Project 2 MI3

Bharathi Thambidurai

2025-10-20

In MI2, we cleaned our data and stored the average minimum and maximum temperatures as time series objects in R. We also performed basic EDA to view how temperatures have changed over time, looking at specific months and looking at the overall trend. As we plotted our time series data, it was obvious there was a seasonal component we would have to keep in mind as we moved on to building a model.

```
# from MI2!!
mintemp <- read.csv("C:\\Users\\gbhar\\OneDrive - University of Virginia\\ds 4002\\mean_min_monthly_temp
maxtemp <- read.csv("C:\\Users\\gbhar\\OneDrive - University of Virginia\\ds 4002\\mean_max_monthly_temp

mintemp_dat <- mintemp[mintemp$Year != c(2024,2025), ]
maxtemp_dat <- maxtemp[maxtemp$Year != c(2024,2025), ]

# order of months
month_levels <- c("Jan", "Feb", "Mar", "Apr", "May", "Jun",
                  "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")

# min temp reordered and stored as time series
mintemp_dat$Month <- factor(mintemp_dat$Month, levels = month_levels)
mintemp_sorted <- mintemp_dat[order(mintemp_dat$Year, mintemp_dat$Month), ]

mintemp_timeseries <- ts(mintemp_sorted$Mean_Min_Temp,frequency=12,start=c(2000,1))

# max temp reordered and stored as time series
maxtemp_dat$Month <- factor(maxtemp_dat$Month, levels = month_levels)
maxtemp_sorted <- maxtemp_dat[order(maxtemp_dat$Year, maxtemp_dat$Month), ]

maxtemp_timeseries <- ts(maxtemp_sorted$Value,frequency=12,start=c(2000,1))
```

We used https://a-little-book-of-r-for-time-series.readthedocs.io/en/latest/src/timeseries.html as a reference to guide us in our analysis.

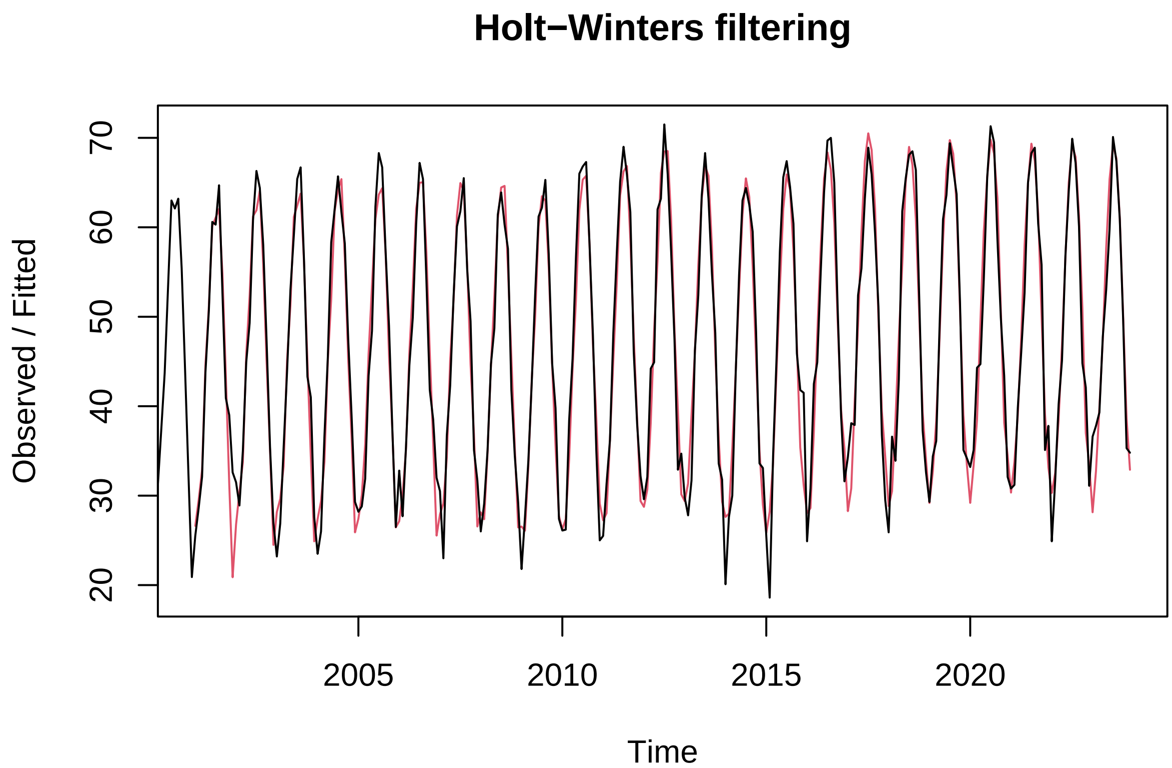
Minimum Average Temperatures

We first built a model using exponential smoothing. Due to the obvious seaonality, we opted to use Holt-Winters exponential smoothing.

```
# fit predictive model
mintempmodel <- HoltWinters(mintemp_timeseries)
mintempmodel # estimated values of alpha, beta, gamma
```

```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = mintemp_timeseries)
##
## Smoothing parameters:
##  alpha: 0.1172071
##  beta : 0.01432105
##  gamma: 0.1856539
##
## Coefficients:
##              [,1]
## a    47.940093133
## b     0.002574581
## s1  -18.105619060
## s2  -15.060639672
## s3   -9.520059313
## s4   -1.797080738
## s5    7.640912560
## s6   15.723107829
## s7   21.270803516
## s8   19.466157860
## s9   12.894557061
## s10   1.806070982
## s11 -10.017447454
## s12 -14.509994740
```

```
plot(mintempmodel) # plot model on top of existing data
```



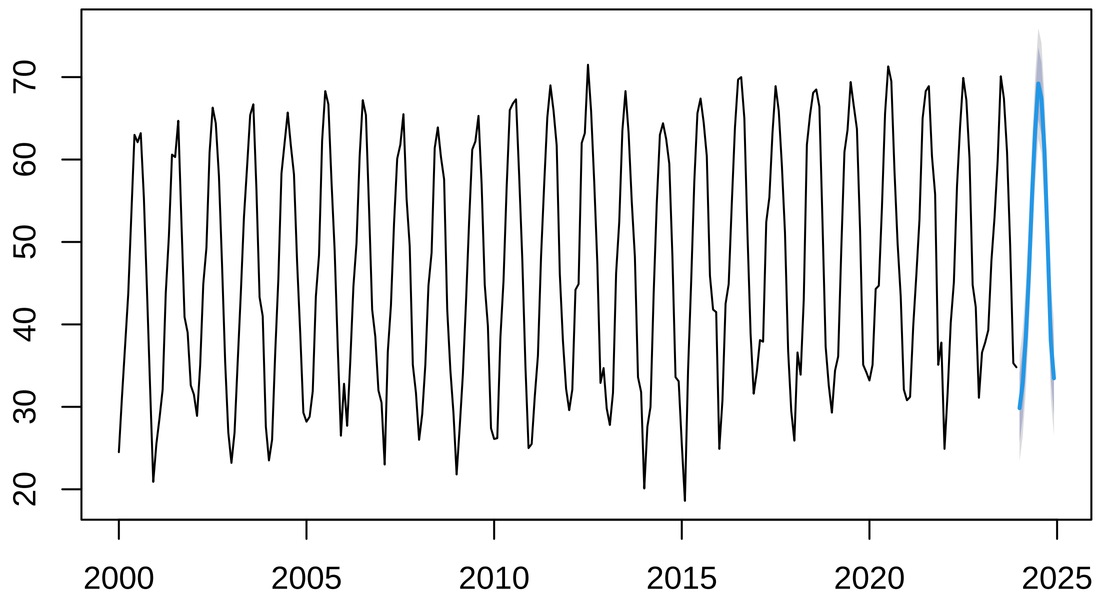

```r
library(forecast)
```

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

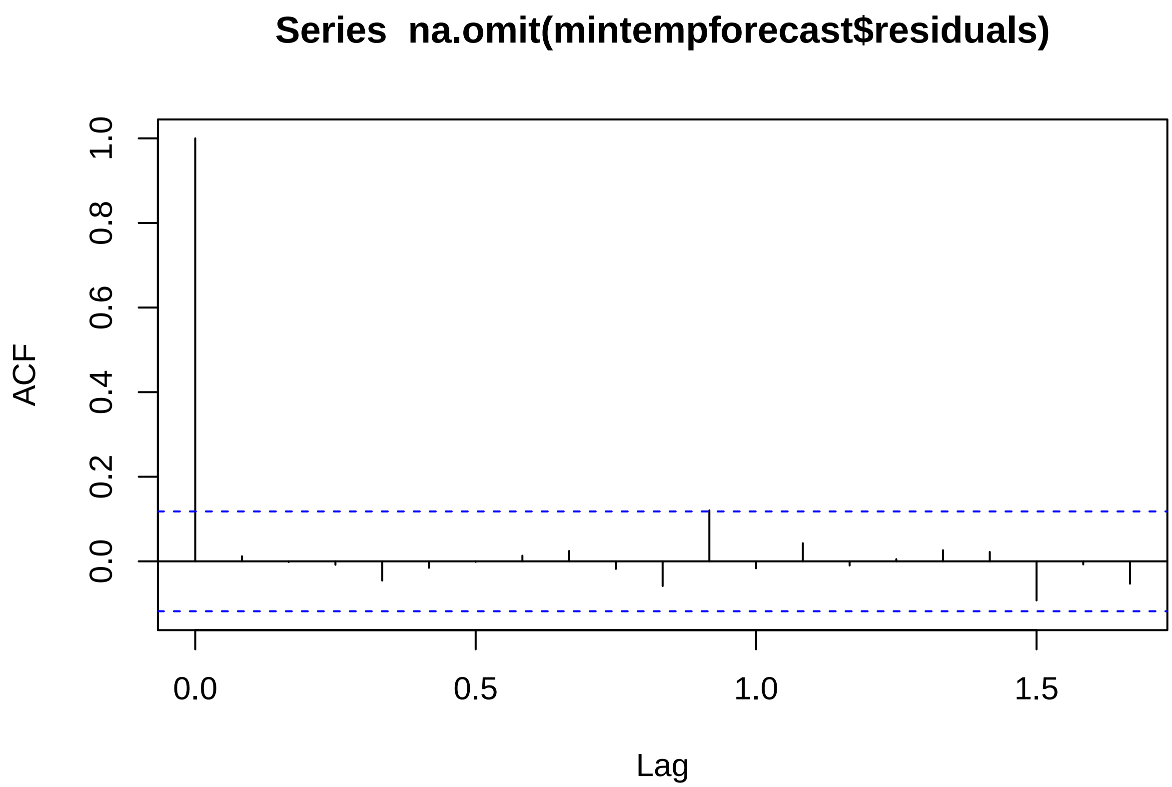
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```r
# forecasting
mintempforecast <- forecast(mintempmodel, h=12) # 12 more months (2024)
plot(mintempforecast) # plot our forecast values
```

Forecasts from HoltWinters




```r
# checking if there are any non-zero autocorrelations
acf(na.omit(mintempforecast$residuals), lag.max=20) # correlogram
```




```r
Box.test(na.omit(mintempforecast$residuals), lag=20, type = "Ljung-Box") # Ljung-Box test
```

```
##
##  Box-Ljung test
##
## data:  na.omit(mintempforecast$residuals)
## X-squared = 10.616, df = 20, p-value = 0.9556
```

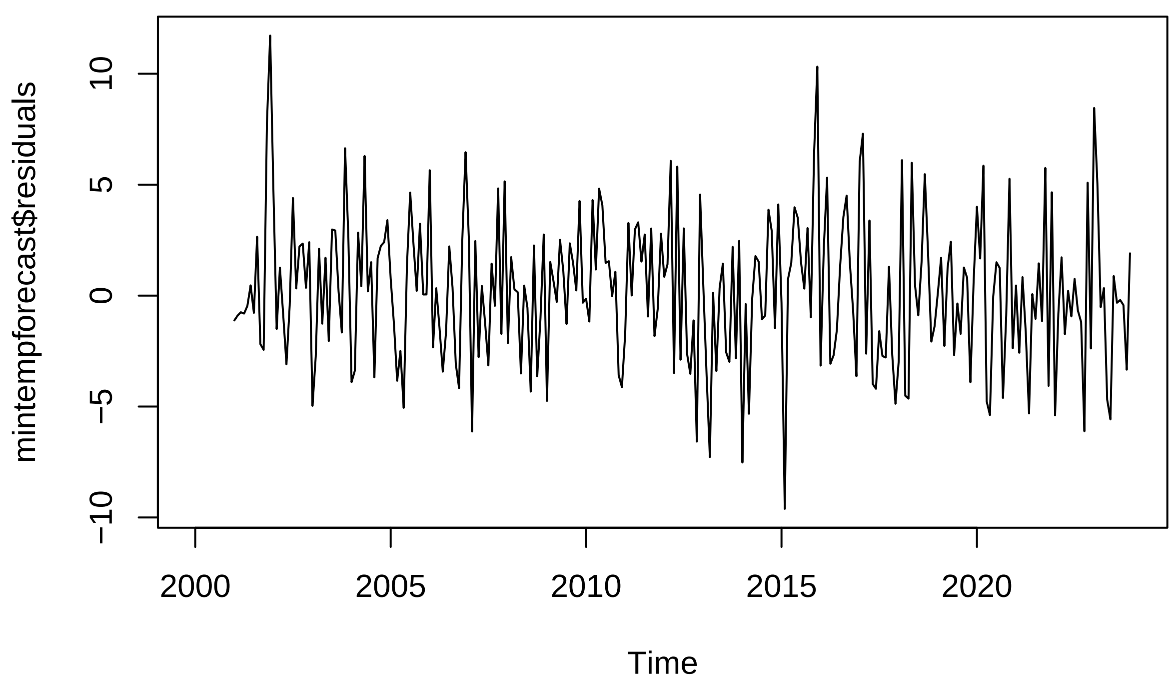
Fitting our model and forecasting for 2024 shows that the model does pick up on seasonality and follow existing trends. To check our accuracy, we can see in the correlogram that autocorrelations for forecast errors do not exceed significance bounds for almost all lags. The p-value for our Ljung-Box test is 0.95, further proving there is little evidence of non-zero autocorrelations at lags 1-20.

We can also check if forecast errors have constant variance over time and are normally distributed with mean zero.

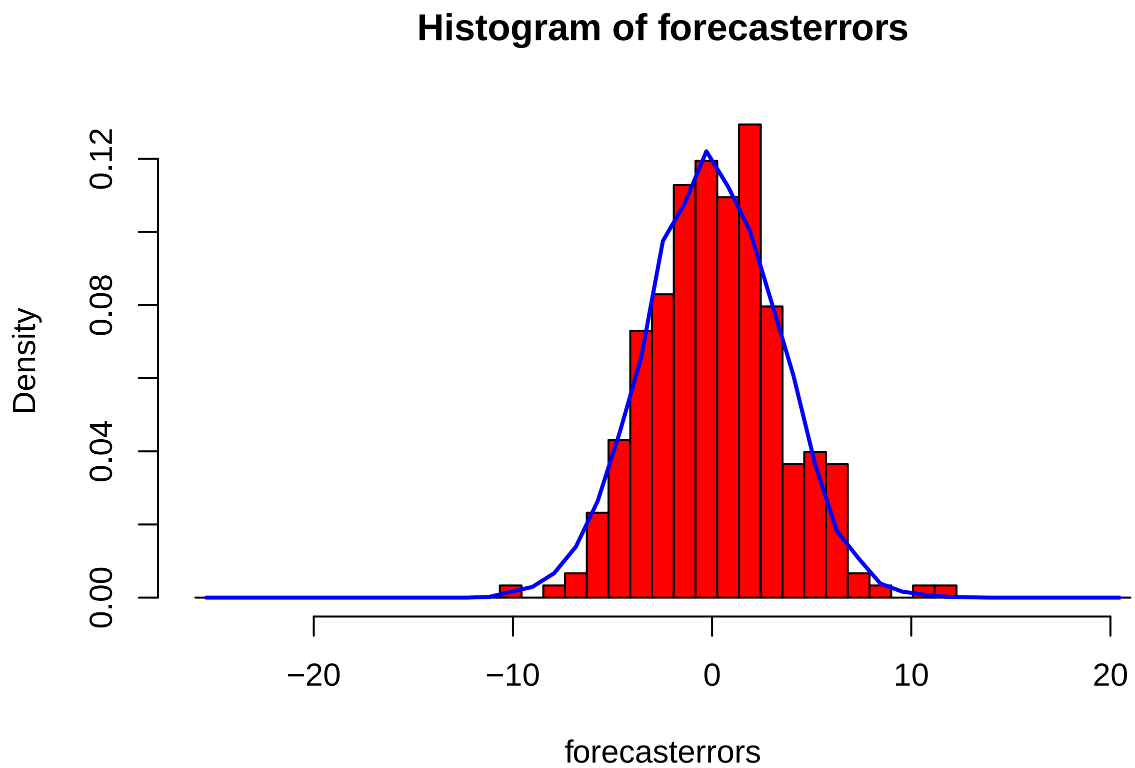
```r
# from the guided tutorial we followed
plotForecastErrors <- function(forecasterrors)
  {
    forecasterrors <- na.omit(forecasterrors)
     # make a histogram of the forecast errors:
     mybinsize <- IQR(forecasterrors)/4
     mysd   <- sd(forecasterrors, na.rm=T)
     mymin  <- min(forecasterrors, na.rm=T) - mysd*5
     mymax  <- max(forecasterrors, na.rm=T) + mysd*3
```

```
    # generate normally distributed data with mean 0 and standard deviation mysd
    mynorm <- rnorm(10000, mean=0, sd=mysd)
    mymin2 <- min(mynorm)
    mymax2 <- max(mynorm)
    if (mymin2 < mymin) { mymin <- mymin2 }
    if (mymax2 > mymax) { mymax <- mymax2 }
    # make a red histogram of the forecast errors, with the normally distributed data overlaid:
    mybins <- seq(mymin, mymax, mybinsize)
    hist(forecasterrors, col="red", freq=FALSE, breaks=mybins)
    # freq=FALSE ensures the area under the histogram = 1
    # generate normally distributed data with mean 0 and standard deviation mysd
    myhist <- hist(mynorm, plot=FALSE, breaks=mybins)
    # plot the normal curve as a blue line on top of the histogram of forecast errors:
    points(myhist$mids, myhist$density, type="l", col="blue", lwd=2)
  }


plot.ts(mintempforecast$residuals)
```




```
plotForecastErrors(mintempforecast$residuals) # normally distributed with mean 0
```



From the time plot, we can see that the forecast errors have constant variance over time. From the histogram, we can also assume that forecast errors are normally distributed with mean zero.

ARIMA MODEL

While Holt-Winters exponential smoothing provides an adequate predictive model, we can make a better model by taking correlations in the data into account. We wanted to try making an ARIMA (Autoregressive Integrated Moving Average) model as well.

Based on the tutorial and some trial-and-error in our model building, we realized our data needed to be differenced prior to determing what ARIMA parameters would best fit our model.

Once forcing a manual difference, we can use the auto.arima function to figure out the parameters of our ARIMA model.

```
mintemp_arima <- auto.arima(
  mintemp_timeseries,
  d = 1,                # force one regular difference
  D = 1,                # seasonal difference (12-month)
  max.p = 5, max.q = 5,
  max.P = 2, max.Q = 2,
  stepwise = FALSE, approx = FALSE,
  trace = TRUE
)
```

```
##
##  ARIMA(0,1,0)(0,1,0)[12]                    : 1753.354
##  ARIMA(0,1,0)(0,1,1)[12]                    : Inf
```

```
## ARIMA(0,1,0)(0,1,2)[12]                    : 1589.917
## ARIMA(0,1,0)(1,1,0)[12]                    : 1665.576
## ARIMA(0,1,0)(1,1,1)[12]                    : 1590.103
## ARIMA(0,1,0)(1,1,2)[12]                    : Inf
## ARIMA(0,1,0)(2,1,0)[12]                    : 1643.012
## ARIMA(0,1,0)(2,1,1)[12]                    : 1591.984
## ARIMA(0,1,0)(2,1,2)[12]                    : Inf
## ARIMA(0,1,1)(0,1,0)[12]                    : Inf
## ARIMA(0,1,1)(0,1,1)[12]                    : Inf
## ARIMA(0,1,1)(0,1,2)[12]                    : Inf
## ARIMA(0,1,1)(1,1,0)[12]                    : 1513.352
## ARIMA(0,1,1)(1,1,1)[12]                    : Inf
## ARIMA(0,1,1)(1,1,2)[12]                    : Inf
## ARIMA(0,1,1)(2,1,0)[12]                    : 1490.029
## ARIMA(0,1,1)(2,1,1)[12]                    : Inf
## ARIMA(0,1,1)(2,1,2)[12]                    : Inf
## ARIMA(0,1,2)(0,1,0)[12]                    : Inf
## ARIMA(0,1,2)(0,1,1)[12]                    : Inf
## ARIMA(0,1,2)(0,1,2)[12]                    : Inf
## ARIMA(0,1,2)(1,1,0)[12]                    : 1515.409
## ARIMA(0,1,2)(1,1,1)[12]                    : Inf
## ARIMA(0,1,2)(1,1,2)[12]                    : Inf
## ARIMA(0,1,2)(2,1,0)[12]                    : 1491.978
## ARIMA(0,1,2)(2,1,1)[12]                    : Inf
## ARIMA(0,1,3)(0,1,0)[12]                    : Inf
## ARIMA(0,1,3)(0,1,1)[12]                    : Inf
## ARIMA(0,1,3)(0,1,2)[12]                    : Inf
## ARIMA(0,1,3)(1,1,0)[12]                    : 1517.363
## ARIMA(0,1,3)(1,1,1)[12]                    : Inf
## ARIMA(0,1,3)(2,1,0)[12]                    : 1494.027
## ARIMA(0,1,4)(0,1,0)[12]                    : Inf
## ARIMA(0,1,4)(0,1,1)[12]                    : Inf
## ARIMA(0,1,4)(1,1,0)[12]                    : 1519.449
## ARIMA(0,1,5)(0,1,0)[12]                    : Inf
## ARIMA(1,1,0)(0,1,0)[12]                    : 1661.144
## ARIMA(1,1,0)(0,1,1)[12]                    : Inf
## ARIMA(1,1,0)(0,1,2)[12]                    : Inf
## ARIMA(1,1,0)(1,1,0)[12]                    : 1587.862
## ARIMA(1,1,0)(1,1,1)[12]                    : Inf
## ARIMA(1,1,0)(1,1,2)[12]                    : Inf
## ARIMA(1,1,0)(2,1,0)[12]                    : 1567.045
## ARIMA(1,1,0)(2,1,1)[12]                    : Inf
## ARIMA(1,1,0)(2,1,2)[12]                    : Inf
## ARIMA(1,1,1)(0,1,0)[12]                    : Inf
## ARIMA(1,1,1)(0,1,1)[12]                    : Inf
## ARIMA(1,1,1)(0,1,2)[12]                    : Inf
## ARIMA(1,1,1)(1,1,0)[12]                    : Inf
## ARIMA(1,1,1)(1,1,1)[12]                    : Inf
## ARIMA(1,1,1)(1,1,2)[12]                    : Inf
## ARIMA(1,1,1)(2,1,0)[12]                    : Inf
## ARIMA(1,1,1)(2,1,1)[12]                    : Inf
## ARIMA(1,1,2)(0,1,0)[12]                    : Inf
## ARIMA(1,1,2)(0,1,1)[12]                    : Inf
## ARIMA(1,1,2)(0,1,2)[12]                    : Inf
```

```
##  ARIMA(1,1,2)(1,1,0)[12]                    : Inf
##  ARIMA(1,1,2)(1,1,1)[12]                    : Inf
##  ARIMA(1,1,2)(2,1,0)[12]                    : 1494.057
##  ARIMA(1,1,3)(0,1,0)[12]                    : Inf
##  ARIMA(1,1,3)(0,1,1)[12]                    : Inf
##  ARIMA(1,1,3)(1,1,0)[12]                    : Inf
##  ARIMA(1,1,4)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,0)(0,1,0)[12]                    : 1630.335
##  ARIMA(2,1,0)(0,1,1)[12]                    : Inf
##  ARIMA(2,1,0)(0,1,2)[12]                    : Inf
##  ARIMA(2,1,0)(1,1,0)[12]                    : 1562.511
##  ARIMA(2,1,0)(1,1,1)[12]                    : Inf
##  ARIMA(2,1,0)(1,1,2)[12]                    : Inf
##  ARIMA(2,1,0)(2,1,0)[12]                    : 1532.239
##  ARIMA(2,1,0)(2,1,1)[12]                    : Inf
##  ARIMA(2,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,1)(0,1,1)[12]                    : Inf
##  ARIMA(2,1,1)(0,1,2)[12]                    : Inf
##  ARIMA(2,1,1)(1,1,0)[12]                    : Inf
##  ARIMA(2,1,1)(1,1,1)[12]                    : Inf
##  ARIMA(2,1,1)(2,1,0)[12]                    : 1494.019
##  ARIMA(2,1,2)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,2)(0,1,1)[12]                    : Inf
##  ARIMA(2,1,2)(1,1,0)[12]                    : 1519.455
##  ARIMA(2,1,3)(0,1,0)[12]                    : Inf
##  ARIMA(3,1,0)(0,1,0)[12]                    : 1620.793
##  ARIMA(3,1,0)(0,1,1)[12]                    : Inf
##  ARIMA(3,1,0)(0,1,2)[12]                    : Inf
##  ARIMA(3,1,0)(1,1,0)[12]                    : 1550.529
##  ARIMA(3,1,0)(1,1,1)[12]                    : Inf
##  ARIMA(3,1,0)(2,1,0)[12]                    : 1523.49
##  ARIMA(3,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(3,1,1)(0,1,1)[12]                    : Inf
##  ARIMA(3,1,1)(1,1,0)[12]                    : 1519.443
##  ARIMA(3,1,2)(0,1,0)[12]                    : Inf
##  ARIMA(4,1,0)(0,1,0)[12]                    : 1614.582
##  ARIMA(4,1,0)(0,1,1)[12]                    : Inf
##  ARIMA(4,1,0)(1,1,0)[12]                    : 1545.551
##  ARIMA(4,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(5,1,0)(0,1,0)[12]                    : 1604.366
##
##
##
##  Best model: ARIMA(0,1,1)(2,1,0)[12]
```

```
mintemp_arima
```

```
## Series: mintemp_timeseries
## ARIMA(0,1,1)(2,1,0)[12]
##
## Coefficients:
##           ma1     sar1     sar2
##       -0.8623  -0.6417  -0.3145
## s.e.   0.0341   0.0598   0.0603
```

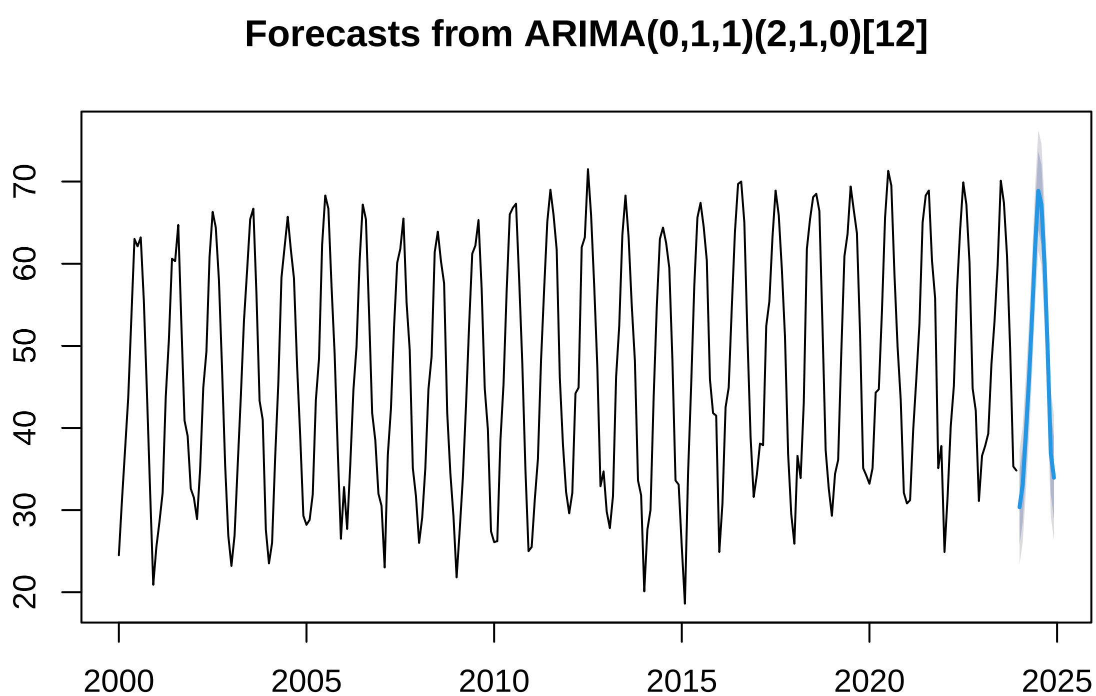
```
##
## sigma^2 = 12.62:  log likelihood = -740.94
## AIC=1489.88   AICc=1490.03   BIC=1504.35
```

Based on the auto.arima() function, we can determine that an ARIMA(0,1,1)x(2,1,0)[12] best fits our data. We can use this model to forecast for 2024. We want to check that this model meets all assumptions.

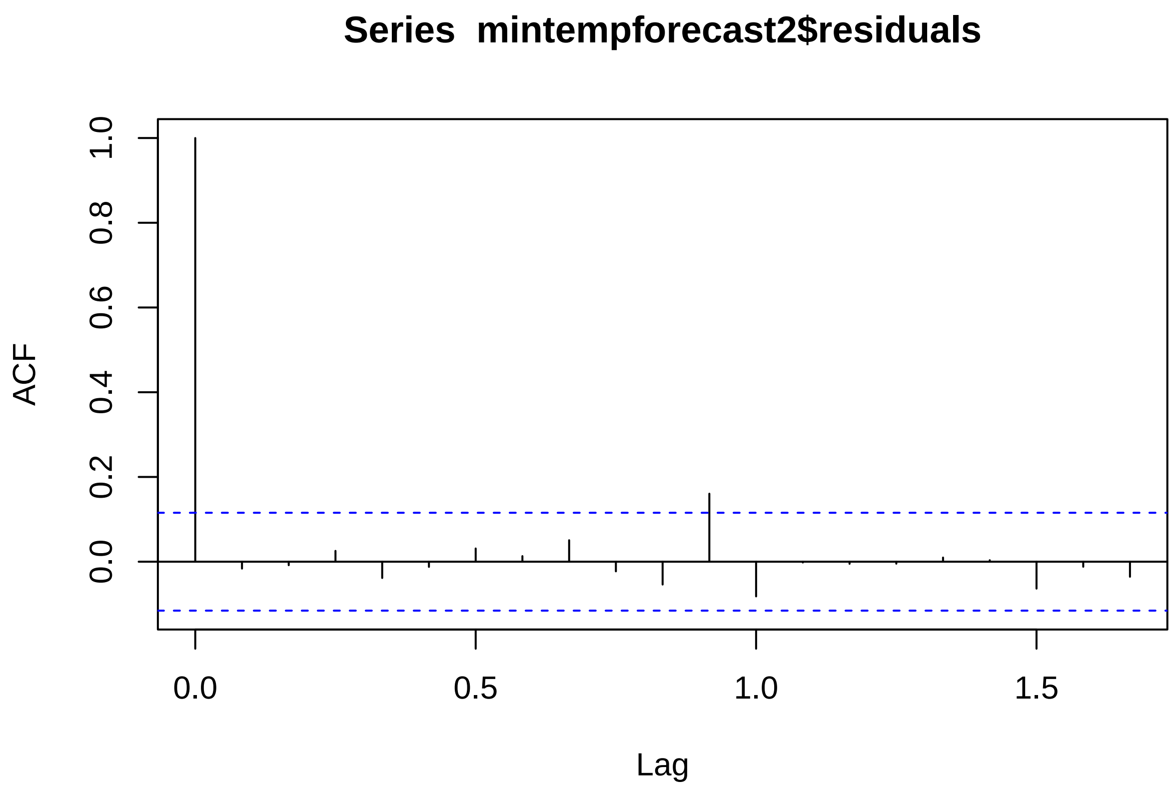
```
mintempforecast2 <- forecast(mintemp_arima, h=12) # forecast using ARIMA model
mintempforecast2
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 2024       30.34521 25.79311 34.89731 23.38337 37.30705
## Feb 2024       33.12620 28.53111 37.72128 26.09862 40.15377
## Mar 2024       39.14963 34.51196 43.78729 32.05693 46.24232
## Apr 2024       45.71244 41.03259 50.39230 38.55522 52.86967
## May 2024       53.54706 48.82538 58.26873 46.32588 60.76824
## Jun 2024       62.24026 57.47713 67.00338 54.95569 69.52483
## Jul 2024       68.86638 64.06216 73.67060 61.51897 76.21380
## Aug 2024       67.20413 62.35917 72.04909 59.79440 74.61386
## Sep 2024       59.87884 54.99348 64.76421 52.40732 67.35036
## Oct 2024       49.37681 44.45137 54.30225 41.84400 56.90962
## Nov 2024       36.86023 31.89505 41.82542 29.26664 44.45383
## Dec 2024       33.93048 28.92586 38.93510 26.27657 41.58438
```

```
plot(mintempforecast2) # add forecast values into time series plot
```



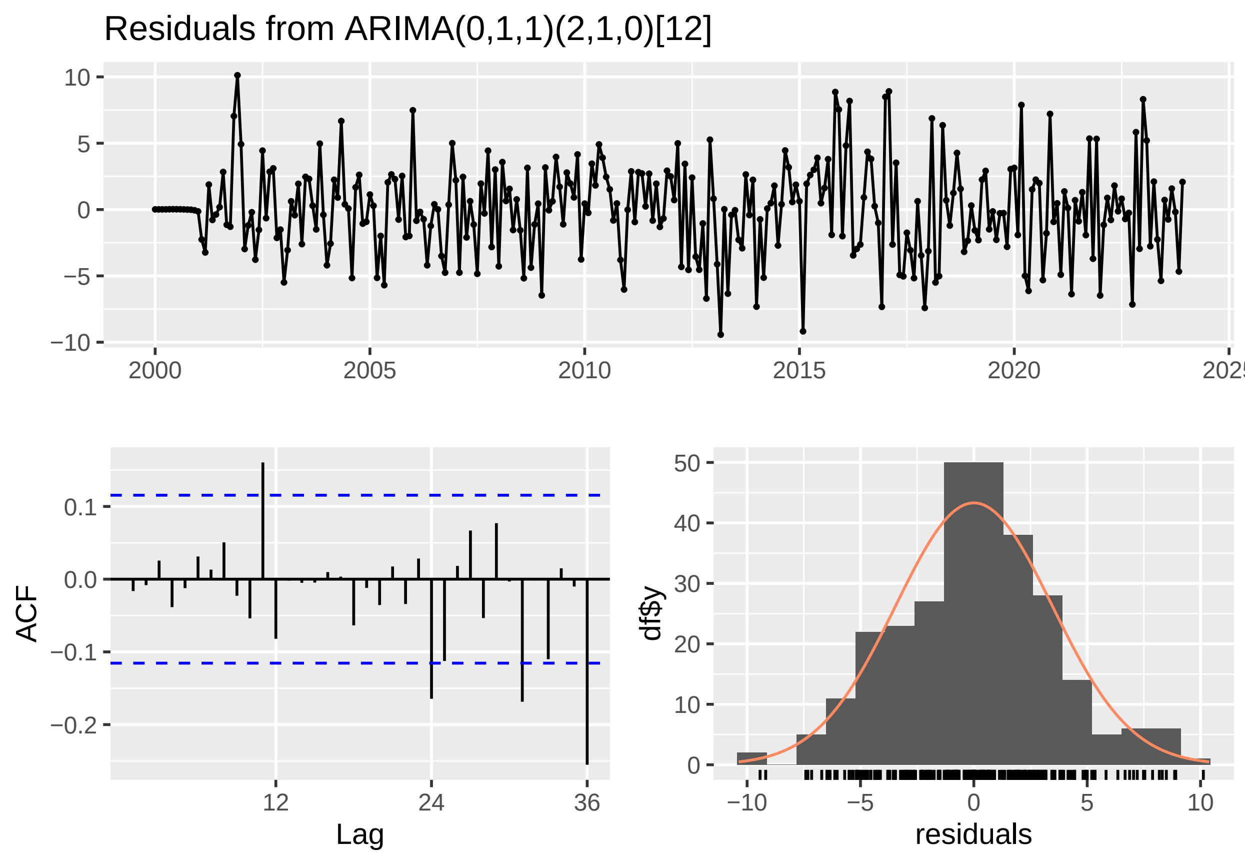
```r
acf(mintempforecast2$residuals,lag.max=20) # correlogram
```




```r
Box.test(mintempforecast2$residuals, lag=20, type="Ljung-Box") # Ljung-Box test
```

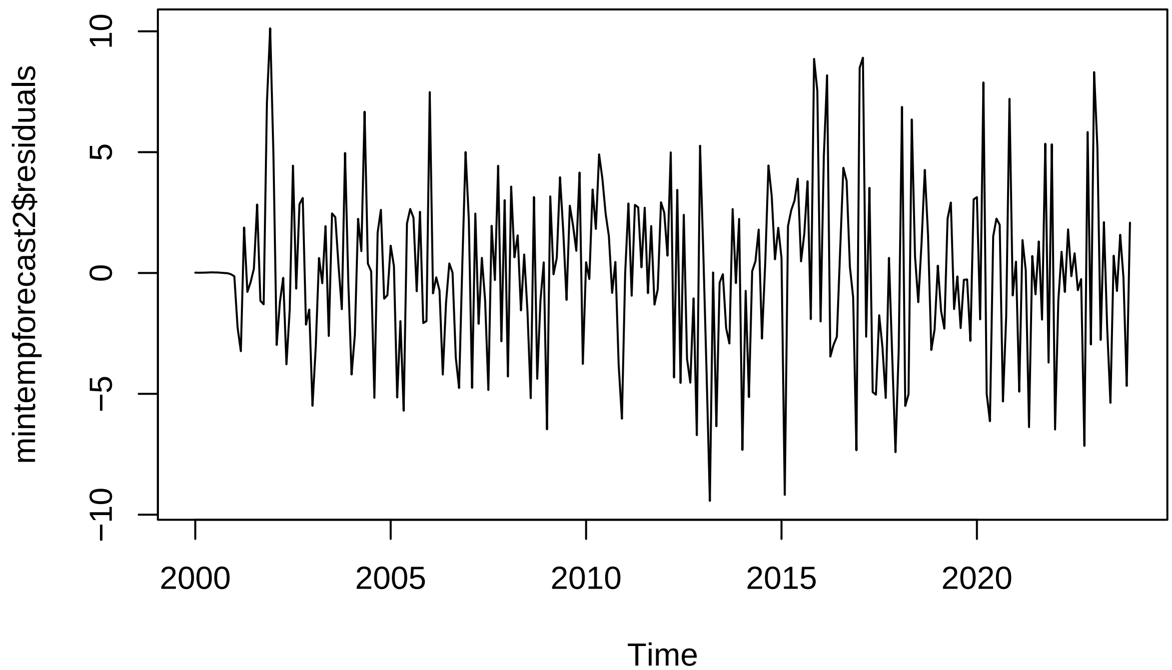
```
##
##  Box-Ljung test
##
## data:  mintempforecast2$residuals
## X-squared = 14.434, df = 20, p-value = 0.8078
```

```r
checkresiduals(mintemp_arima) # chceck that residuals are insignificant
```

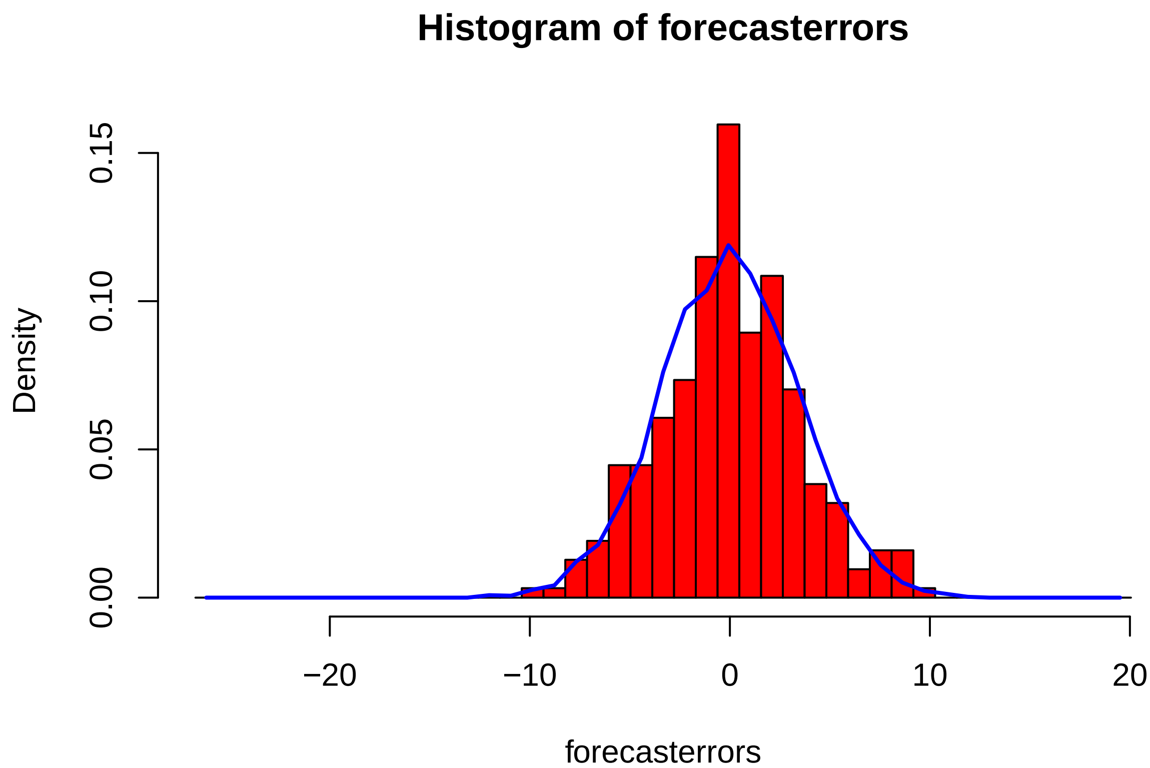



```
##
## 	Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)(2,1,0)[12]
## Q* = 23.707, df = 21, p-value = 0.3075
##
## Model df: 3.   Total lags used: 24
```

```
plot.ts(mintempforecast2$residuals)
```




```
plotForecastErrors(mintempforecast2$residuals) # normally distributed with mean 0
```



Based on our model and forecast values, we see that the correlogram shows that barely any of the sample autocorrelations for lags 1-20 exceed significance bounds, the p-value of the Ljung-Box test is 0.81, and the p-value of our residuals is 0.31. We can conclude there is little evidence for non-zero autocorrelations in the forecast errors.

From the time plot, we can see that the forecast errors have constant variance over time. From the histogram, we can also assume that forecast errors are normally distributed with mean zero.

Now, let's compare our forecast values from the ARIMA model to our actual values for 2024.

```
# extract 2024 from our data
mintemp_2024 <- mintemp[mintemp$Year==2024, ]
mintemp_2024
```

```
##     Month_Year Year Month Mean_Min_Temp
## 25    Jan 2024 2024   Jan          30.9
## 51    Feb 2024 2024   Feb          34.5
## 77    Mar 2024 2024   Mar          42.8
## 103   Apr 2024 2024   Apr          51.4
## 129   May 2024 2024   May          58.9
## 155   Jun 2024 2024   Jun          65.4
## 181   Jul 2024 2024   Jul          68.1
## 207   Aug 2024 2024   Aug          65.0
## 233   Sep 2024 2024   Sep          58.5
## 259   Oct 2024 2024   Oct          46.2
## 285   Nov 2024 2024   Nov          40.8
## 311   Dec 2024 2024   Dec          30.1
```

```r
mintempforecast2
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 2024       30.34521 25.79311 34.89731 23.38337 37.30705
## Feb 2024       33.12620 28.53111 37.72128 26.09862 40.15377
## Mar 2024       39.14963 34.51196 43.78729 32.05693 46.24232
## Apr 2024       45.71244 41.03259 50.39230 38.55522 52.86967
## May 2024       53.54706 48.82538 58.26873 46.32588 60.76824
## Jun 2024       62.24026 57.47713 67.00338 54.95569 69.52483
## Jul 2024       68.86638 64.06216 73.67060 61.51897 76.21380
## Aug 2024       67.20413 62.35917 72.04909 59.79440 74.61386
## Sep 2024       59.87884 54.99348 64.76421 52.40732 67.35036
## Oct 2024       49.37681 44.45137 54.30225 41.84400 56.90962
## Nov 2024       36.86023 31.89505 41.82542 29.26664 44.45383
## Dec 2024       33.93048 28.92586 38.93510 26.27657 41.58438
```

By comparing our forecast values of 2024 to our actual values of 2024, we can see that all the actual data falls in the 95% prediction interval for the forecast. Thus, we meet our goal of being at least 80% accurate with our model.

Maximum Average Temperatures

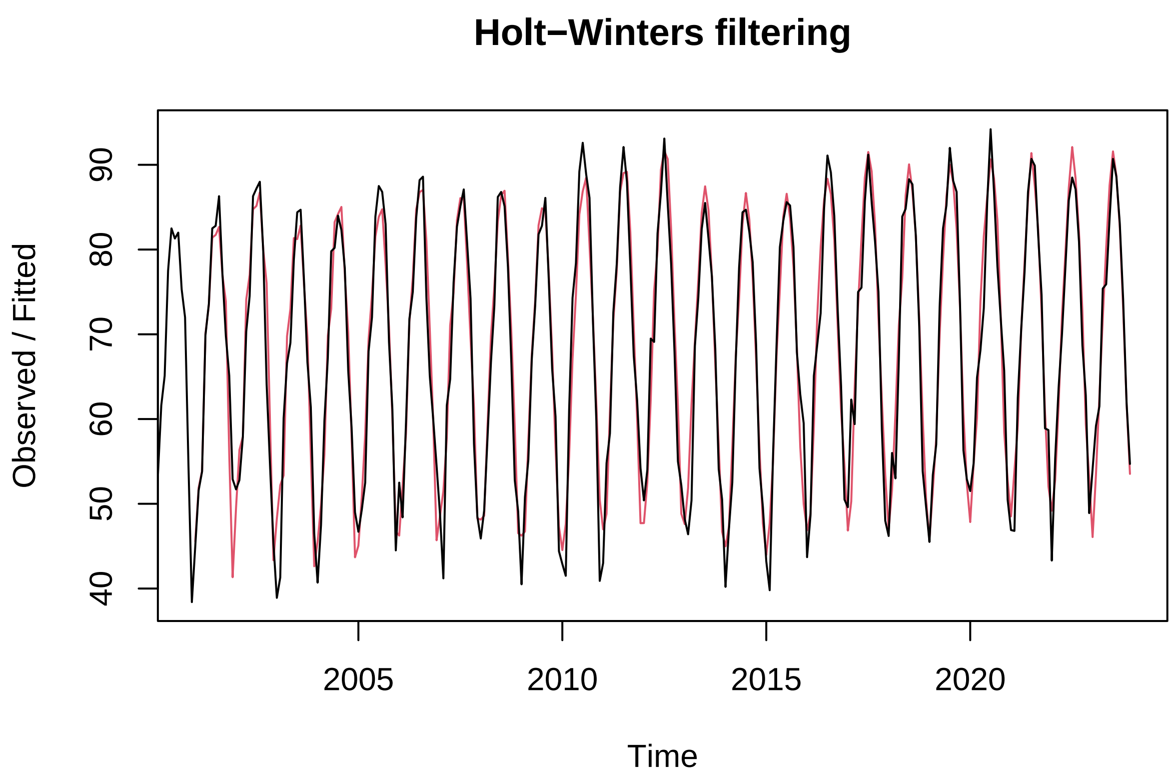
We again built a model using exponential smoothing. Due to the obvious seaonality, we opted to use Holt-Winters exponential smoothing.

```r
# fit predictive model
maxtempmodel <- HoltWinters(maxtemp_timeseries)
maxtempmodel # estimated values of alpha, beta, gamma
```

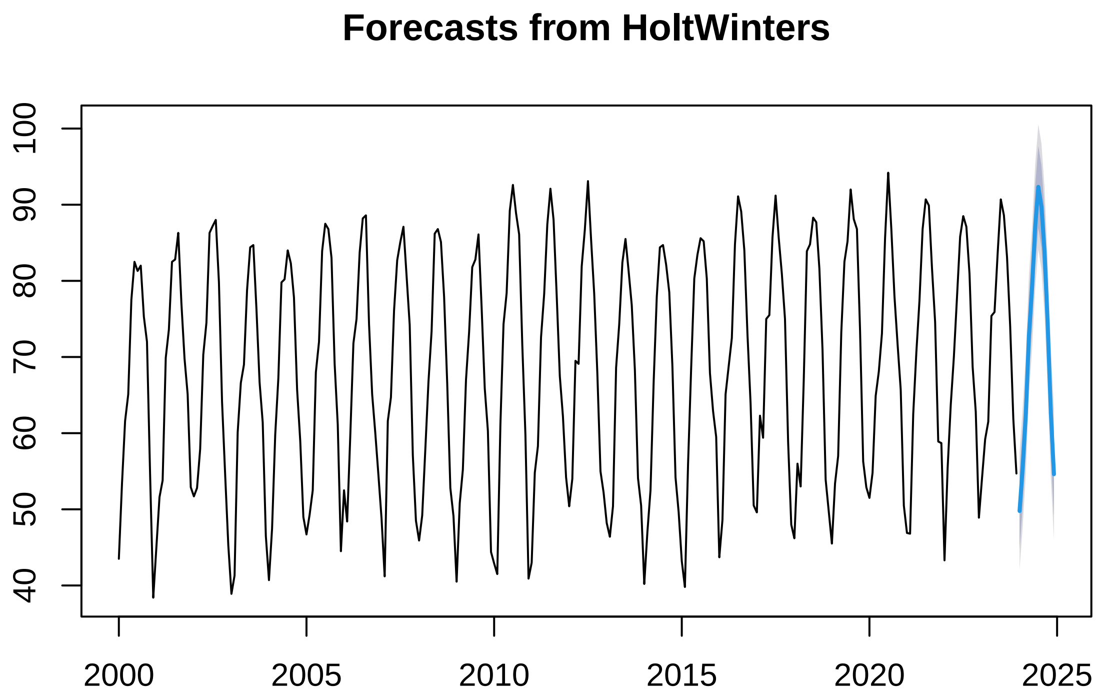
```
## Holt-Winters exponential smoothing with trend and additive seasonal component.
##
## Call:
## HoltWinters(x = maxtemp_timeseries)
##
## Smoothing parameters:
##  alpha: 0.1484763
##  beta : 0.001741349
##  gamma: 0.2358454
##
## Coefficients:
##             [,1]
## a    72.78009694
## b     0.05615067
## s1  -23.05466568
## s2  -17.37428096
## s3  -10.47583962
## s4   -0.20418467
## s5    6.01131442
## s6   13.99878463
## s7   19.16614683
## s8   16.54661897
```

```
## s9   10.52109934
## s10   0.88235963
## s11 -10.66149562
## s12 -18.84281187
```

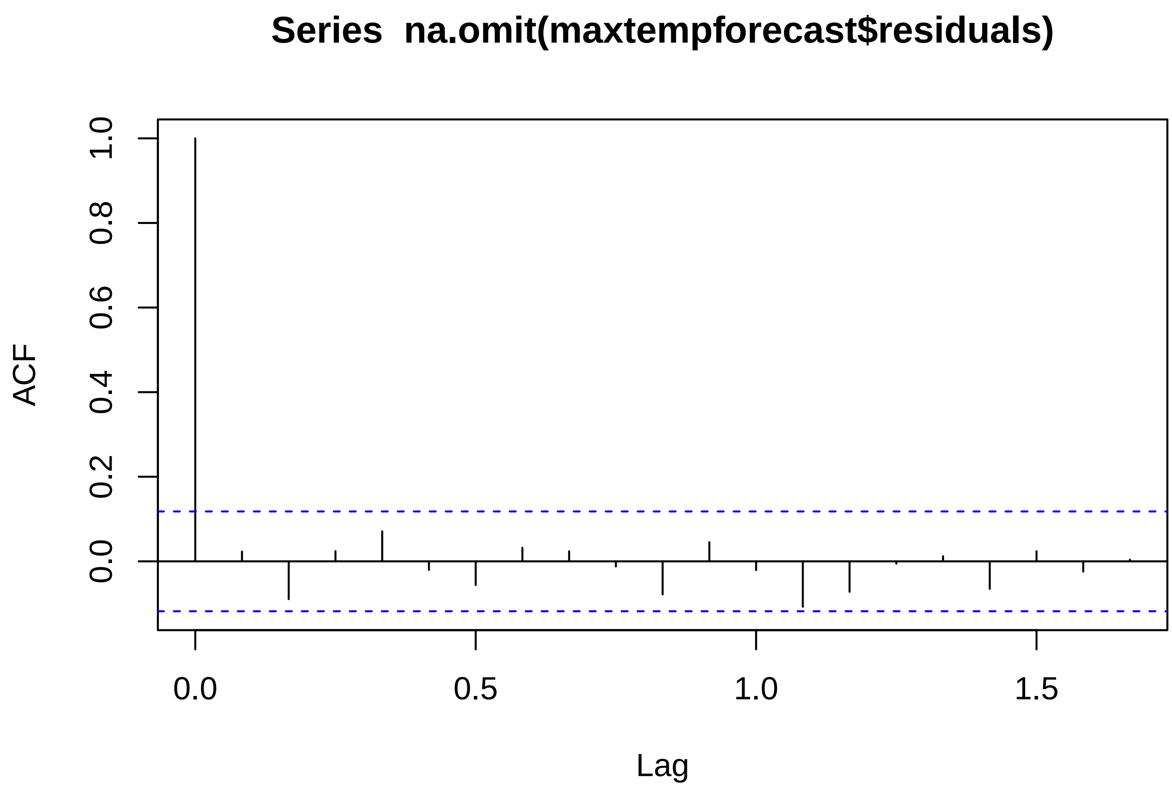
```
plot(maxtempmodel) # plot model on top of existing data
```




```
# forecasting
maxtempforecast <- forecast(maxtempmodel, h=12) # 12 more months (2024)
plot(maxtempforecast) # plot our forecast values
```




```r
# checking if there are any non-zero autocorrelations
acf(na.omit(maxtempforecast$residuals), lag.max=20) # correlogram
```



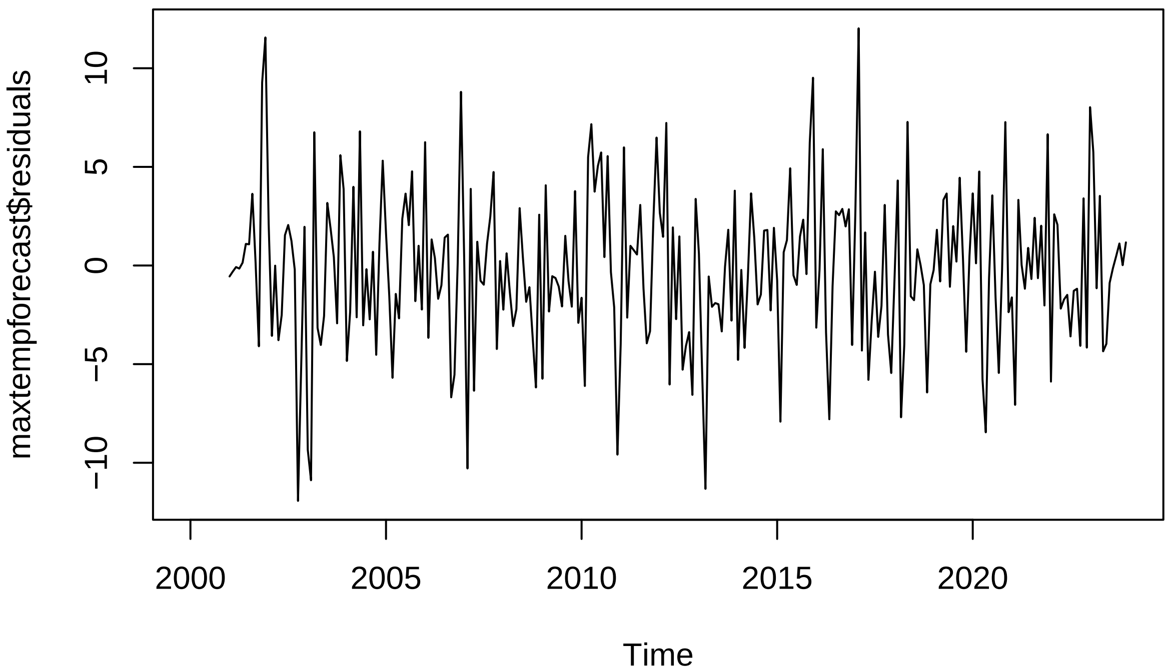

```r
Box.test(na.omit(maxtempforecast$residuals), lag=20, type = "Ljung-Box") # Ljung-Box test
```

```
##
##  Box-Ljung test
##
## data:  na.omit(maxtempforecast$residuals)
## X-squared = 14.52, df = 20, p-value = 0.8032
```

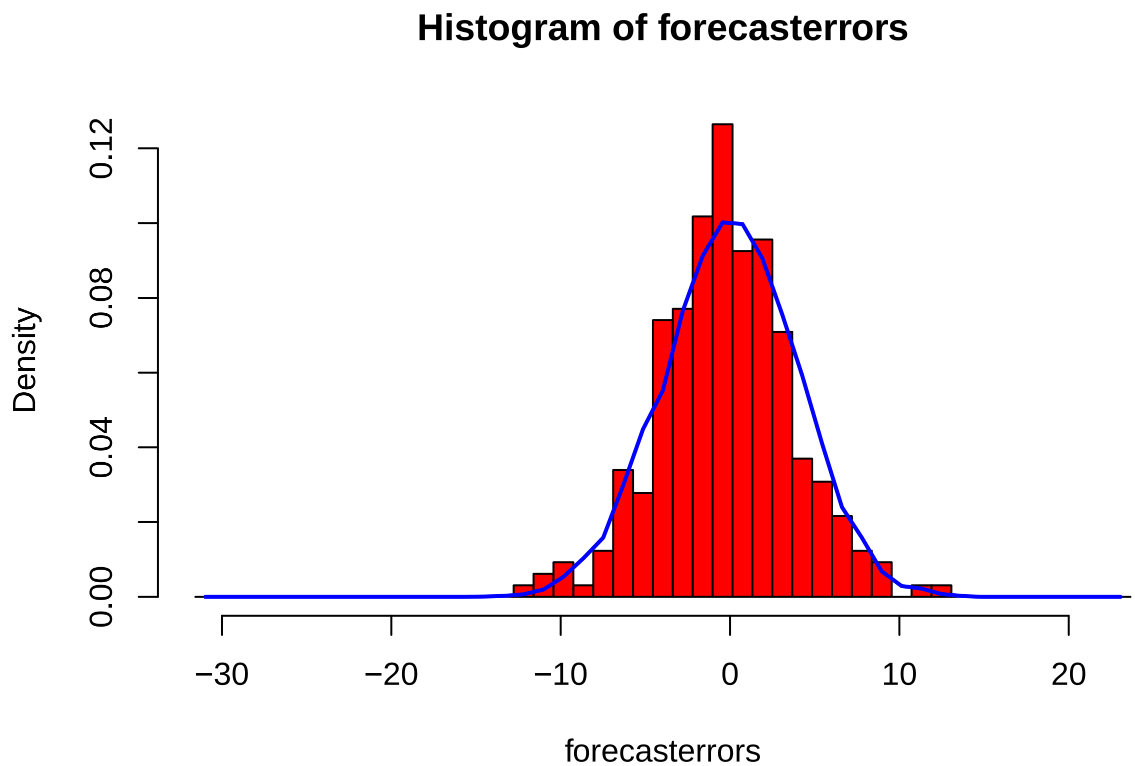
Fitting our model and forecasting for 2024 shows that the model again does pick up on seasonality and follow existing trends. To check our accuracy, we can see in the correlogram that autocorrelations for forecast errors do not exceed significance bounds for all lags. The p-value for our Ljung-Box test is 0.80, further proving there is little evidence of non-zero autocorrelations at lags 1-20.

We can also check if forecast errors have constant variance over time and are normally distributed with mean zero.

```r
plot.ts(maxtempforecast$residuals)
```




```
plotForecastErrors(maxtempforecast$residuals) # normally distributed with mean 0
```



From the time plot, we can see that the forecast errors have constant variance over time. From the histogram, we can also assume that forecast errors are normally distributed with mean zero. Despite the curve being a bit wonky, it generally follows the normal distribution curve.

ARIMA MODEL

While Holt-Winters exponential smoothing provides an adequate predictive model, we again wanted to make a better model by taking correlations in the data into account. We wanted to try making an ARIMA (Autoregressive Integrated Moving Average) model for maximum average temperatures as well.

Based on the tutorial and some trial-and-error in our model building, we again noticed our data needed to be differenced prior to determing what ARIMA parameters would best fit our model.

Once forcing a manual difference, we can use the auto.arima function to figure out the parameters of our ARIMA model.

```
maxtemp_arima <- auto.arima(
  maxtemp_timeseries,
  d = 1,              # force one regular difference
  D = 1,              # seasonal difference (12-month)
  max.p = 5, max.q = 5,
  max.P = 2, max.Q = 2,
  stepwise = FALSE, approx = FALSE,
  trace = TRUE
)
```

```
##
##  ARIMA(0,1,0)(0,1,0)[12]                    : 1811.063
```

```
##  ARIMA(0,1,0)(0,1,1)[12]                    : Inf
##  ARIMA(0,1,0)(0,1,2)[12]                    : Inf
##  ARIMA(0,1,0)(1,1,0)[12]                    : 1761.423
##  ARIMA(0,1,0)(1,1,1)[12]                    : Inf
##  ARIMA(0,1,0)(1,1,2)[12]                    : Inf
##  ARIMA(0,1,0)(2,1,0)[12]                    : 1724.567
##  ARIMA(0,1,0)(2,1,1)[12]                    : 1657.14
##  ARIMA(0,1,0)(2,1,2)[12]                    : 1652.784
##  ARIMA(0,1,1)(0,1,0)[12]                    : 1683.043
##  ARIMA(0,1,1)(0,1,1)[12]                    : Inf
##  ARIMA(0,1,1)(0,1,2)[12]                    : Inf
##  ARIMA(0,1,1)(1,1,0)[12]                    : 1624.995
##  ARIMA(0,1,1)(1,1,1)[12]                    : Inf
##  ARIMA(0,1,1)(1,1,2)[12]                    : Inf
##  ARIMA(0,1,1)(2,1,0)[12]                    : 1592.023
##  ARIMA(0,1,1)(2,1,1)[12]                    : Inf
##  ARIMA(0,1,1)(2,1,2)[12]                    : 1525.06
##  ARIMA(0,1,2)(0,1,0)[12]                    : Inf
##  ARIMA(0,1,2)(0,1,1)[12]                    : Inf
##  ARIMA(0,1,2)(0,1,2)[12]                    : Inf
##  ARIMA(0,1,2)(1,1,0)[12]                    : 1626.913
##  ARIMA(0,1,2)(1,1,1)[12]                    : Inf
##  ARIMA(0,1,2)(1,1,2)[12]                    : Inf
##  ARIMA(0,1,2)(2,1,0)[12]                    : 1594.074
##  ARIMA(0,1,2)(2,1,1)[12]                    : Inf
##  ARIMA(0,1,3)(0,1,0)[12]                    : Inf
##  ARIMA(0,1,3)(0,1,1)[12]                    : Inf
##  ARIMA(0,1,3)(0,1,2)[12]                    : Inf
##  ARIMA(0,1,3)(1,1,0)[12]                    : 1627.682
##  ARIMA(0,1,3)(1,1,1)[12]                    : Inf
##  ARIMA(0,1,3)(2,1,0)[12]                    : 1594.704
##  ARIMA(0,1,4)(0,1,0)[12]                    : Inf
##  ARIMA(0,1,4)(0,1,1)[12]                    : Inf
##  ARIMA(0,1,4)(1,1,0)[12]                    : 1629.772
##  ARIMA(0,1,5)(0,1,0)[12]                    : Inf
##  ARIMA(1,1,0)(0,1,0)[12]                    : 1748.582
##  ARIMA(1,1,0)(0,1,1)[12]                    : Inf
##  ARIMA(1,1,0)(0,1,2)[12]                    : Inf
##  ARIMA(1,1,0)(1,1,0)[12]                    : 1701.23
##  ARIMA(1,1,0)(1,1,1)[12]                    : Inf
##  ARIMA(1,1,0)(1,1,2)[12]                    : Inf
##  ARIMA(1,1,0)(2,1,0)[12]                    : 1662.546
##  ARIMA(1,1,0)(2,1,1)[12]                    : Inf
##  ARIMA(1,1,0)(2,1,2)[12]                    : 1593.679
##  ARIMA(1,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(1,1,1)(0,1,1)[12]                    : Inf
##  ARIMA(1,1,1)(0,1,2)[12]                    : Inf
##  ARIMA(1,1,1)(1,1,0)[12]                    : 1626.938
##  ARIMA(1,1,1)(1,1,1)[12]                    : Inf
##  ARIMA(1,1,1)(1,1,2)[12]                    : Inf
##  ARIMA(1,1,1)(2,1,0)[12]                    : Inf
##  ARIMA(1,1,1)(2,1,1)[12]                    : Inf
##  ARIMA(1,1,2)(0,1,0)[12]                    : Inf
##  ARIMA(1,1,2)(0,1,1)[12]                    : Inf
```

```
##  ARIMA(1,1,2)(0,1,2)[12]                    : Inf
##  ARIMA(1,1,2)(1,1,0)[12]                    : Inf
##  ARIMA(1,1,2)(1,1,1)[12]                    : Inf
##  ARIMA(1,1,2)(2,1,0)[12]                    : Inf
##  ARIMA(1,1,3)(0,1,0)[12]                    : Inf
##  ARIMA(1,1,3)(0,1,1)[12]                    : Inf
##  ARIMA(1,1,3)(1,1,0)[12]                    : Inf
##  ARIMA(1,1,4)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,0)(0,1,0)[12]                    : 1716.751
##  ARIMA(2,1,0)(0,1,1)[12]                    : Inf
##  ARIMA(2,1,0)(0,1,2)[12]                    : Inf
##  ARIMA(2,1,0)(1,1,0)[12]                    : 1664.955
##  ARIMA(2,1,0)(1,1,1)[12]                    : Inf
##  ARIMA(2,1,0)(1,1,2)[12]                    : Inf
##  ARIMA(2,1,0)(2,1,0)[12]                    : 1621.051
##  ARIMA(2,1,0)(2,1,1)[12]                    : Inf
##  ARIMA(2,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,1)(0,1,1)[12]                    : Inf
##  ARIMA(2,1,1)(0,1,2)[12]                    : Inf
##  ARIMA(2,1,1)(1,1,0)[12]                    : 1627.423
##  ARIMA(2,1,1)(1,1,1)[12]                    : Inf
##  ARIMA(2,1,1)(2,1,0)[12]                    : 1594.292
##  ARIMA(2,1,2)(0,1,0)[12]                    : Inf
##  ARIMA(2,1,2)(0,1,1)[12]                    : Inf
##  ARIMA(2,1,2)(1,1,0)[12]                    : 1629.51
##  ARIMA(2,1,3)(0,1,0)[12]                    : Inf
##  ARIMA(3,1,0)(0,1,0)[12]                    : 1699.997
##  ARIMA(3,1,0)(0,1,1)[12]                    : Inf
##  ARIMA(3,1,0)(0,1,2)[12]                    : Inf
##  ARIMA(3,1,0)(1,1,0)[12]                    : 1643.218
##  ARIMA(3,1,0)(1,1,1)[12]                    : Inf
##  ARIMA(3,1,0)(2,1,0)[12]                    : 1605.336
##  ARIMA(3,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(3,1,1)(0,1,1)[12]                    : Inf
##  ARIMA(3,1,1)(1,1,0)[12]                    : 1629.5
##  ARIMA(3,1,2)(0,1,0)[12]                    : Inf
##  ARIMA(4,1,0)(0,1,0)[12]                    : 1697.362
##  ARIMA(4,1,0)(0,1,1)[12]                    : Inf
##  ARIMA(4,1,0)(1,1,0)[12]                    : 1640.132
##  ARIMA(4,1,1)(0,1,0)[12]                    : Inf
##  ARIMA(5,1,0)(0,1,0)[12]                    : 1697.758
##
##
##
##  Best model: ARIMA(0,1,1)(2,1,2)[12]
```

```
maxtemp_arima
```

```
## Series: maxtemp_timeseries
## ARIMA(0,1,1)(2,1,2)[12]
##
## Coefficients:
##           ma1   sar1     sar2     sma1    sma2
##       -0.8468  0.375  -0.1598  -1.3755  0.4570
```

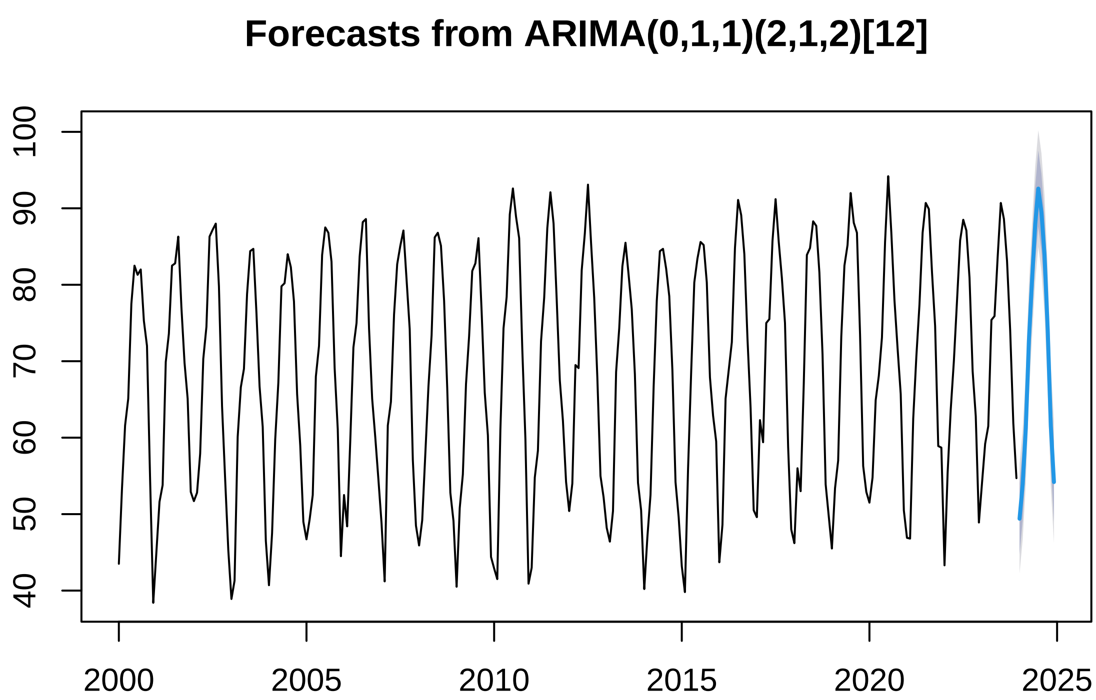
```
## s.e.   0.0430  0.215   0.0744   0.2065  0.2054
##
## sigma^2 = 13.28:  log likelihood = -756.37
## AIC=1524.75   AICc=1525.06   BIC=1546.45
```

Based on the auto.arima() function, we can determine that an ARIMA(0,1,1)x(2,1,2)[12] best fits our data. We can use this model to forecast for 2024. We want to check that this model meets all assumptions.

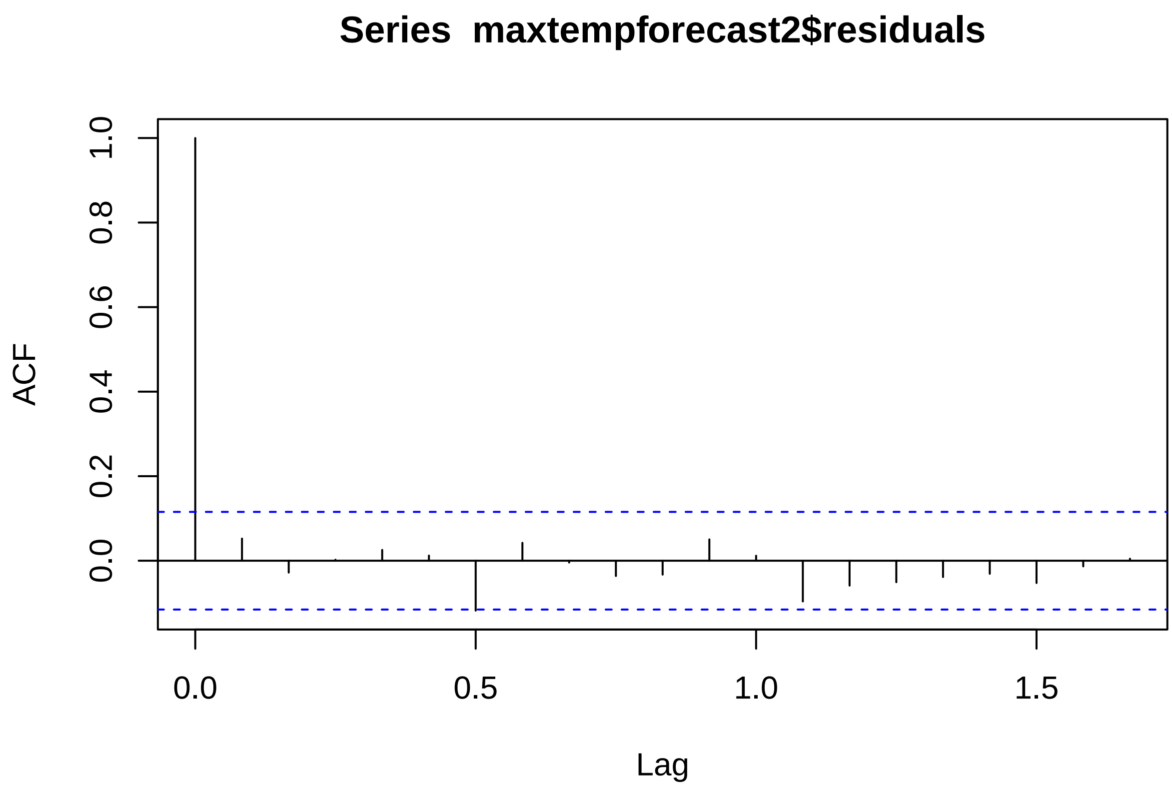
```
maxtempforecast2 <- forecast(maxtemp_arima, h=12) # forecast using ARIMA model
maxtempforecast2
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95     Hi 95
## Jan 2024       49.40025 44.73044 54.07006 42.25839  56.54211
## Feb 2024       53.90951 49.18524 58.63379 46.68436  61.13467
## Mar 2024       61.29836 56.52025 66.07648 53.99086  68.60586
## Apr 2024       72.74837 67.91701 77.57973 65.35944  80.13730
## May 2024       80.60167 75.71765 85.48569 73.13220  88.07114
## Jun 2024       87.89314 82.95702 92.82926 80.34399  95.44229
## Jul 2024       92.58003 87.59235 97.56771 84.95203 100.20803
## Aug 2024       89.47833 84.43962 94.51704 81.77229  97.18438
## Sep 2024       83.84218 78.75296 88.93141 76.05888  91.62549
## Oct 2024       74.10981 68.97056 79.24906 66.25001  81.96961
## Nov 2024       61.50452 56.31573 66.69331 53.56895  69.44008
## Dec 2024       54.19398 48.95612 59.43184 46.18337  62.20459
```

```
plot(maxtempforecast2) # add forecast values into time series plot
```



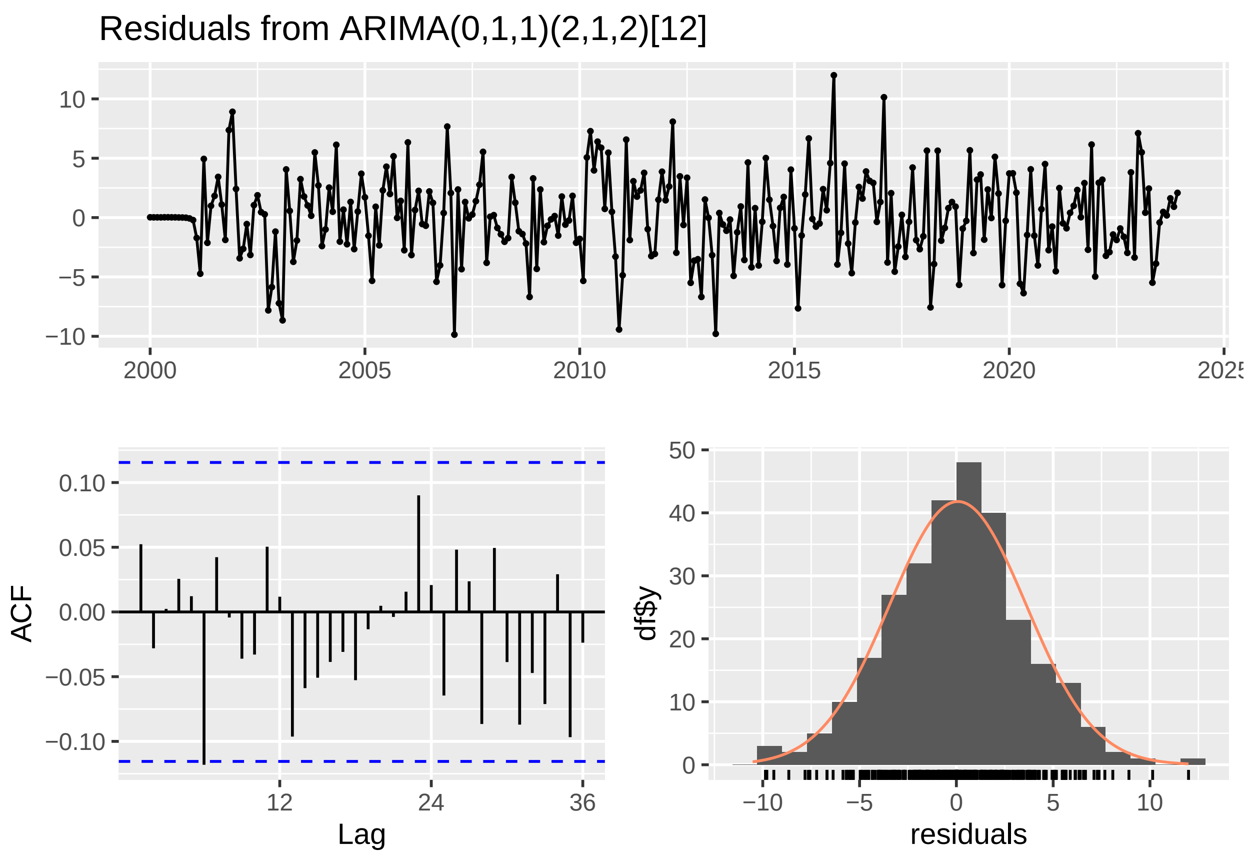
```r
acf(maxtempforecast2$residuals,lag.max=20) # correlogram
```




```r
Box.test(maxtempforecast2$residuals, lag=20, type="Ljung-Box") # Ljung-Box test
```

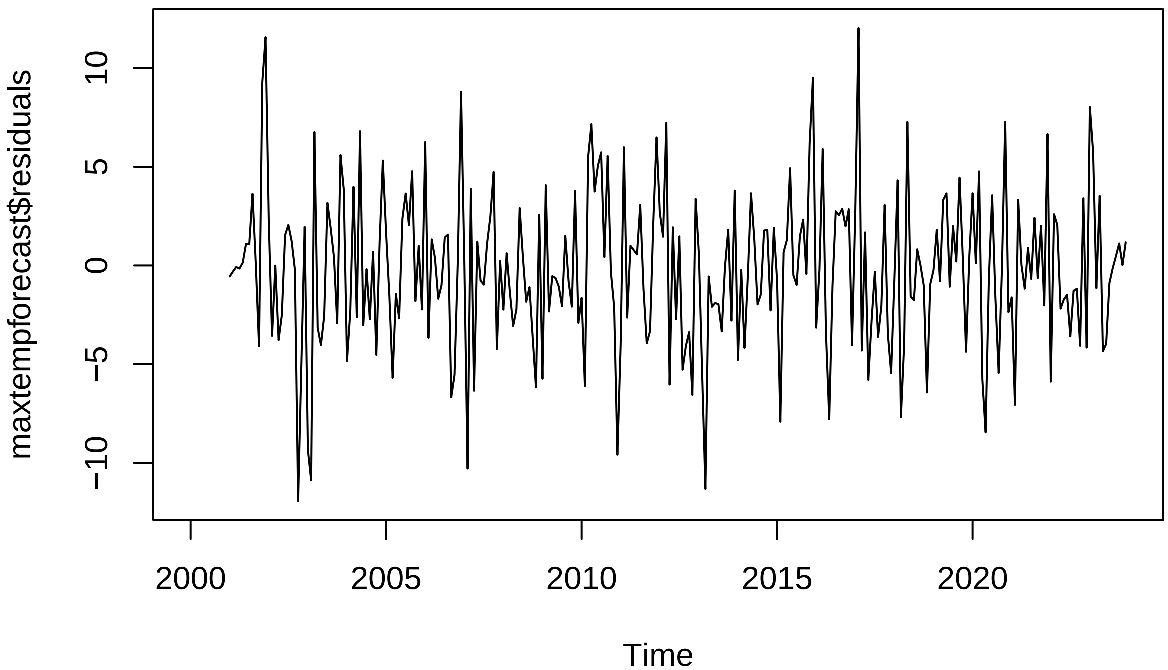
```
##
##  Box-Ljung test
##
## data:  maxtempforecast2$residuals
## X-squared = 13.791, df = 20, p-value = 0.8409
```

```r
checkresiduals(maxtemp_arima) # chceck that residuals are insignificant
```

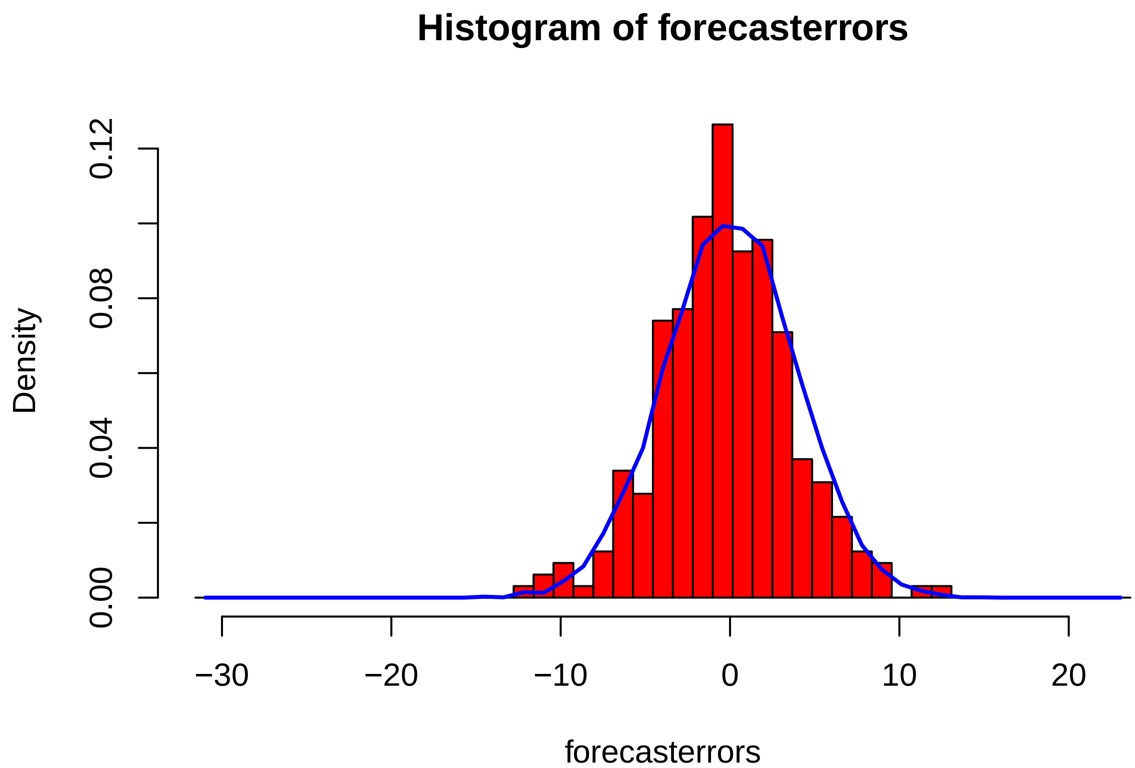



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)(2,1,2)[12]
## Q* = 16.568, df = 19, p-value = 0.6191
##
## Model df: 5.   Total lags used: 24
```

```
plot.ts(maxtempforecast$residuals)
```




```
plotForecastErrors(maxtempforecast$residuals) # normally distributed with mean 0
```



Based on our model and forecast values, we see that the correlogram shows that barely any of the sample autocorrelations for lags 1-20 exceed significance bounds, the p-value of the Ljung-Box test is 0.84, and the p-value of our residuals is 0.62. We can conclude there is little evidence for non-zero autocorrelations in the forecast errors.

From the time plot, we can see that the forecast errors have constant variance over time. From the histogram, we can also assume that forecast errors are normally distributed with mean zero.

Now, let's compare our forecast values from the ARIMA model to our actual values for 2024.

```
# extract 2024 from our data
maxtemp_2024 <- maxtemp[maxtemp$Year==2024, ]
maxtemp_2024
```

```
##     Month_Year Year Month Value
## 25    Jan_2024 2024   Jan  48.5
## 51    Feb_2024 2024   Feb  56.2
## 77    Mar_2024 2024   Mar  64.7
## 103   Apr_2024 2024   Apr  72.4
## 129   May_2024 2024   May  79.4
## 155   Jun_2024 2024   Jun  89.2
## 181   Jul_2024 2024   Jul  89.3
## 207   Aug_2024 2024   Aug  86.2
## 233   Sep_2024 2024   Sep  76.2
## 259   Oct_2024 2024   Oct  71.9
## 285   Nov_2024 2024   Nov  61.8
## 311   Dec_2024 2024   Dec  46.6
```

```r
maxtempforecast2
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95     Hi 95
## Jan 2024       49.40025 44.73044 54.07006 42.25839  56.54211
## Feb 2024       53.90951 49.18524 58.63379 46.68436  61.13467
## Mar 2024       61.29836 56.52025 66.07648 53.99086  68.60586
## Apr 2024       72.74837 67.91701 77.57973 65.35944  80.13730
## May 2024       80.60167 75.71765 85.48569 73.13220  88.07114
## Jun 2024       87.89314 82.95702 92.82926 80.34399  95.44229
## Jul 2024       92.58003 87.59235 97.56771 84.95203 100.20803
## Aug 2024       89.47833 84.43962 94.51704 81.77229  97.18438
## Sep 2024       83.84218 78.75296 88.93141 76.05888  91.62549
## Oct 2024       74.10981 68.97056 79.24906 66.25001  81.96961
## Nov 2024       61.50452 56.31573 66.69331 53.56895  69.44008
## Dec 2024       54.19398 48.95612 59.43184 46.18337  62.20459
```

By comparing our forecast values of 2024 to our actual values of 2024, we again can see that all the actual data falls in the 95% prediction interval for the forecast. Thus, we meet our goal of being at least 80% accurate with our model.

Using ARIMA models with differencing and a seasonality component allowed us to build accurate models which generated prediction intervals that managed to capture our actual data points.