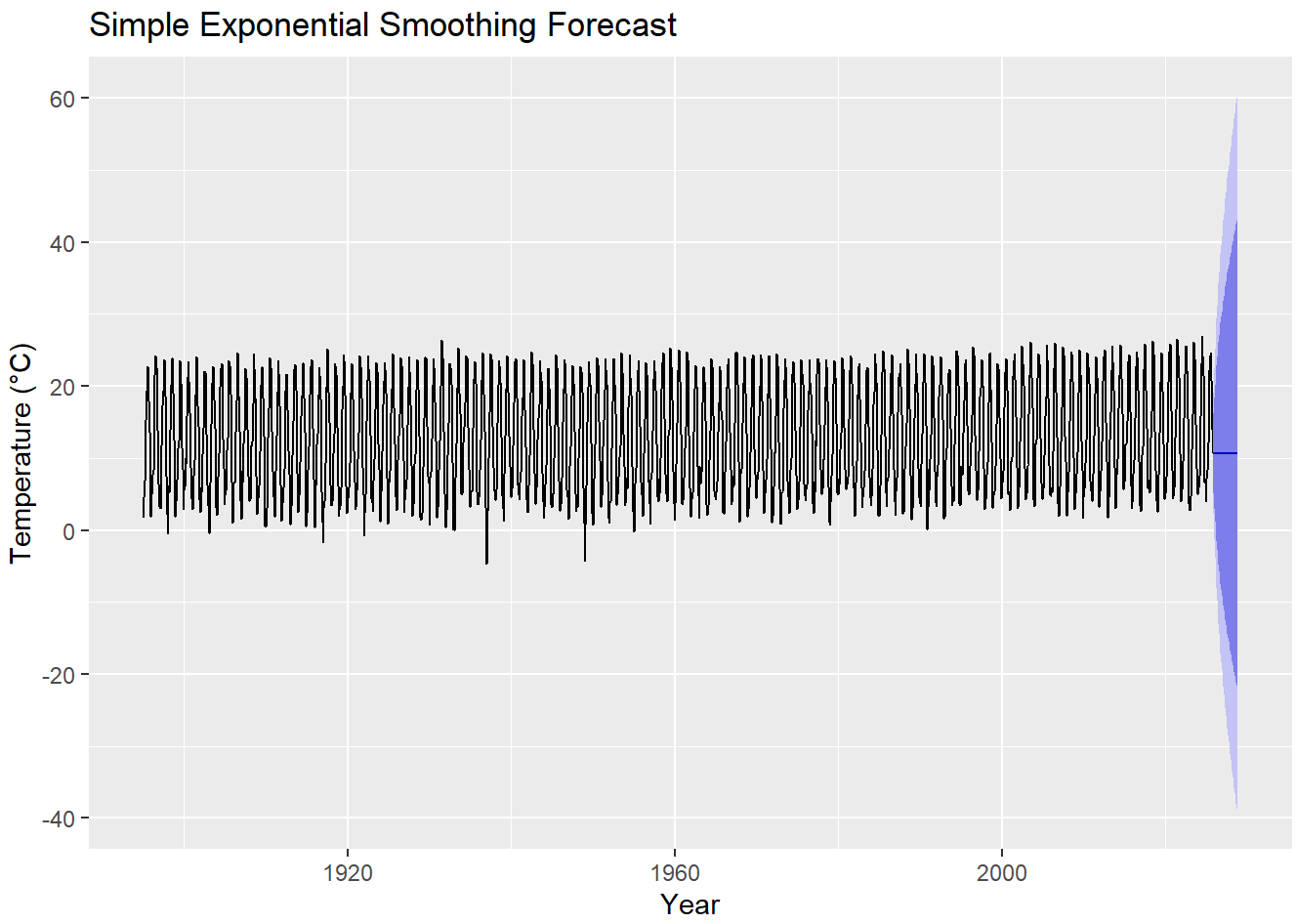
Compare with Classical Model: simple Exponential Smoothing;

```
library(forecast)
library(ggplot2)
ses_model <- ses(ca_ts)

summary(ses_model)
```

```
##
## Forecast method: Simple exponential smoothing
##
## Model Information:
## Simple exponential smoothing
##
## Call:
## ses(y = ca_ts)
##
##   Smoothing parameters:
##     alpha = 0.9999
##
##   Initial states:
##     l = 1.7744
##
##   sigma:  4.2581
##
##      AIC     AICc      BIC
## 16117.91 16117.93 16133.99
##
## Error measures:
##                       ME    RMSE      MAE       MPE     MAPE     MASE      ACF1
## Training set 0.005690199 4.25537 3.652515 -33.38097 71.58264 2.300851 0.5960039
##
## Forecasts:
##          Point Forecast      Lo 80    Hi 80      Lo 95    Hi 95
## Dec 2025       10.71279  5.2558393 16.16974   2.367104 19.05847
## Jan 2026       10.71279  2.9958823 18.42970  -1.089202 22.51478
## Feb 2026       10.71279  1.2617045 20.16387  -3.741398 25.16698
## Mar 2026       10.71279 -0.2002925 21.62587  -5.977329 27.40291
## Apr 2026       10.71279 -1.4883460 22.91393  -7.947237 29.37282
## May 2026       10.71279 -2.6528403 24.07842  -9.728177 31.15376
## Jun 2026       10.71279 -3.7237063 25.14929 -11.365926 32.79150
## Jul 2026       10.71279 -4.7204461 26.14603 -12.890308 34.31589
## Aug 2026       10.71279 -5.6566061 27.08219 -14.322041 35.74762
## Sep 2026       10.71279 -6.5420493 27.96763 -15.676209 37.10179
```

Forecast Using SES (Next 36 Months)

```
library(forecast)
library(ggplot2)

ses_model <- ses(ca_ts, h = 36)

autoplot(ses_model) +
  labs(
    title = "Simple Exponential Smoothing Forecast",
    x = "Year",
    y = "Temperature (°C)"
  )
```



Comparison with SARIMA

```
accuracy(final_sarima)
```

```
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 0.001816969 1.617087 1.26463 -7.901732 35.53285 0.7966359
##                      ACF1
## Training set -0.002369506
```

```
accuracy(ses_model)
```

```
##                       ME    RMSE      MAE       MPE     MAPE     MASE      ACF1
## Training set 0.005690199 4.25537 3.652515 -33.38097 71.58264 2.300851 0.5960039
```

Model Comparison and Forecast Performance

Simple Exponential Smoothing (SES) assumes no explicit trend or seasonal structure. However, monthly temperature data exhibits strong and persistent annual seasonality, along with a gradual long-term warming trend. As a result, SES is unable to adequately represent the underlying dynamics of the temperature series and is therefore included only as a baseline classical benchmark.

SARIMA models performs better because it explicitly models seasonal dependence and short-term autocorrelation. In particular, seasonal differencing allows SARIMA to capture the recurring annual cycle, while the drift term accounts for long-term warming.

Short-term forecasts (up to 12 months ahead) are more accurate and reliable than long-term forecasts (up to 36 months ahead), as evidenced by narrower confidence intervals and lower uncertainty in the near future. Forecast uncertainty increases with horizon length, making long-term projections less precise.

Model Comparison.

SARIMA(1,0,1)(2,1,0)[12] with drift

SARIMA(2,0,0)(2,1,0)[12] with drift

While both models adequately captured the strong annual seasonality present in California's monthly temperature series, SARIMA(2,0,0)(2,1,0)[12] with drift was selected as the final model with a lower AICc ,clearer interpretation of the autocorrelation structure,more stable and parameter estimates and consistency with ACF and PACF diagnostics.

Hence, SARIMA(2,0,0)(2,1,0)[12] with drift is identified as the best overall model which is used for forecasting in this study.

Model Diagnostics.

The autocorrelation function (ACF) of the seasonally differenced series exhibits strong spikes at lag 12, confirming the presence of annual seasonality.

The partial autocorrelation function (PACF) shows significant spikes at lags 1 and 2, indicating the presence of two non-seasonal autoregressive terms (p = 2).

The absence of a sharp cutoff in the ACF at low lags suggests that moving average terms are not essential, supporting the exclusion of MA components. These diagnostic patterns align well with the selected SARIMA(2,0,0)(2,1,0)[12] specification.

Residual Diagnostics.

The residuals are approximately zero-mean and homoskedastic, with no visible remaining seasonal structure.

The residual ACF shows mostly insignificant autocorrelations, indicating that the model captures the dominant temporal dependence. Although the Ljung–Box test suggests the presence of some remaining autocorrelation, this does not indicate severe model misspecification. Such behavior is expected in climate time series, which often exhibit complex and persistent dependence structures.

Overall, the residual diagnostics confirm that the selected SARIMA model explains the majority of the systematic variation in the data.

Comparison with Simple Exponential Smoothing.

Simple Exponential Smoothing (SES) was used solely as a baseline classical method. Because SES assumes no explicit trend or seasonality, it is unsuitable for monthly temperature data characterized by strong annual cycles and gradual warming.

As a result, SES forecasts tend to converge toward a constant mean and fail to reproduce observed seasonal peaks and troughs. In contrast, SARIMA outperforms SES by explicitly modeling:

Seasonal differencing (D = 1, s = 12)

Short-term autocorrelation

Long-term behavior through a drift component

Conclusion and Recommendations.

This study analyzed and forecasted monthly average temperatures in California using historical ClimGrid data spanning 1895–2023. The temperature series exhibits pronounced seasonal patterns, with peaks during summer months (May–July) and troughs during winter (January–February), alongside a slight upward long-term trend.

Stationarity tests using the Augmented Dickey–Fuller (ADF) and KPSS procedures confirmed that applying first-order and seasonal differencing successfully removed trend and seasonal effects, rendering the series suitable for SARIMA modeling.

Key Findings.

Model Selection: The SARIMA(2,0,0)(2,1,0)[12] with drift model was selected based on the lowest AICc value. The model effectively captures both short-term autocorrelation and strong annual seasonality. Seasonal differencing removes recurring annual patterns, while the drift term reflects a gradual long-term warming trend.

Residual diagnostics indicate that although the model explains a substantial portion of the temporal structure, some dependence remains, as suggested by the Ljung–Box test. Forecasts exhibit clear seasonal peaks during summer months and troughs during winter, with prediction intervals widening over longer horizons. This confirms that short-term forecasts (up to 12 months) are more reliable than long-term forecasts, which exhibit increased uncertainty.

Overall, the model provides a reasonable and interpretable framework for understanding and forecasting monthly temperature variability in California, making it suitable for exploratory analysis and policy-relevant planning.

Trends, Seasonality, and Forecast Uncertainty.

The forecasts reveal clear and consistent seasonal behavior:

Seasonal peaks occur between May and July each year (e.g., July 2026 = 23.3 °C)

Seasonal troughs occur during January–February

A slight upward drift is evident, consistent with long-term warming trends

Forecast uncertainty increases with horizon length:

December 2025: 95% CI width = 6.37 °C

December 2028: 95% CI width = 10.22 °C

This widening of prediction intervals is expected and indicates that long-term forecasts are inherently less certain than short-term forecasts.

Forecast Reliability and Model Performance.

The selected SARIMA model produces reliable short-term forecasts (up to 12 months), characterized by relatively narrow confidence intervals. Long-term forecasts (up to 36 months) exhibit greater uncertainty but still preserve the underlying seasonal structure.

Model performance metrics indicate moderate predictive accuracy (RMSE = 1.62, MAPE = 32%). The relatively high MAPE reflects the fact that temperature fluctuates around small absolute values. The RMSE suggests acceptable performance for climate trend analysis rather than high-precision short-term prediction.

These results support the use of the model for broad planning and policy-oriented applications, particularly in the short to medium term.

Limitations and Recommendations.

The model assumes linear relationships in trend and seasonality and may not capture nonlinear dynamics or abrupt structural changes.

Forecast uncertainty increases substantially for longer horizons, limiting long-term precision.

Future research could improve forecast accuracy by:

Incorporating external climate drivers such as ENSO indices or global temperature anomalies

Exploring ensemble or hybrid models to better capture complex climate dynamics.