Ice Caps

Josiah Chung

2023-04-28

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.3     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.4     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.0
## ✔ purrr     1.0.2
## ── 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 conflic
ts to become errors
```

```
library(dplyr)
library(lubridate)
library(zoo)
```

```
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(ggplot2)
library(tseries)
```

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

```
library(forecast)
library(fpp)
```

```
## Loading required package: fma
## Loading required package: expsmooth
## Loading required package: lmtest
```

```
library(vars)
```

```
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
## The following objects are masked from 'package:fma':
##
##     cement, housing, petrol
##
## The following object is masked from 'package:dplyr':
##
##     select
##
## Loading required package: strucchange
## Loading required package: sandwich
##
## Attaching package: 'strucchange'
##
## The following object is masked from 'package:stringr':
##
##     boundary
##
## Loading required package: urca
```

```
library(TSA)
```

```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast
##
## Attaching package: 'TSA'
##
## The following object is masked from 'package:readr':
##
##     spec
##
## The following objects are masked from 'package:stats':
##
##     acf, arima
##
## The following object is masked from 'package:utils':
##
##     tar
```

Combining and Preparing Data: North

```
icecaps_N <- data.frame()

# Loop through the 12 CSV files
for (i in 1:12) {
  if (i < 10){
    df <- read.csv(paste0("N_0", i, "_extent_v3.0.csv"))
  }
  if (i >= 10){
    df <- read.csv(paste0("N_", i, "_extent_v3.0.csv"))
  }
  icecaps_N <- rbind(icecaps_N, df)
}

# Combine the year and month columns into a new "date" column and convert to a year-m
onth format
icecaps_N <- icecaps_N %>%
  mutate(date = ym(paste(year, str_pad(mo, 2, pad = "0"), sep = "-"))) %>%
  mutate(date = as.yearmon(date)) %>%
  dplyr::select(date, extent, area) %>% # Select only the date and extent columns
  arrange(date) %>% # Sort by the date column
  mutate(extent = replace(extent, extent == -9999.00, NA), # Replace -9999.00 with NA
in the extent column
         area = replace(area, area == -9999.00, NA)) # Replace -9999.00 with NA in th
e area column

head(icecaps_N)
```

```
##       date extent  area
## 1 Nov 1978  11.65  9.04
## 2 Dec 1978  13.67 10.90
## 3 Jan 1979  15.41 12.41
## 4 Feb 1979  16.18 13.18
## 5 Mar 1979  16.34 13.21
## 6 Apr 1979  15.45 12.53
```

Combining and Preparing Data: South

```
icecaps_S <- data.frame()

# Loop through the 12 CSV files
for (i in 1:12) {
  if (i < 10){
    df <- read.csv(paste0("S_0", i, "_extent_v3.0.csv"))
  }
  if (i >= 10){
    df <- read.csv(paste0("S_", i, "_extent_v3.0.csv"))
  }
  icecaps_S <- rbind(icecaps_S, df)
}

# Combine the year and month columns into a new "date" column and convert to a year-m
onth format
icecaps_S <- icecaps_S %>%
  mutate(date = ym(paste(year, str_pad(mo, 2, pad = "0"), sep = "-"))) %>%
  mutate(date = as.yearmon(date)) %>%
  dplyr::select(date, extent, area) %>% # Select only the date and extent columns
  arrange(date) %>% # Sort by the date column
  mutate(extent = replace(extent, extent == -9999.00, NA), # Replace -9999.00 with NA
in the extent column
         area = replace(area, area == -9999.00, NA)) # Replace -9999.00 with NA in th
e area column


head(icecaps_S)
```

```
##       date extent  area
## 1 Nov 1978  15.90 11.69
## 2 Dec 1978  10.40  6.97
## 3 Jan 1979   5.40  3.47
## 4 Feb 1979   3.14  2.11
## 5 Mar 1979   4.00  2.66
## 6 Apr 1979   7.49  5.45
```

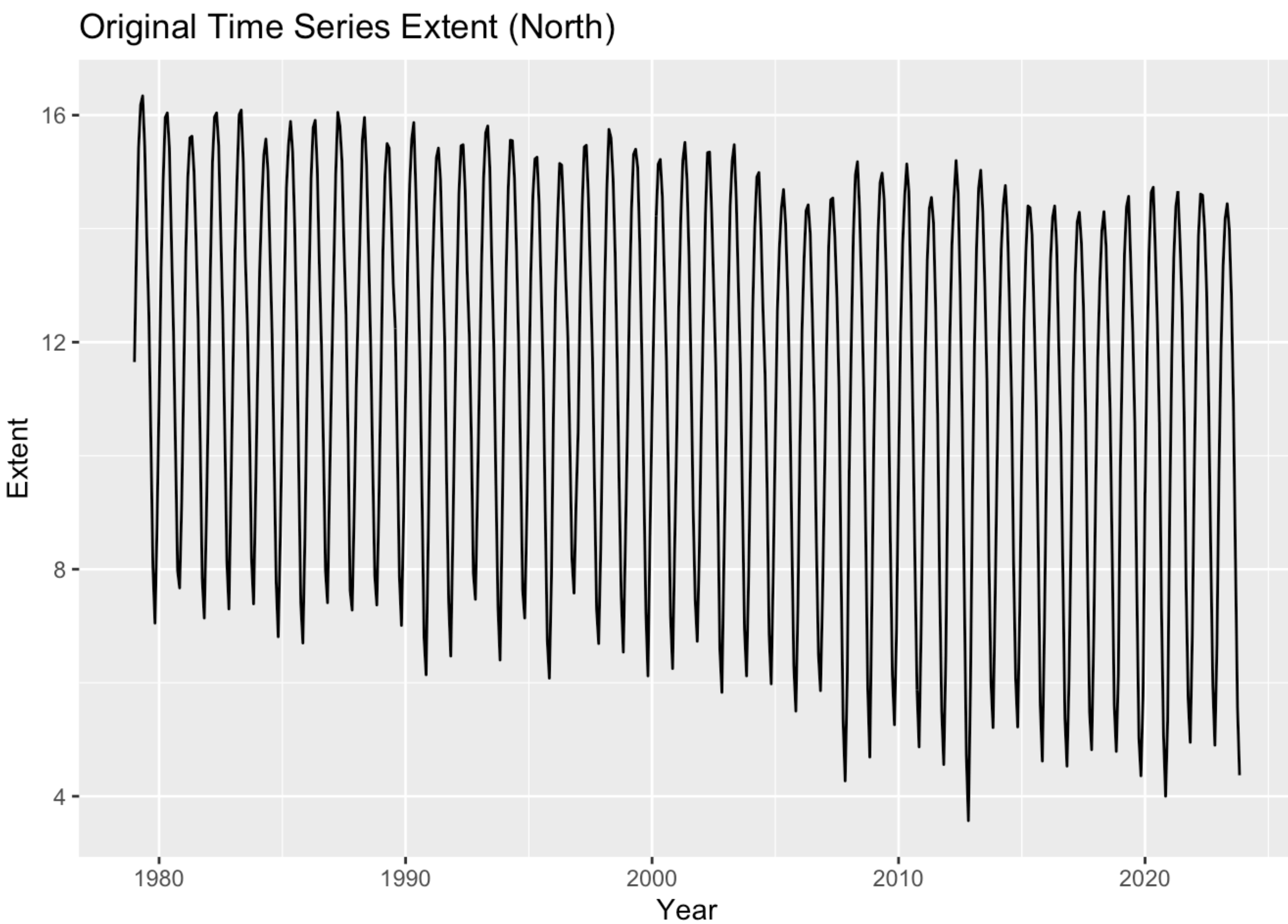
Handling Missing Data for North and South

```
# Function that imputates missing data by replacing missing values with the average o
f all values of the same month

fill_missing <- function(df) {
  for (i in 1:nrow(df)){
    if (is.na(df[i, "extent"])){ # if there is a missing value in extent column
      my_sum <- 0
      month_count <- 0
      df_month <- month(df[i,"date"]) # store the month of missing value
      for (j in 1:nrow(df)){
          if (i != j){
            # take the average of extents of all same months
            if (month(df[j,"date"]) == df_month){
              my_sum <- my_sum + (df[j,"extent"])
              month_count <- month_count + 1
            }
          }
      }
      # replace missing value with calculated average
      df[i, "extent"] <- (my_sum/month_count)
    }

    if (is.na(df[i, "area"])){ # if there is a missing value in area column
      my_sum <- 0
      month_count <- 0
      df_month <- month(df[i,"date"]) # store the month of missing value
      for (j in 1:nrow(df)){
          if (i != j){
            # take the average of areas of all same months
            if (month(df[j,"date"]) == df_month){
              my_sum <- my_sum + (df[j,"area"])
              month_count <- month_count + 1
            }
          }
      }
      # replace missing value with calculated average
      df[i, "area"] <- (my_sum/month_count)
    }
  }
  df
  }
```

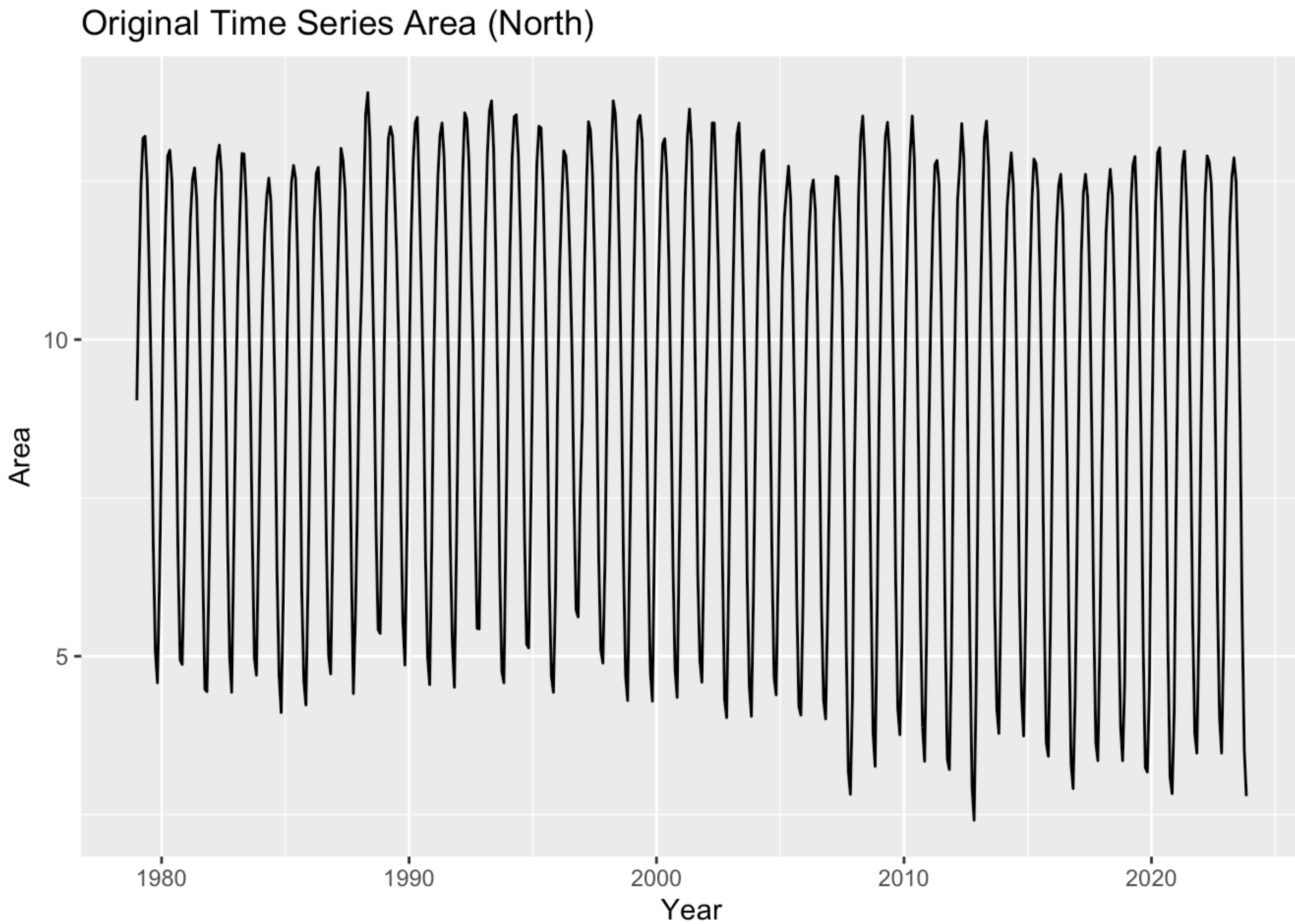
```
icecaps_N <- fill_missing(icecaps_N)
icecaps_S <- fill_missing(icecaps_S)
```

Plotting the Data

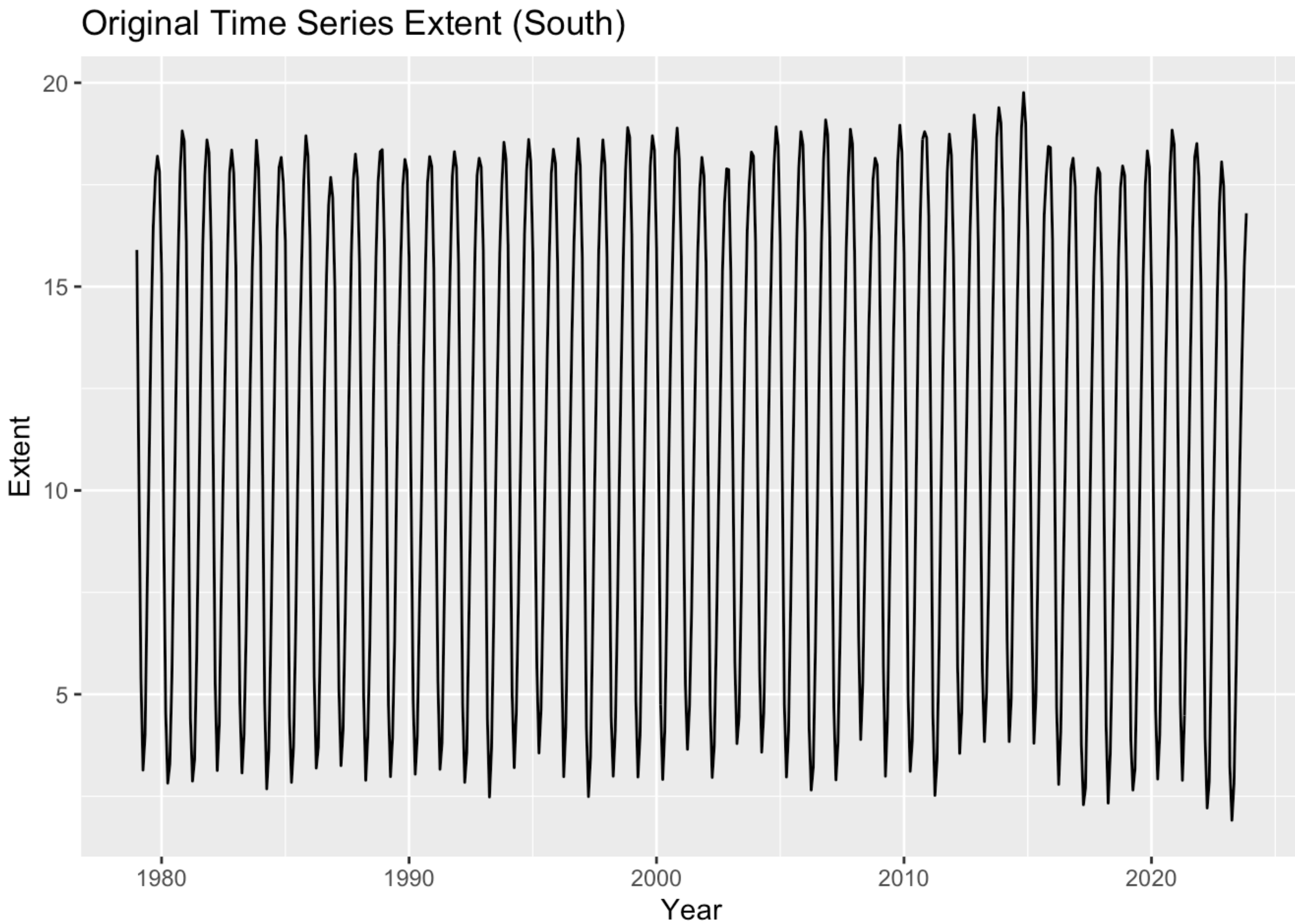
```
ts_extent_N <- ts(icecaps_N$extent, start=c(1979,1), frequency=12)
ts_area_N <- ts(icecaps_N$area, start=c(1979,1), frequency=12)


autoplot(ts_extent_N, xlab = "Year", ylab = "Extent", main = "Original Time Series Ex
tent (North)")
```



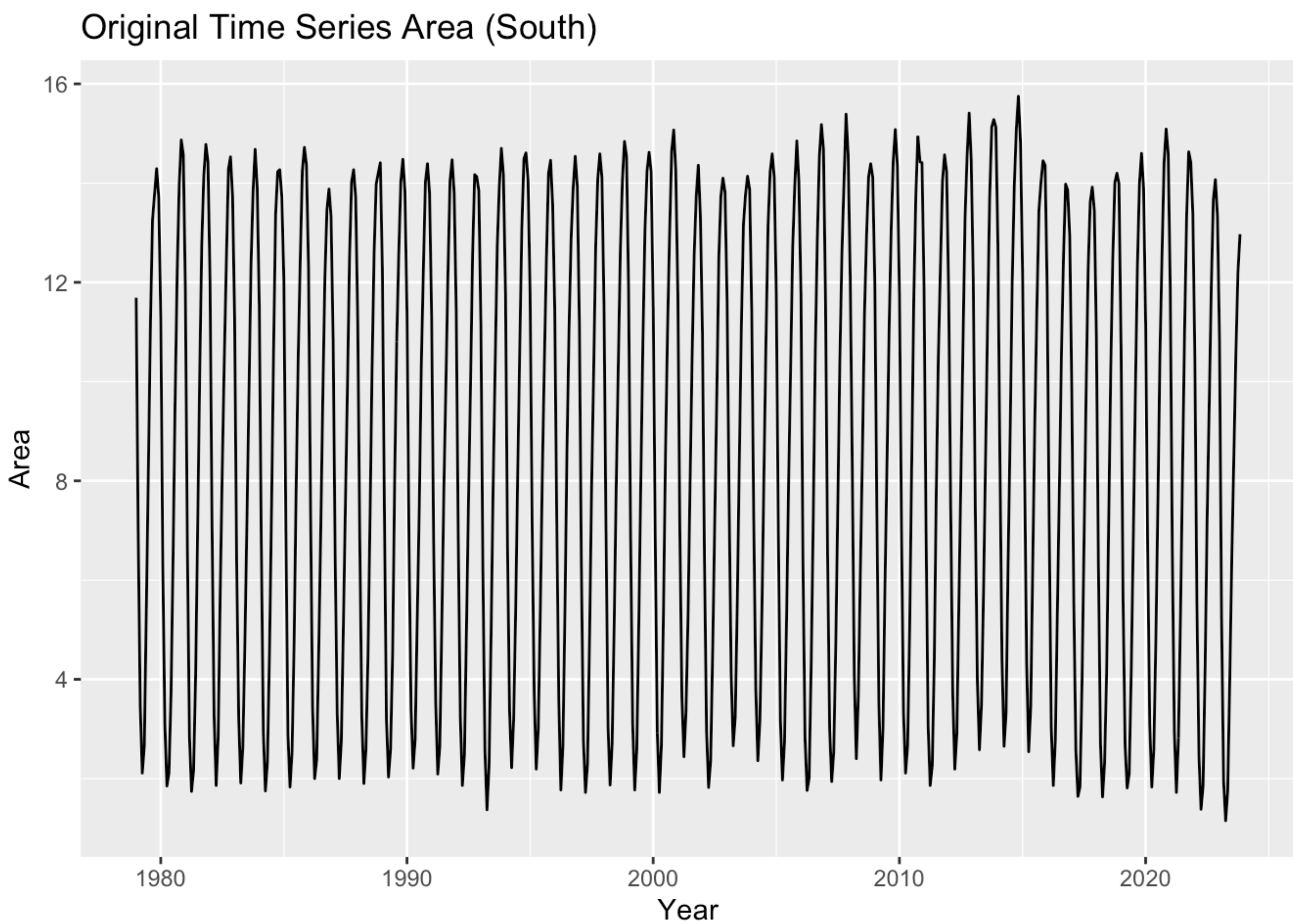

```
autoplot(ts_area_N, xlab = "Year", ylab = "Area", main = "Original Time Series Area (
North)")
```




```
ts_extent_S <- ts(icecaps_S$extent, start=c(1979,1), frequency=12)
ts_area_S <- ts(icecaps_S$area, start=c(1979,1), frequency=12)

autoplot(ts_extent_S, xlab = "Year", ylab = "Extent", main = "Original Time Series Ex
tent (South)")
```




```
autoplot(ts_area_S, xlab = "Year", ylab = "Area", main = "Original Time Series Area (
South)")
```

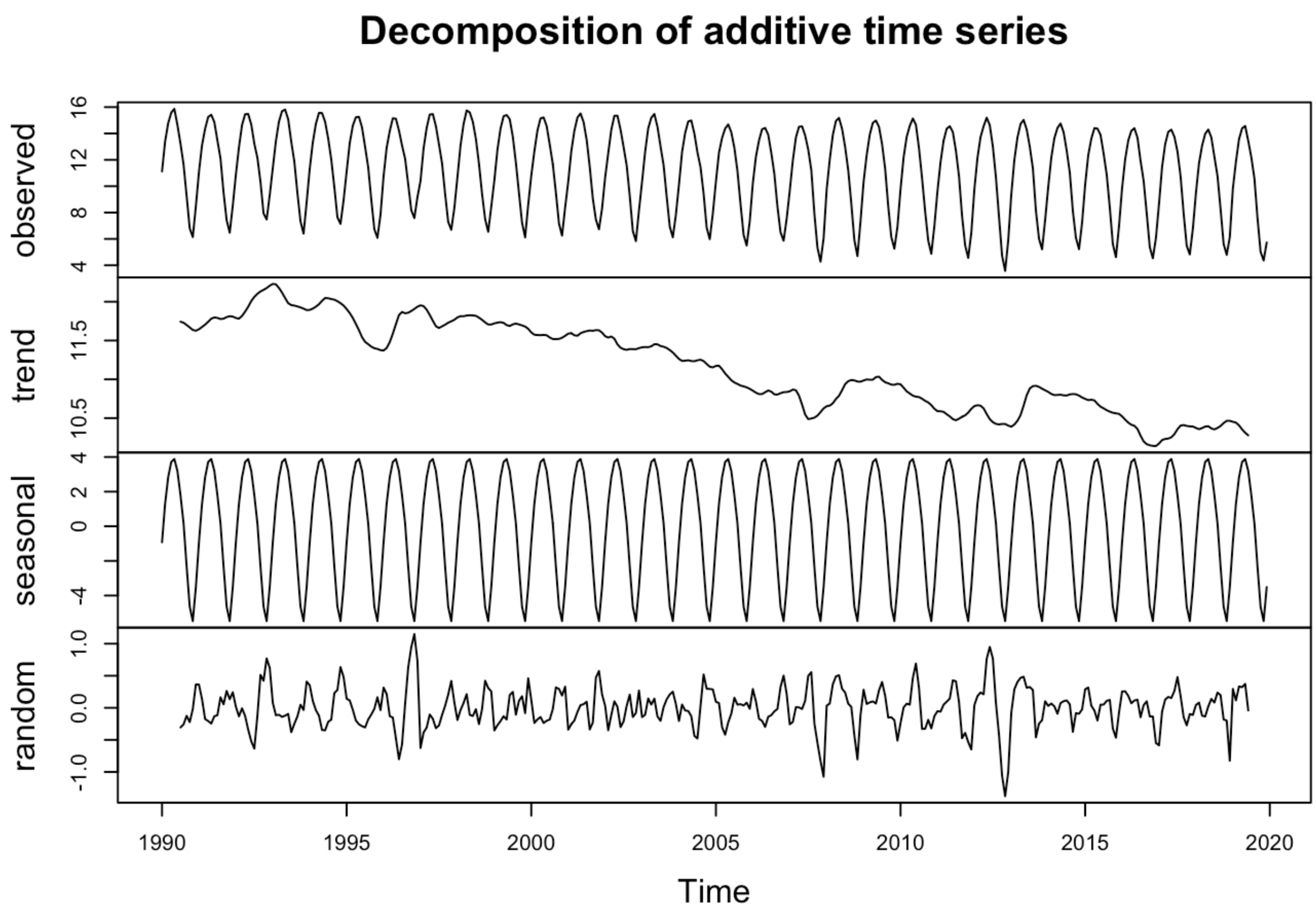


Getting train and test data

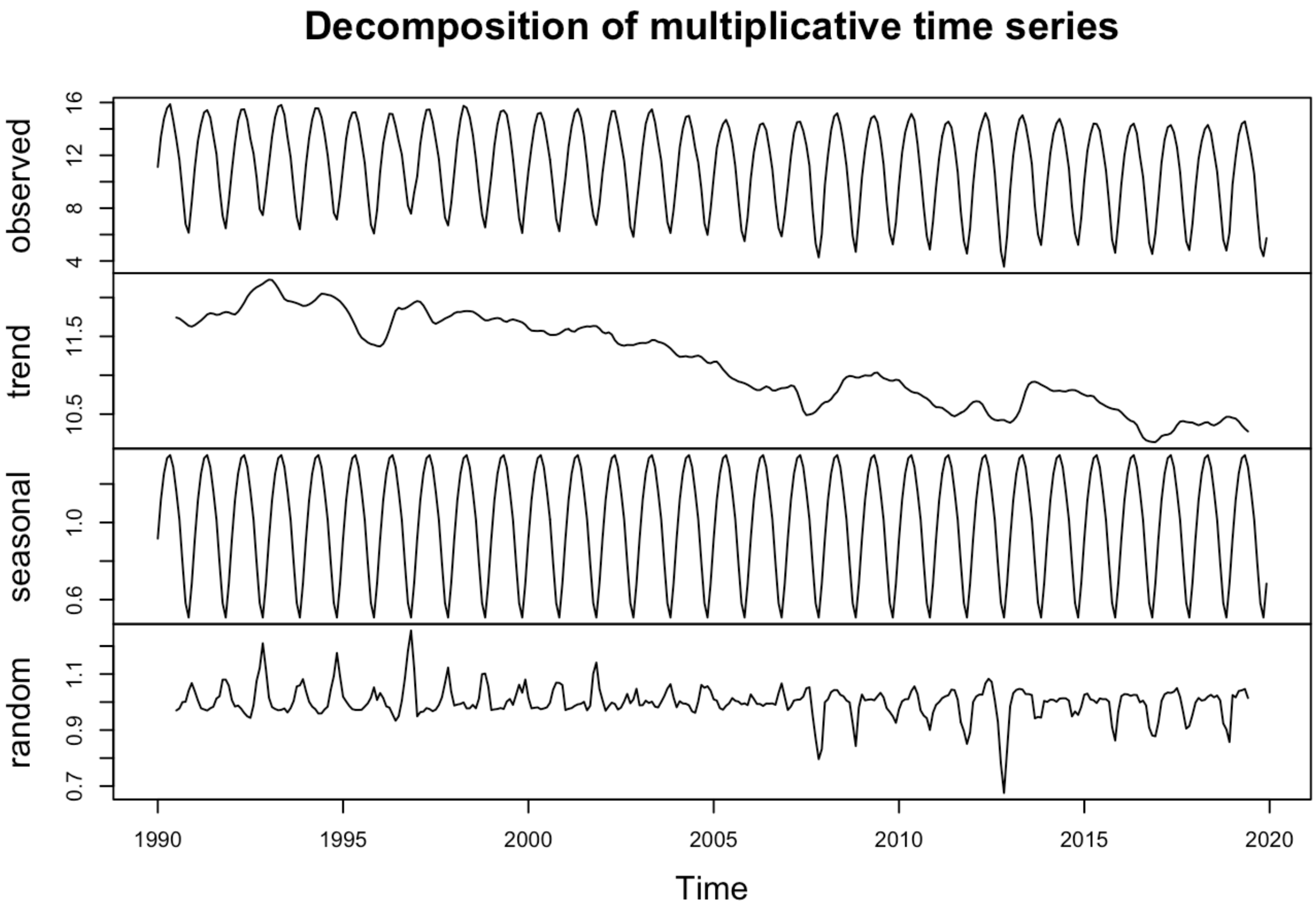
```
train_extent_N <- window(ts_extent_N, start = c(1990, 1), end = c(2019, 12))
test_extent_N <- window(ts_extent_N, start = c(2020, 1))
```

```
train_area_N <- window(ts_area_N, start = c(1990, 1), end = c(2019, 12))
test_area_N <- window(ts_area_N, start = c(2020, 1))
```

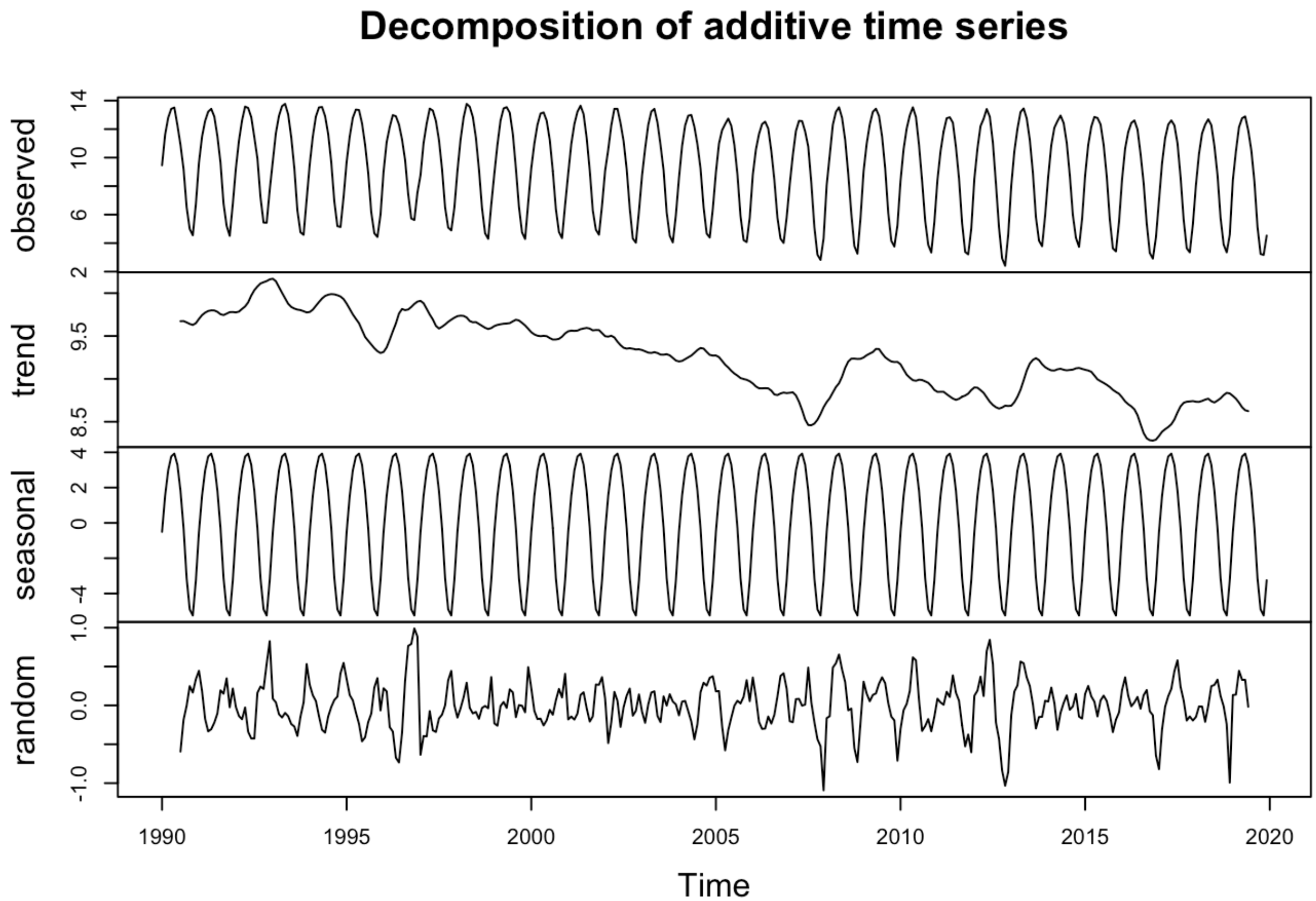
```
#exponential smoothning
fit_add <- decompose(train_extent_N, type="additive")
plot(fit_add)
```



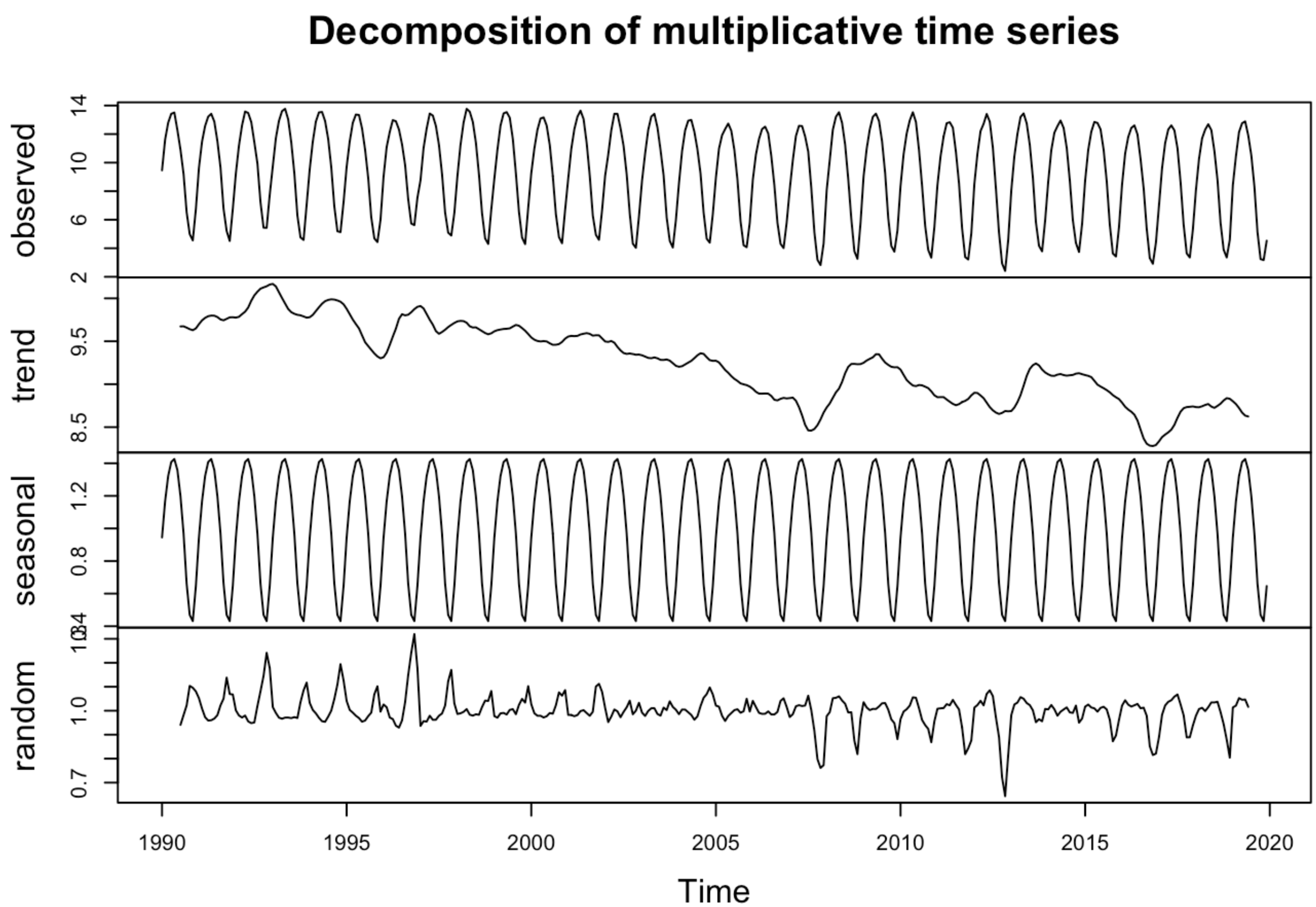

```
fit_mult <- decompose(train_extent_N, type="multiplicative")
plot(fit_mult)
```




```
fit_add <- decompose(train_area_N, type="additive")
plot(fit_add)
```




```
fit_mult <- decompose(train_area_N, type="multiplicative")
plot(fit_mult)
```



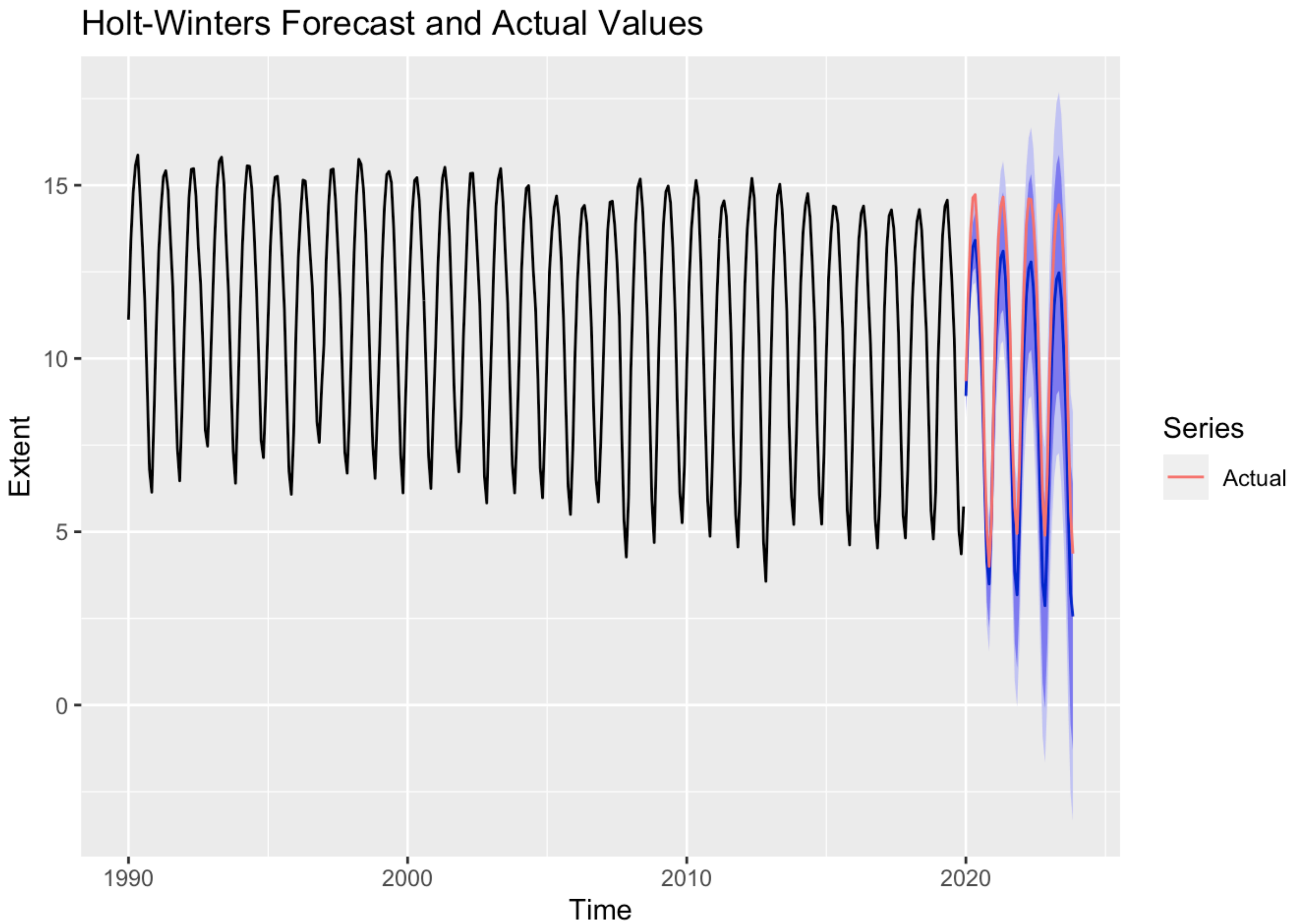

```
#holt winters additive
fit_hw_add <- hw(train_extent_N, h=length(test_extent_N), seasonal="add")
summary(fit_hw_add)
```

```
##
## Forecast method: Holt-Winters' additive method
##
## Model Information:
## Holt-Winters' additive method
##
## Call:
##  hw(y = train_extent_N, h = length(test_extent_N), seasonal = "add")
##
##   Smoothing parameters:
##     alpha = 0.8553
##     beta  = 0.0244
##     gamma = 0.1446
##
##   Initial states:
```

```
##     l = 12.003
##     b = -0.0651
##     s = -3.115 -5.2471 -4.5906 -2.5271 -0.2114 1.33
##            2.9469 3.8459 3.8495 2.9995 1.4686 -0.7493
##
##   sigma:  0.2988
##
##      AIC     AICc      BIC
## 1266.877 1268.667 1332.941
##
## Error measures:
##                       ME      RMSE       MAE        MPE     MAPE      MASE
## Training set 0.004451767 0.2920815 0.2137866 -0.3050211 2.274539 0.6206018
##                  ACF1
## Training set 0.192518
##
## Forecasts:
##          Point Forecast       Lo 80     Hi 80        Lo 95     Hi 95
## Jan 2020       8.919274  8.53635046  9.302198  8.333642910  9.504905
## Feb 2020      11.076380 10.56637746 11.586382 10.296398538 11.856361
## Mar 2020      12.482069 11.86567314 13.098464 11.539373061 13.424764
## Apr 2020      13.222261 12.51069392 13.933828 12.134013089 14.310509
## May 2020      13.411099 12.61145596 14.210742 12.188150553 14.634047
## Jun 2020      12.679263 11.79641434 13.562111 11.329062705 14.029462
## Jul 2020      11.249293 10.28675575 12.211830  9.777219286 12.721367
## Aug 2020       9.436336  8.39673332 10.475939  7.846400877 11.026271
## Sep 2020       6.582164  5.46749742  7.696831  4.877428416  8.286900
## Oct 2020       4.155141  2.96695840  5.343323  2.337972663  5.972309
## Nov 2020       3.493723  2.23323425  4.754212  1.565971631  5.421475
## Dec 2020       5.417641  4.08579155  6.749490  3.380753104  7.454529
## Jan 2021       8.606855  7.18606732 10.027642  6.433947808 10.779762
## Feb 2021      10.763961  9.27398818 12.253933  8.485244442 13.042677
## Mar 2021      12.169649 10.61085048 13.728448  9.785672163 14.553627
## Apr 2021      12.909842 11.28247555 14.537208 10.420999942 15.398683
## May 2021      13.098680 11.40292293 14.794436 10.505243523 15.692116
## Jun 2021      12.366843 10.60280277 14.130884  9.668976093 15.064711
## Jul 2021      10.936874  9.10459679 12.769151  8.134647928 13.739100
## Aug 2021       9.123917  7.22339987 11.024434  6.217326974 12.030507
## Sep 2021       6.269745  4.30094094  8.238549  3.258718987  9.280771
## Oct 2021       3.842722  1.80554497  5.879898  0.727128851  6.958314
## Nov 2021       3.181304  1.07563697  5.286971 -0.039035888  6.401644
## Dec 2021       5.105222  2.93091686  7.279526  1.779909439  8.430534
## Jan 2022       8.294436  6.03707679 10.551795  4.842103173 11.746768
## Feb 2022      10.451541  8.12560055 12.777482  6.894321872 14.008761
## Mar 2022      11.857230  9.46247032 14.251990  8.194761012 15.519699
## Apr 2022      12.597422 10.13359088 15.061254  8.829317224 16.365528
## May 2022      12.786260 10.25309087 15.319430  8.912111895 16.660409
```

```
## Jun 2022      12.054424  9.45163814 14.657210  8.073806470 16.035042
## Jul 2022      10.624455  7.95176303 13.297146  6.536925577 14.711984
## Aug 2022       8.811498  6.06860166 11.554394  4.616600273 13.006395
## Sep 2022       5.957326  3.14391826  8.770733  1.654590245 10.260061
## Oct 2022       3.530302  0.64606810  6.414536 -0.880753250  7.941358
## Nov 2022       2.868885 -0.08649759  5.824267 -1.650982577  7.388752
## Dec 2022       4.792802  1.76594389  7.819661  0.163621773  9.421983
## Jan 2023       7.982016  4.87103010 11.093003  3.224173402 12.739859
## Feb 2023      10.139122  6.95626788 13.321976  5.271366637 15.006878
## Mar 2023      11.544811  8.28973559 14.799886  6.566602765 16.523019
## Apr 2023      12.285003  8.95735028 15.612656  7.195797208 17.374209
## May 2023      12.473841  9.07325151 15.874431  7.273088046 17.674594
## Jun 2023      11.742005  8.26811684 15.215893  6.429151585 17.054858
## Jul 2023      10.312035  6.76448530 13.859585  4.886525731 15.737545
## Aug 2023       8.499078  4.87750079 12.120656  2.960353427 14.037803
## Sep 2023       5.644907  1.94893449  9.340879 -0.007594972 11.297408
## Oct 2023       3.217883 -0.55285203  6.988618 -2.548958613  8.984725
## Nov 2023       2.556466 -1.28940190  6.402333 -3.325281212  8.438213
```

```
autoplot(fit_hw_add) +
  autolayer(test_extent_N, series = "Actual") +
  xlab("Time") +
  ylab("Extent") +
  ggtitle("Holt-Winters Forecast and Actual Values") +
  guides(color = guide_legend(title = "Series"))
```




```
#Red- Actual
# blue line predicted
# blue shade confidence interval
```

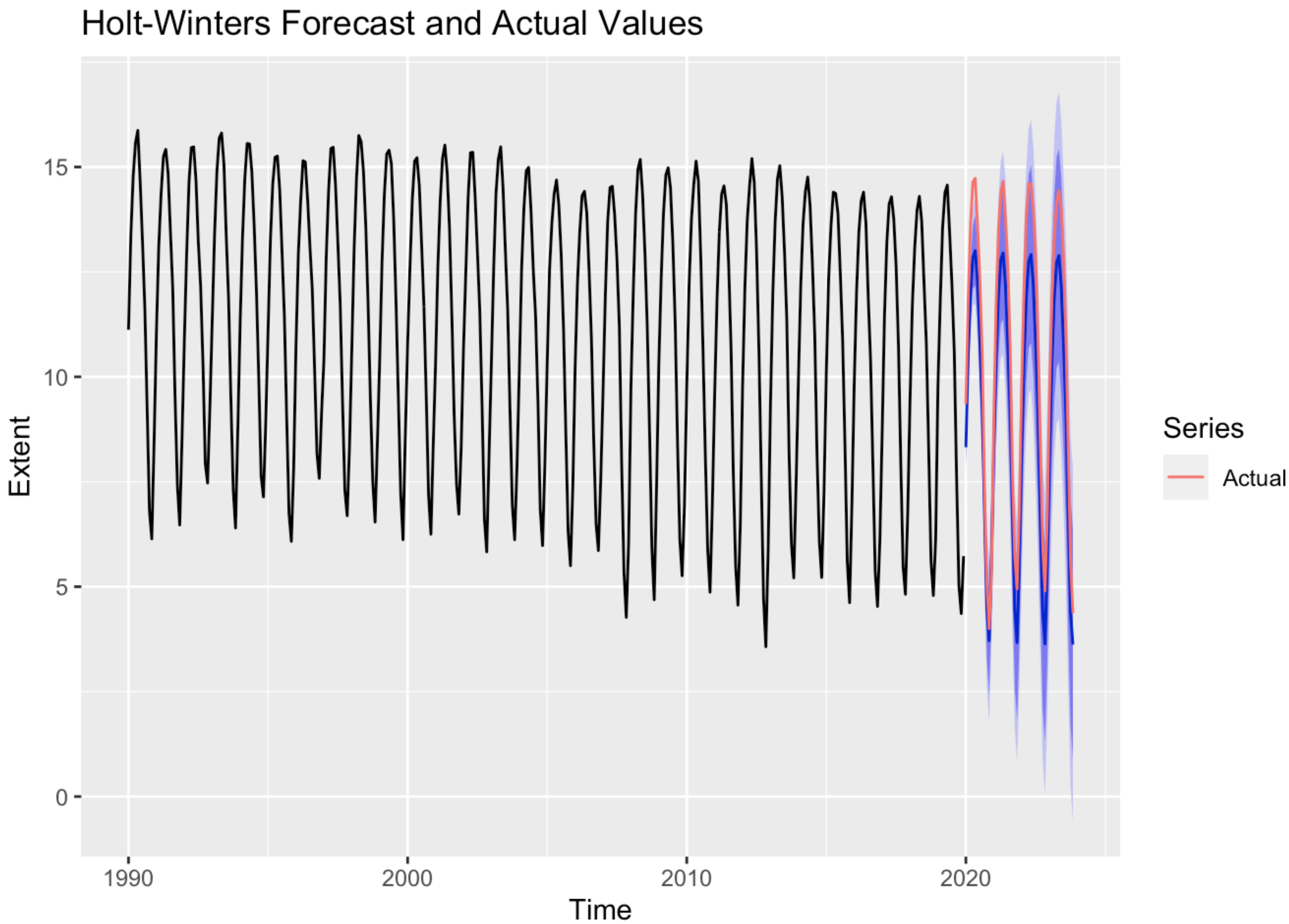
```
#holt winters additive damping
fit_hw_add_damp <- hw(train_extent_N, h=length(test_extent_N), seasonal="add",
           damped=TRUE)
summary(fit_hw_add_damp)
```

```
##
## Forecast method: Damped Holt-Winters' additive method
##
## Model Information:
## Damped Holt-Winters' additive method
##
## Call:
##  hw(y = train_extent_N, h = length(test_extent_N), seasonal = "add",
##
```

```
## Call:
##     damped = TRUE)
##
##   Smoothing parameters:
##     alpha = 0.9999
##     beta  = 0.0059
##     gamma = 1e-04
##     phi   = 0.9645
##
##   Initial states:
##     l = 11.4974
##     b = 0.0245
##     s = -3.4534 -5.408 -4.6346 -2.3524 0.1138 1.7213
##            3.1181 3.8539 3.6908 2.8529 1.3501 -0.8525
##
##   sigma:  0.2875
##
##      AIC     AICc      BIC
## 1240.114 1242.120 1310.064
##
## Error measures:
##                       ME      RMSE       MAE        MPE    MAPE      MASE
## Training set -0.00758449 0.2806433 0.2026014 -0.2644521 2.16215 0.5881322
##                   ACF1
## Training set 0.1013728
##
## Forecasts:
##          Point Forecast      Lo 80     Hi 80      Lo 95     Hi 95
## Jan 2020       8.324067  7.9556034  8.692531  7.7605505  8.887584
## Feb 2020      10.520270  9.9977183 11.042821  9.7210964 11.319443
## Mar 2020      12.016694 11.3749254 12.658463 11.0351936 12.998194
## Apr 2020      12.848605 12.1055428 13.591667 11.7121895 13.985020
## May 2020      13.005724 12.1727550 13.838693 11.7318078 14.279641
## Jun 2020      12.264377 11.3495479 13.179207 10.8652665 13.663488
## Jul 2020      10.862066  9.8714479 11.852684  9.3470464 12.377085
## Aug 2020       9.249157  8.1875401 10.310773  7.6255542 10.872759
## Sep 2020       6.777849  5.6491381  7.906561  5.0516344  8.504065
## Oct 2020       4.490675  3.2981303  5.683220  2.6668351  6.314516
## Nov 2020       3.712565  2.4589643  4.966165  1.7953482  5.629781
## Dec 2020       5.662783  4.3505314  6.975034  3.6558675  7.669698
## Jan 2021       8.259228  6.8904242  9.628033  6.1658230 10.352634
## Feb 2021      10.457732  9.0342557 11.881208  8.2807130 12.634751
## Mar 2021      11.956376 10.4799013 13.432850  9.6983030 14.214448
## Apr 2021      12.790427 11.2624650 14.318389 10.4536107 15.127244
## May 2021      12.949611 11.3715341 14.527688 10.5361506 15.363072
## Jun 2021      12.210256 10.5833194 13.837192  9.7220713 14.698440
## Jul 2021      10.809865  9.1352242 12.484506  8.2487230 13.371007
```

```
## Aug 2021       9.198808  7.4775306 10.920086  6.5663412 11.831276
## Sep 2021       6.729288  4.9623644  8.496212  4.0270115  9.431565
## Oct 2021       4.443837  2.6321916  6.255483  1.6731643  7.214510
## Nov 2021       3.667389  1.8118856  5.522892  0.8296414  6.505136
## Dec 2021       5.619210  3.7206605  7.517760  2.7156290  8.522791
## Jan 2022       8.217202  6.2763624 10.158042  5.2489438 11.185461
## Feb 2022      10.417197  8.4347964 12.399598  7.3853768 13.449018
## Mar 2022      11.917280  9.8940005 13.940559  8.8229413 15.011618
## Apr 2022      12.752718 10.6892090 14.816228  9.5968530 15.908584
## May 2022      12.913241 10.8101168 15.016365  9.6967902 16.129692
## Jun 2022      12.175176 10.0330243 14.317328  8.8990377 15.451315
## Jul 2022      10.776030  8.5954110 12.956650  7.4410608 14.111000
## Aug 2022       9.166174  6.9476230 11.384726  5.7731928 12.559156
## Sep 2022       6.697812  4.4418418  8.953783  3.2476032 10.148021
## Oct 2022       4.413479  2.1205810  6.706376  0.9067944  7.920163
## Nov 2022       3.638108  1.3087555  5.967460  0.0756710  7.200544
## Dec 2022       5.590968  3.2256159  7.956320  1.9734741  9.208462
## Jan 2023       8.189963  5.7890414 10.590884  4.5180705 11.861854
## Feb 2023      10.390924  7.9548626 12.826986  6.6652895 14.116559
## Mar 2023      11.891939  9.4211441 14.362734  8.1131844 15.670693
## Apr 2023      12.728277 10.2231425 15.233412  8.8970043 16.559550
## May 2023      12.889667 10.3505728 15.428761  9.0064575 16.772876
## Jun 2023      12.152439  9.5797534 14.725124  8.2178559 16.087022
## Jul 2023      10.754100  8.1481799 13.360020  6.7686891 14.739511
## Aug 2023       9.145022  6.5062133 11.783831  5.1093121 13.180733
## Sep 2023       6.677411  4.0060484  9.348773  2.5919145 10.762907
## Oct 2023       4.393801  1.6902110  7.097392  0.2590167  8.528586
## Nov 2023       3.619129  0.8836271  6.354630 -0.5644600  7.802717
```

```
autoplot(fit_hw_add_damp) +
  autolayer(test_extent_N, series = "Actual") +
  xlab("Time") +
  ylab("Extent") +
  ggtitle("Holt-Winters Forecast and Actual Values") +
  guides(color = guide_legend(title = "Series"))
```



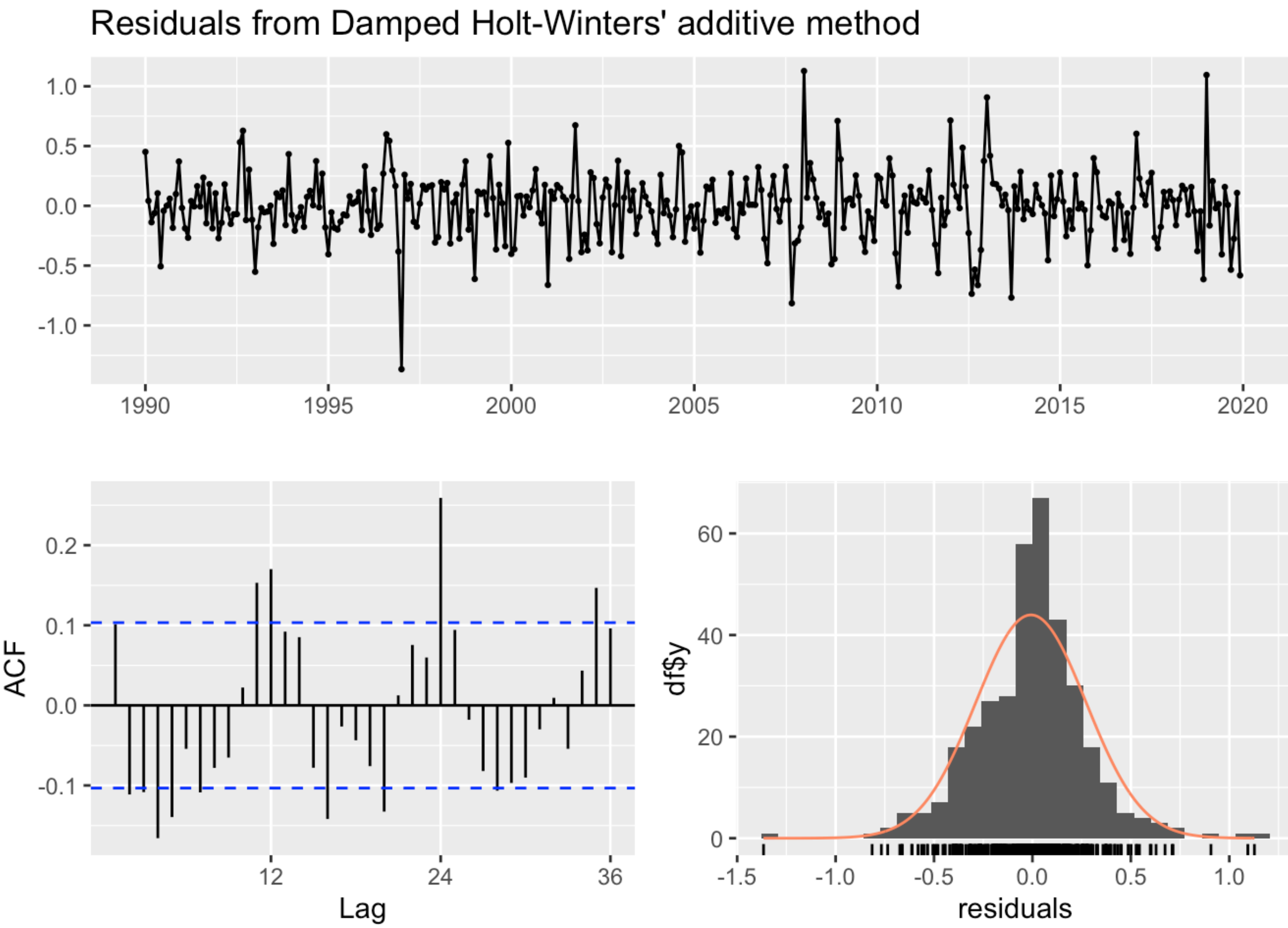

```
# Comparing RMSE Train
print(accuracy(fit_hw_add)[2])
```

```
## [1] 0.2920815
```

```
print(accuracy(fit_hw_add_damp)[2])
```

```
## [1] 0.2806433
```

```
#
checkresiduals(fit_hw_add_damp)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Damped Holt-Winters' additive method
## Q* = 114.26, df = 24, p-value = 1.008e-13
##
## Model df: 0.   Total lags used: 24
```

```
accuracy_hw_add_damp <- accuracy(fit_hw_add_damp, test_extent_N)
accuracy_hw_add_damp
```

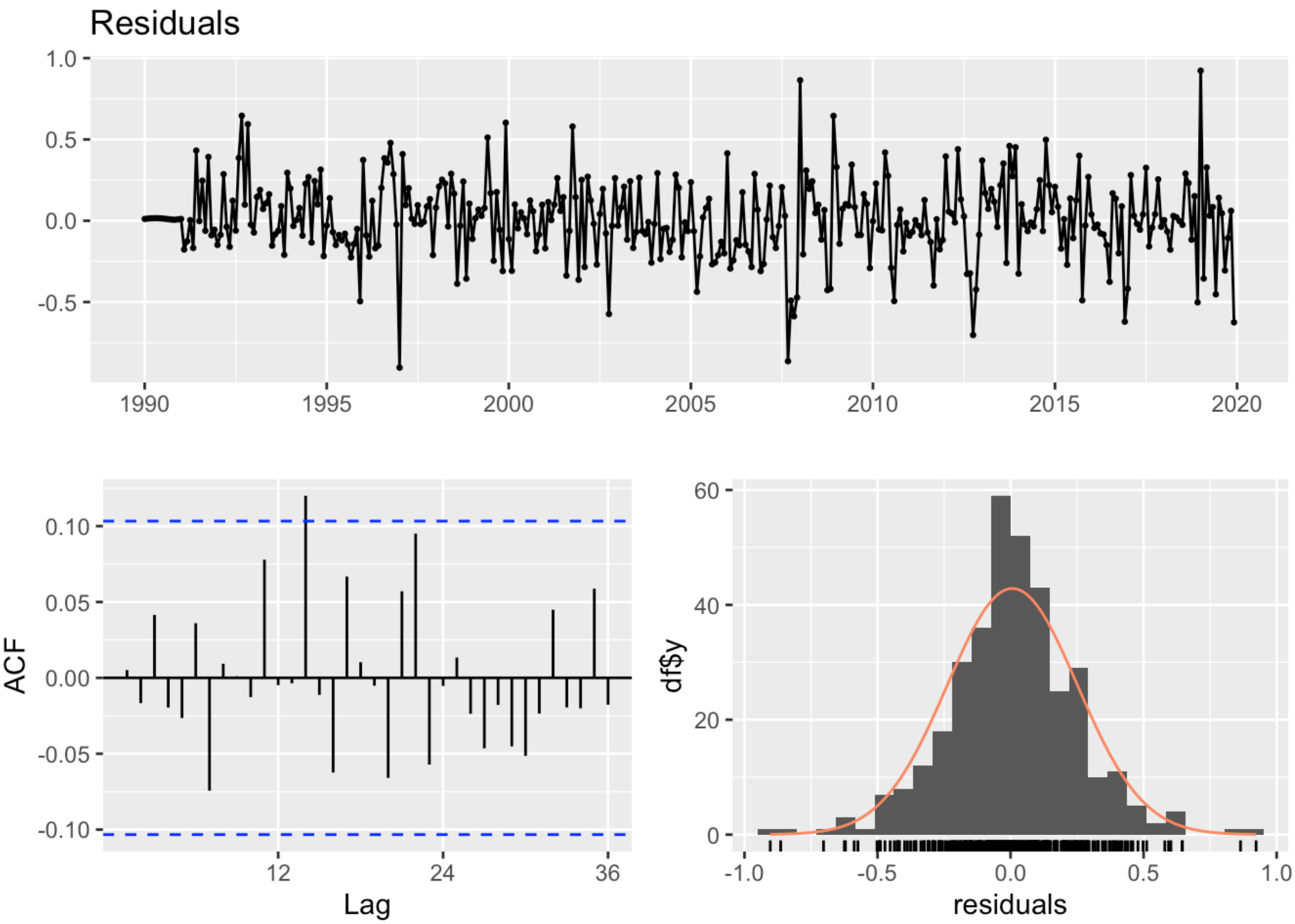
```
##                       ME      RMSE       MAE        MPE     MAPE      MASE
## Training set -0.00758449 0.2806433 0.2026014 -0.2644521  2.16215 0.5881322
## Test set      1.40202303 1.4790112 1.4161840 13.7836164 14.04930 4.1110446
##                   ACF1 Theil's U
## Training set 0.1013728        NA
## Test set     0.7192263 0.6200398
```

Auto Arima

```
arima_model <- auto.arima(train_extent_N, seasonal=TRUE)

arima_model
```

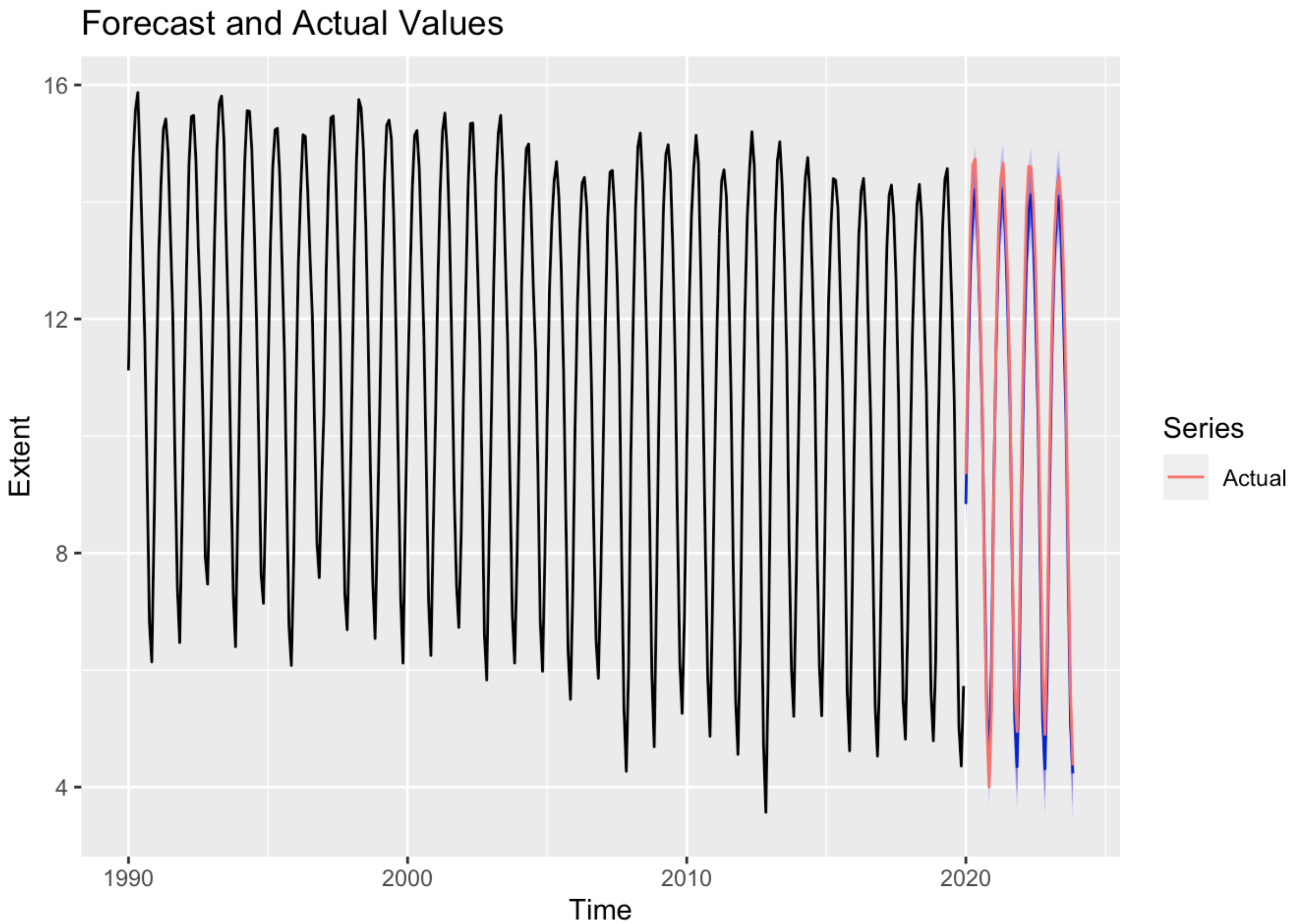
```
## Series: train_extent_N
## ARIMA(2,0,0)(2,1,2)[12] with drift
##
## Coefficients:
##          ar1      ar2     sar1    sar2     sma1     sma2    drift
##       0.8705  -0.1789  -0.6624  0.1020  -0.1334  -0.5246  -0.0051
## s.e.  0.0539   0.0538   0.2918  0.0973   0.2916   0.2024   0.0009
##
## sigma^2 = 0.06315:  log likelihood = -15.89
## AIC=47.78   AICc=48.2   BIC=78.59
```

```
checkresiduals(arima_model$residuals)
```

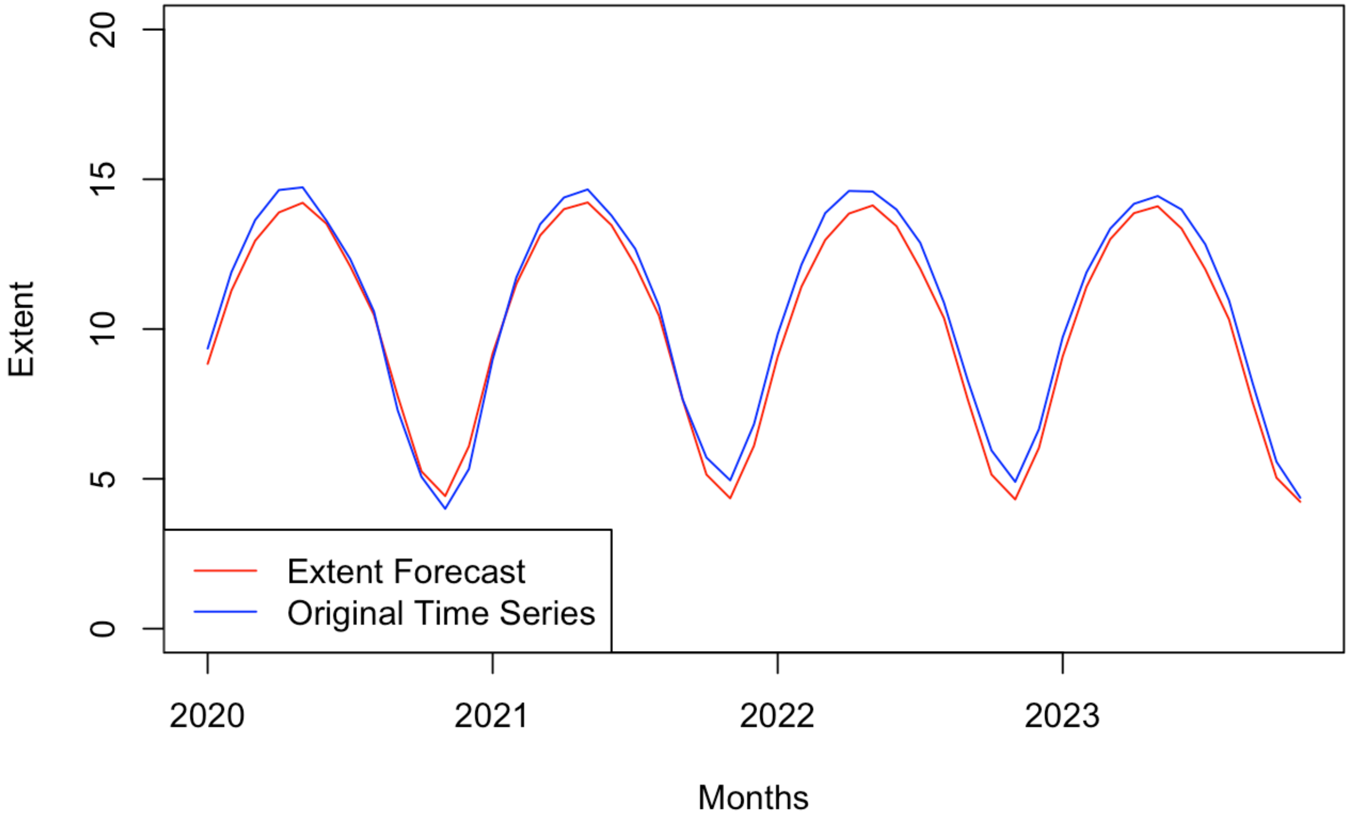



```
##
##  Ljung-Box test
##
## data:  Residuals
## Q* = 22.408, df = 24, p-value = 0.5549
##
## Model df: 0.   Total lags used: 24
```

```
arima_forecast <- forecast(arima_model, h = length(test_extent_N))

autoplot(arima_forecast) +
  autolayer(test_extent_N, series = "Actual") +
  xlab("Time") +
  ylab("Extent") +
  ggtitle("Forecast and Actual Values") +
  guides(color = guide_legend(title = "Series"))
```




```
plot(arima_forecast$mean,  xlab = "Months", ylab = "Extent", col = "red", ylim=c(0,2
0))
lines(test_extent_N, col = "blue")
legend("bottomleft", legend = c("Extent Forecast", "Original Time Series"), col = c("
red", "blue"), lty = c(1,1))
```




```
accuracy(arima_forecast, test_extent_N)
```

```
##                       ME      RMSE       MAE        MPE     MAPE      MASE
## Training set 0.006692764 0.2445760 0.1808994 -0.1175827 1.982333 0.5251335
## Test set     0.419576368 0.5518514 0.5065326  3.9381654 5.523027 1.4704150
##                     ACF1 Theil's U
## Training set 0.005160017        NA
## Test set     0.761791620 0.2848214
```