Assignment 5: Beer Sales

Joshua Goldberg

May, 09 2019

Load data from TSA package (the package is written by authors Jonathan Cryer and Kung-Sik Chan).

```
data(beersales)
```

The data is the monthly beer sales in millions of barrels, 01/1975 - 12/1990.

Train: 01/1975 - 12/1989.

Test: 1990

Part 1

Use `ARIMA(p,d,q)` model to forecast beer sales for all months of 1990 using the following two multi-step forecasting approaches. For each model, check mean, autocorrelation and normality of the residuals. Confirm if the residuals are white noise.

```
train <- window(beersales, c(1975, 1), c(1989, 12))
test <- window(beersales, c(1990, 1), c(1990, 12))
```

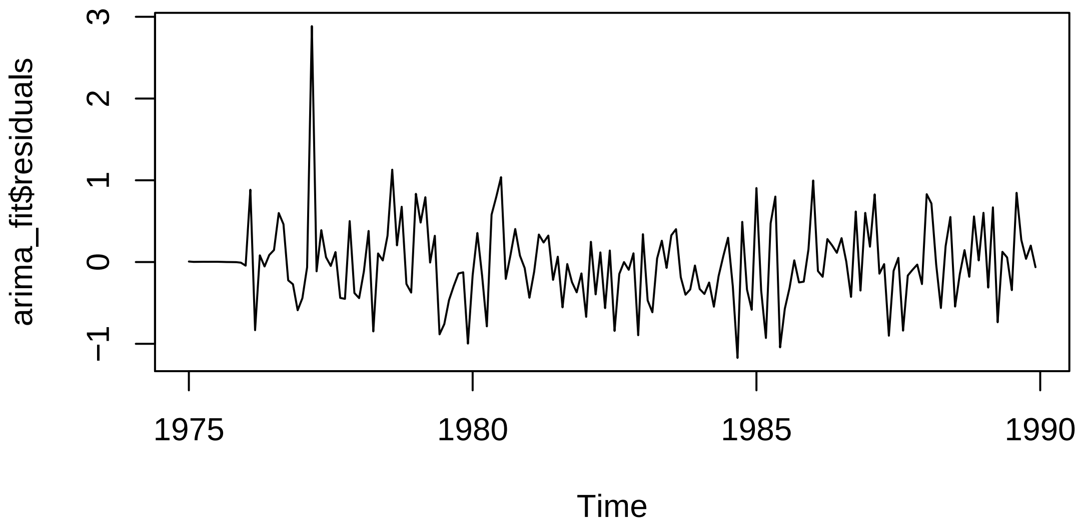
```
arima_fit <- auto.arima(train)
summary(arima_fit)
```

```
## Series: train
## ARIMA(4,1,2)(2,1,2)[12]
##
## Coefficients:
##          ar1      ar2     ar3      ar4      ma1     ma2    sar1    sar2
##       0.5103  -0.1662  0.1032  -0.3966  -1.1757  0.3125  0.6838  -0.592
## s.e.  0.1453   0.0986  0.0863   0.0789   0.1493  0.1421  0.1451   0.165
##          sma1    sma2
##       -1.1967  0.5849
## s.e.   0.1394  0.2087
##
## sigma^2 estimated as 0.2837:  log likelihood=-134.55
## AIC=291.1   AICc=292.81   BIC=325.4
##
## Training set error measures:
##                       ME      RMSE       MAE        MPE     MAPE      MASE
## Training set -0.01226384 0.4974721 0.3570516 -0.1687359 2.535728 0.6855892
##                     ACF1
## Training set 0.001560233
```

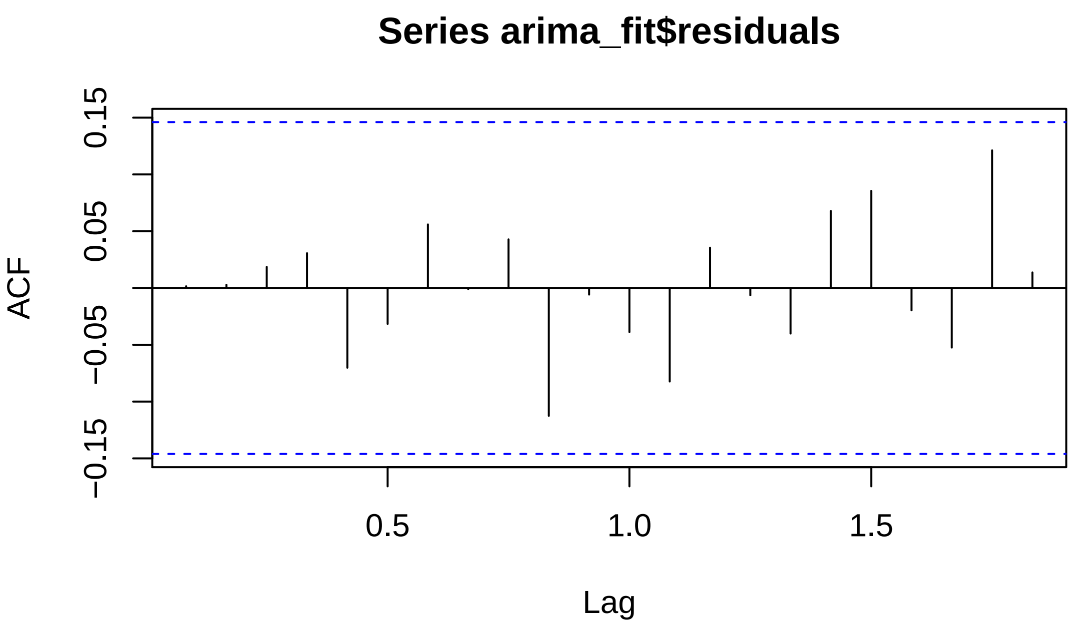
1A

Use the h-period in `forecast()` to forecast each month of 1990. This is also known as recursive forecasting where you fit a model only once and use it recursively for h-periods.

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




```r
acf(arima_fit$residuals)
```




```r
(recursive_forecast <- forecast(arima_fit, h = 12))
```

```
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 1990       13.81601 13.13331 14.49871 12.77191 14.86011
## Feb 1990       13.07707 12.35715 13.79698 11.97605 14.17808
## Mar 1990       14.96181 14.23546 15.68817 13.85095 16.07268
## Apr 1990       15.58503 14.83785 16.33220 14.44232 16.72774
## May 1990       17.24847 16.49698 17.99996 16.09917 18.39777
## Jun 1990       16.86360 16.10993 17.61727 15.71096 18.01624
## Jul 1990       16.95571 16.19987 17.71156 15.79974 18.11168
## Aug 1990       17.02231 16.26451 17.78012 15.86336 18.18127
## Sep 1990       14.28619 13.51600 15.05638 13.10828 15.46410
## Oct 1990       14.55136 13.75967 15.34305 13.34057 15.76214
## Nov 1990       12.89695 12.09174 13.70216 11.66548 14.12841
```

```
## Dec 1990       12.30127 11.48554 13.11699 11.05372 13.54881
```

1B

Use the monthly data as a continuous time series. Forecast for 1990 Jan, Plug forecast into the time series, build a new model to forecast for 1990 Feb. And so on and so forth. In other words, h=1 in all the forecasts. This is known as direct recursive (DirRec) forecasting where you fit a new model for each time step.

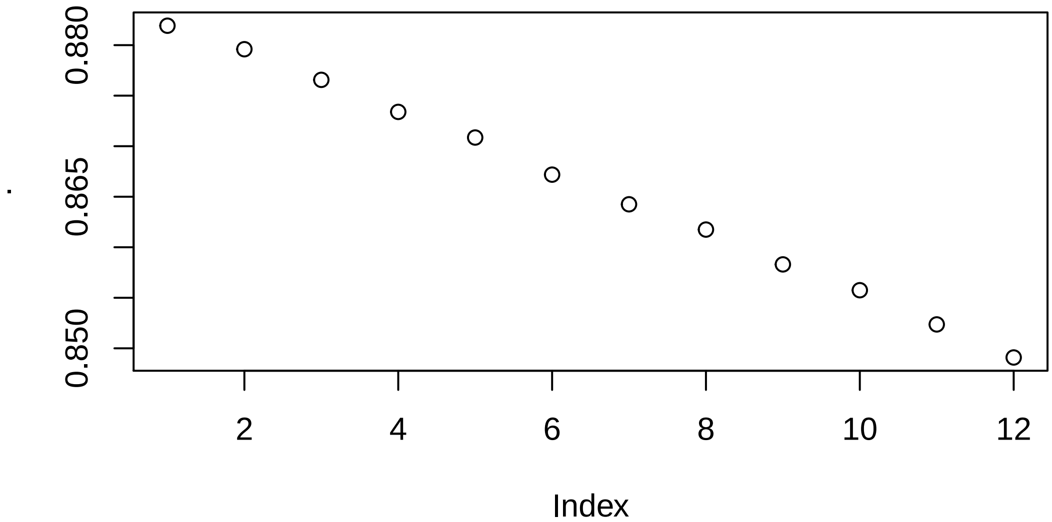
```
dir_rec <- function(.data, .model_count, refit = FALSE) {
  predictions <- vector("numeric", .model_count)
  models <- vector("list", .model_count)

  new_data <- train %>% as_tsibble(index = index) %>% append_row(.model_count)
  model_fit <- auto.arima(new_data %>% drop_na() %>% as_tsibble(index = index) %>% as.ts())
  orders <- arimaorder(model_fit)

  # Index used for appending new data
  index_change <- .model_count - 1

    for (i in 1:.model_count) {

      model_data <- new_data %>% drop_na() %>% as_tsibble(index = index) %>% as.ts()

      model <- Arima(model_data,
                     order = orders[c("p", "d", "q")],
                     seasonal = orders[c("P", "D", "Q")])

      forecast_point <- forecast(model, h = 1)$mean %>% as.numeric()

      predictions[[i]] <- forecast_point
      new_data[nrow(new_data) - index_change, "value"] <- forecast_point
      models[[i]] <- model

      # Reduce index by 1 floored to zero if negative
      index_change <- ifelse(index_change - 1 < 0, 0, index_change - 1)

      # Refit `auto.arima()` to obtain new parameters with prediction in consideration
      if (refit)  {
        model_fit <- auto.arima(model_data, stepwise = FALSE)
        orders <- arimaorder(model_fit)
      }
    }

  list(data = new_data, models = models, predictions = predictions)
}

model_same_params <- dir_rec(train, 12, refit = FALSE)
model_refit <- dir_rec(train, 12, refit = TRUE)
```

1C

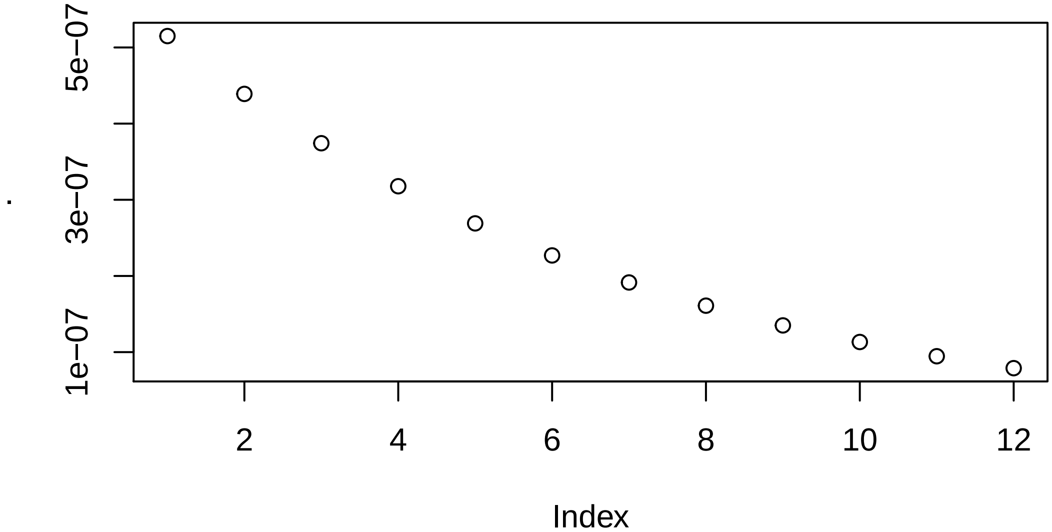
Plot the mean, the p-value of the autocorrelation test and the p-value of the normality test of the residuals of the 12 models. The Box test results fail to reject the null hypothesis (the data are independently distributed).

The data visually do not look too bad, but we reject the null hypothesis (data is normally distributed) of the shapiro test.

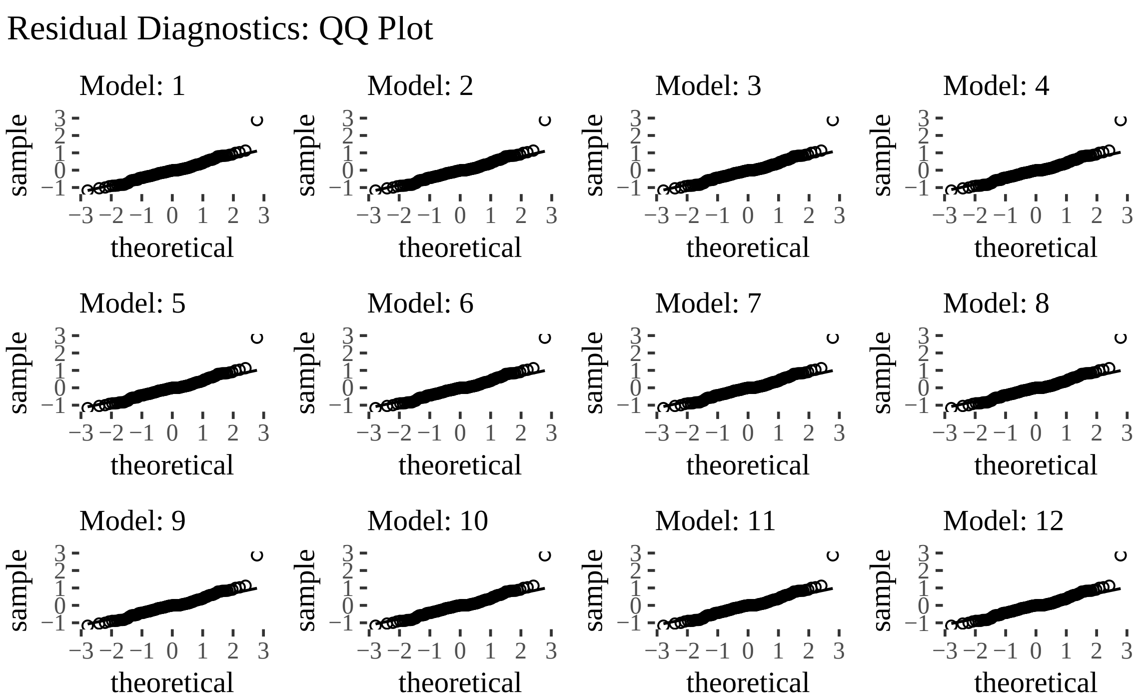
```
map_dbl(model_same_params$models, ~ Box.test(.x$residuals, lag = 24)$p.value) %>% plot()
```



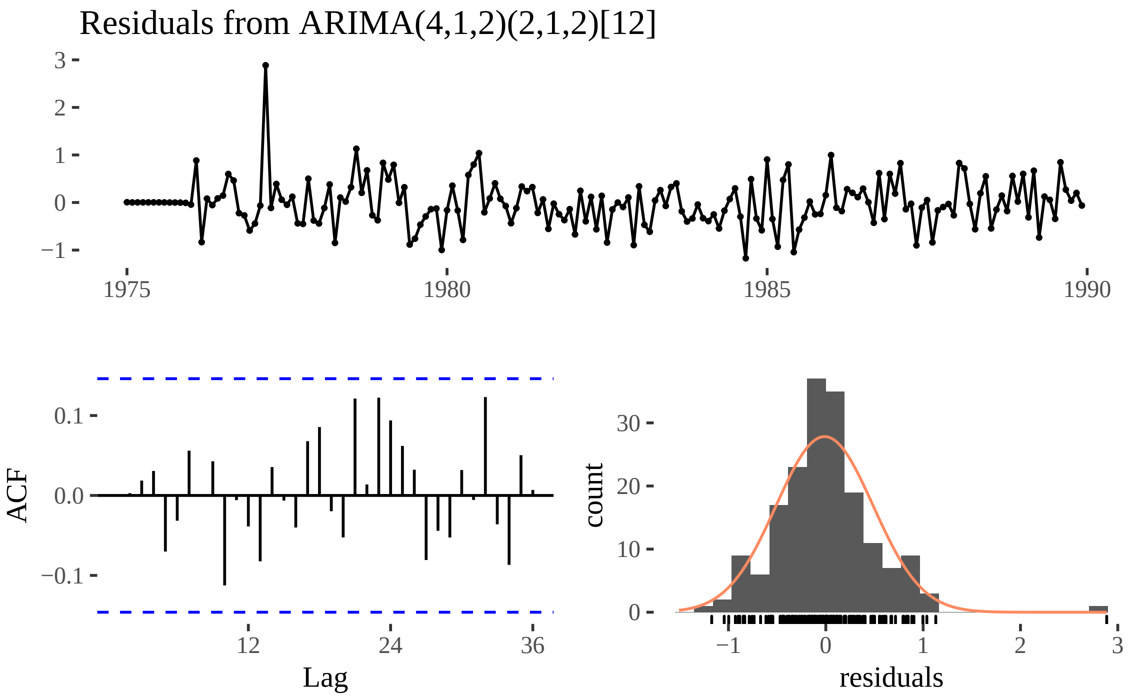

```
map_dbl(model_same_params$models, ~ shapiro.test(.x$residuals)$p.value) %>% plot()
```



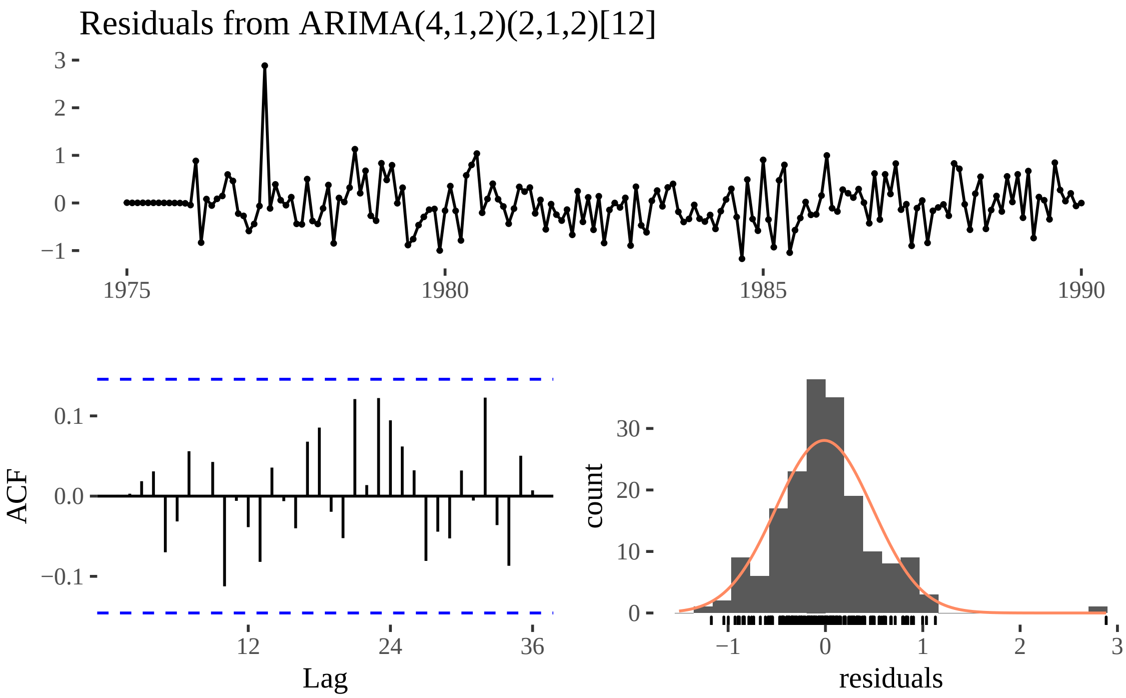

```
map(1:12, ~
data.frame(r = model_same_params$models[[.x]]$residuals %>% as.numeric()) %>%
ggplot(aes(sample = r)) +
stat_qq(pch = 1) +
stat_qq_line() +
labs(subtitle = paste0("Model: ", .x))) %>%
patchwork::wrap_plots() +
patchwork::plot_annotation(title = "Residual Diagnostics: QQ Plot")
```



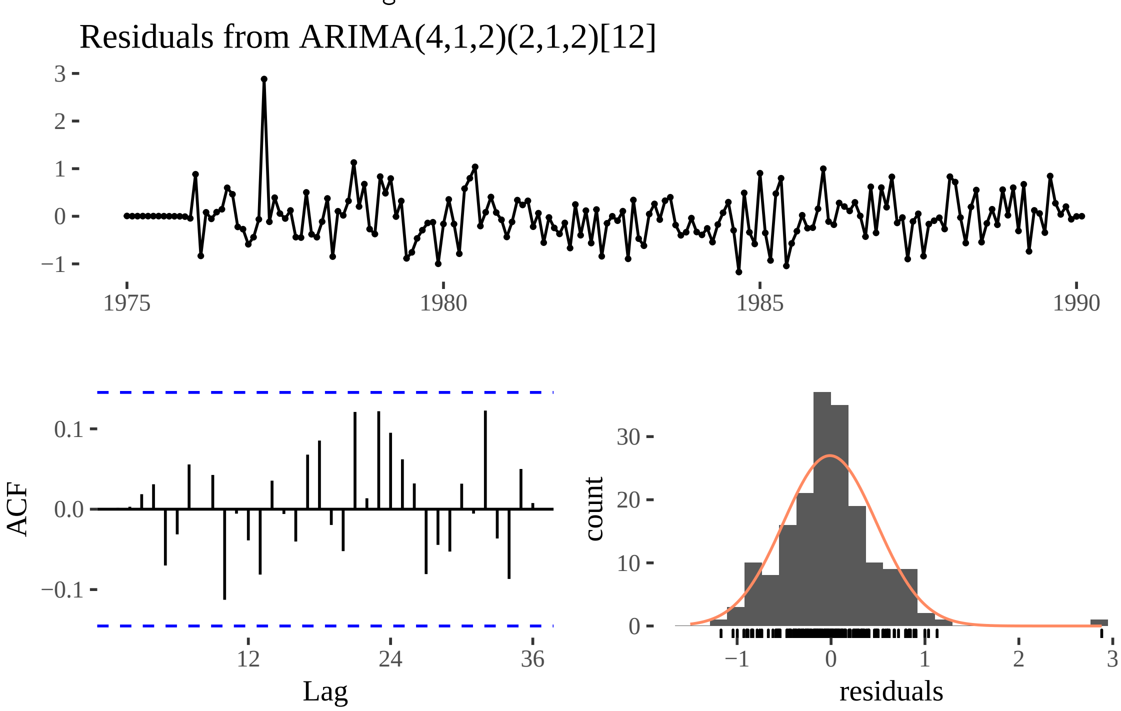

```r
walk(model_same_params$models, ~ checkresiduals(.x))
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.035, df = 14, p-value = 0.2052
##
## Model df: 10.   Total lags used: 24
```

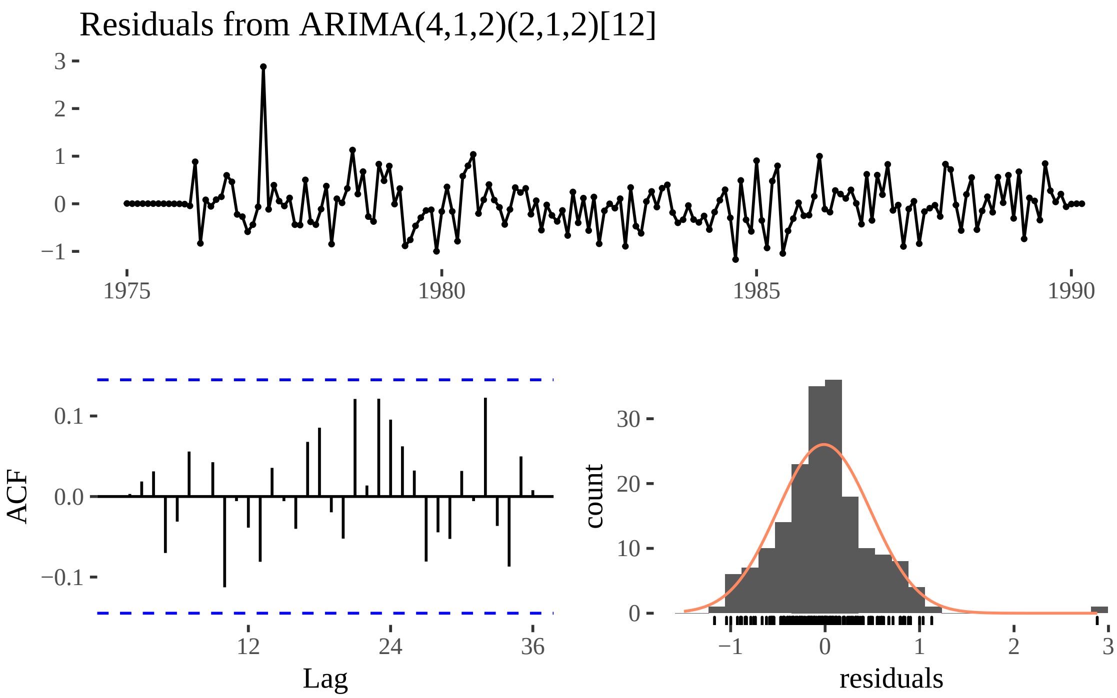



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.095, df = 14, p-value = 0.2025
##
## Model df: 10.   Total lags used: 24
```

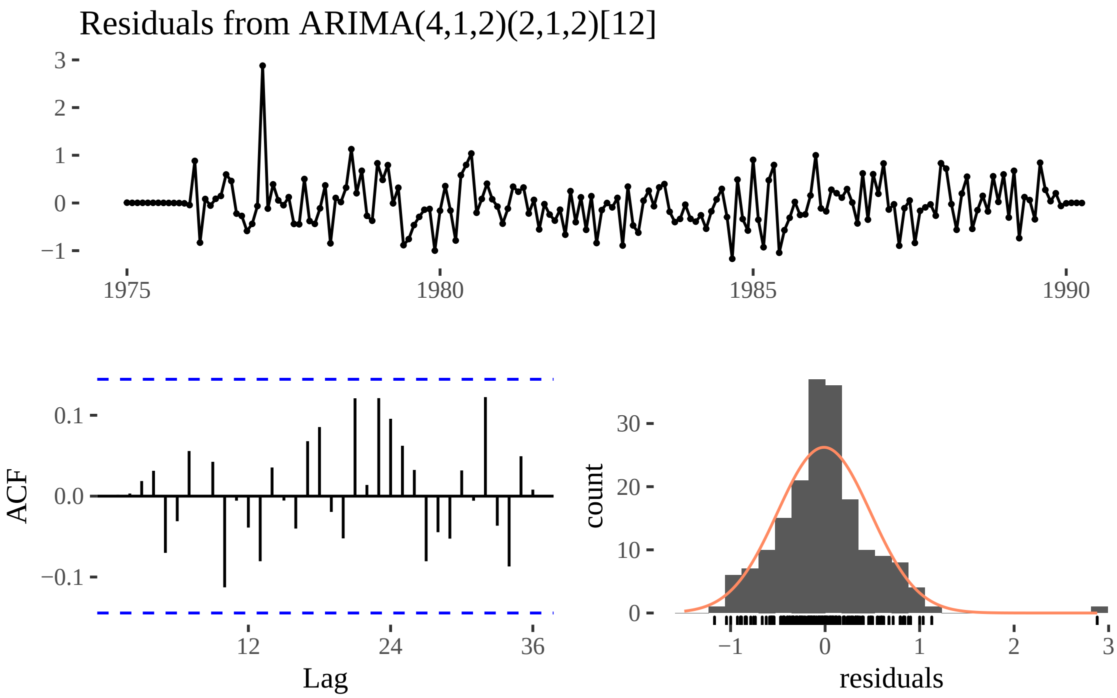



```
##
##  Ljung-Box test
```

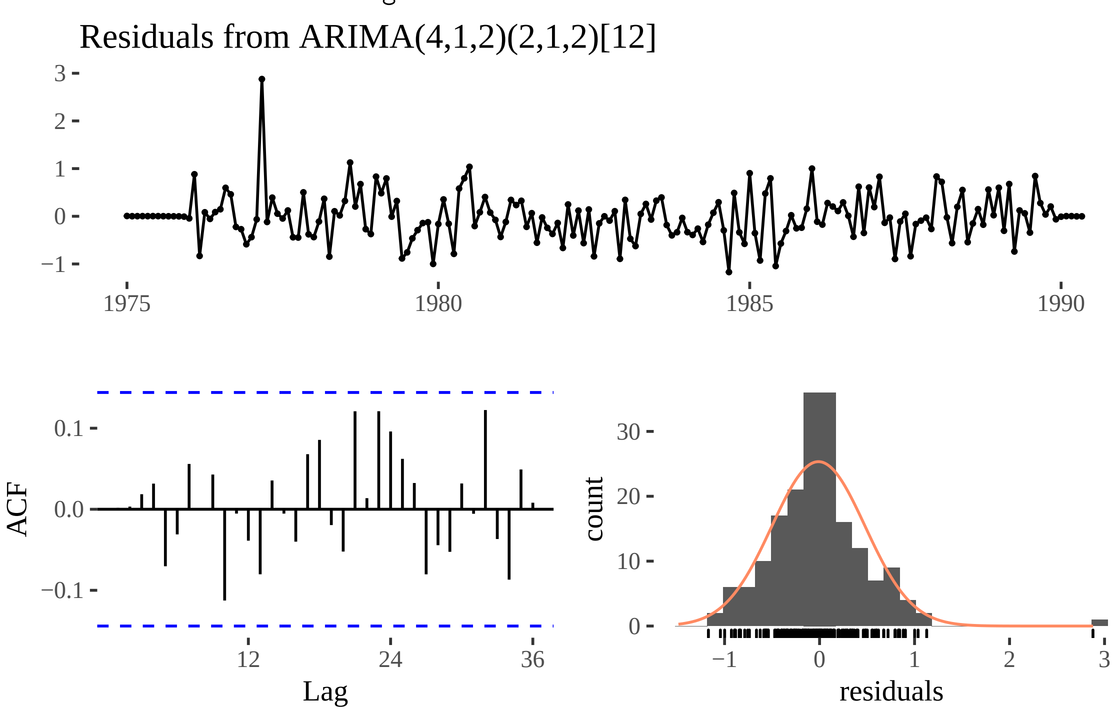
```
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.173, df = 14, p-value = 0.199
##
## Model df: 10.   Total lags used: 24
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.253, df = 14, p-value = 0.1955
##
## Model df: 10.   Total lags used: 24
```

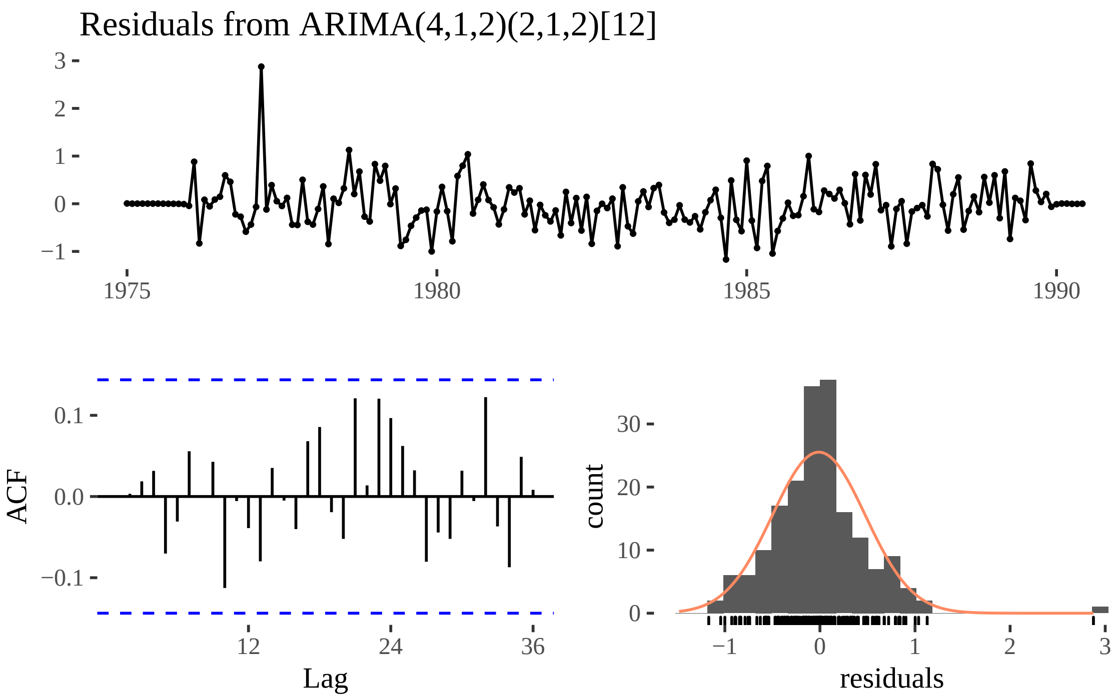



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.315, df = 14, p-value = 0.1928
##
## Model df: 10.   Total lags used: 24
```

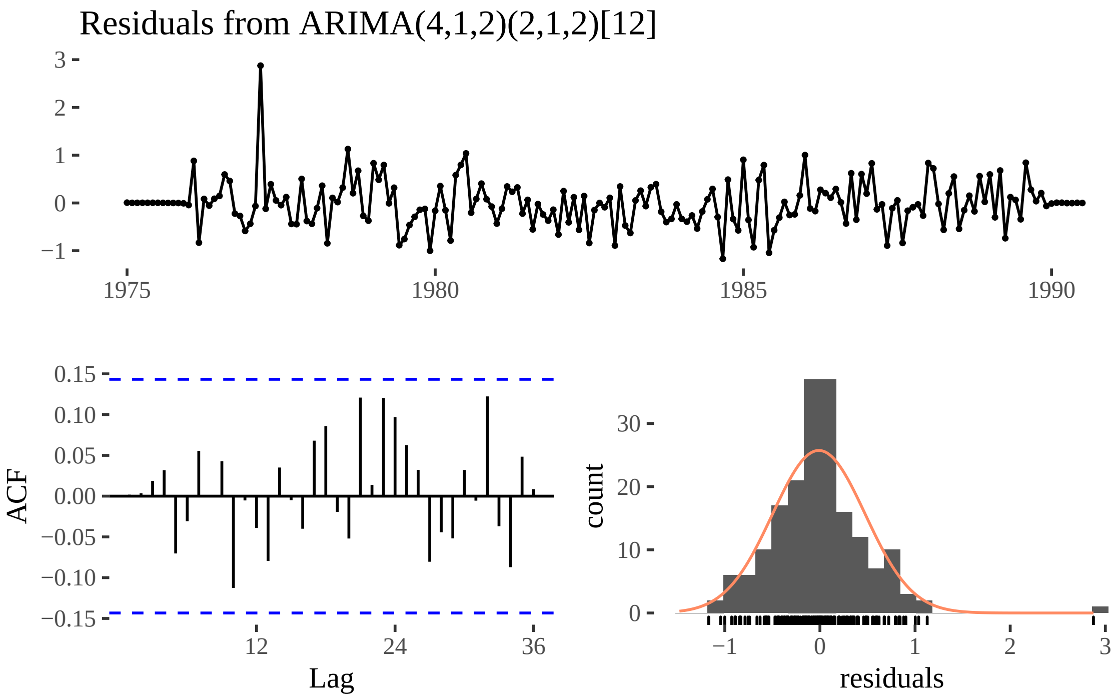



```
##
##  Ljung-Box test
```

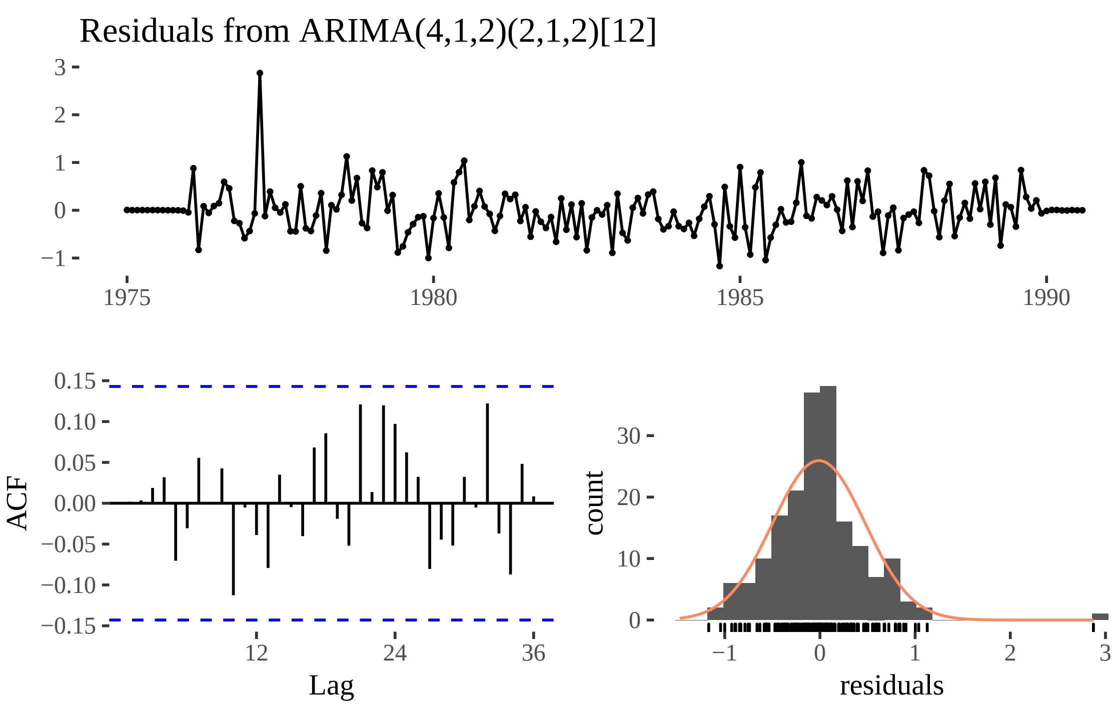
```
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.406, df = 14, p-value = 0.1889
##
## Model df: 10.   Total lags used: 24
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.478, df = 14, p-value = 0.1859
##
## Model df: 10.   Total lags used: 24
```

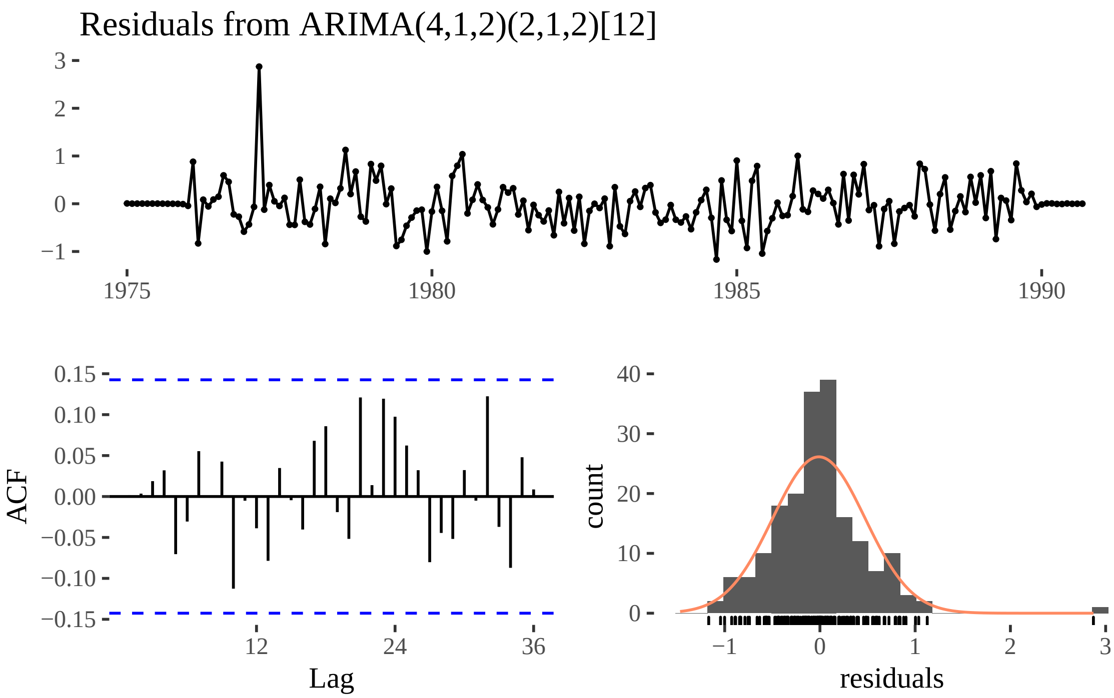



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.535, df = 14, p-value = 0.1835
##
## Model df: 10.   Total lags used: 24
```

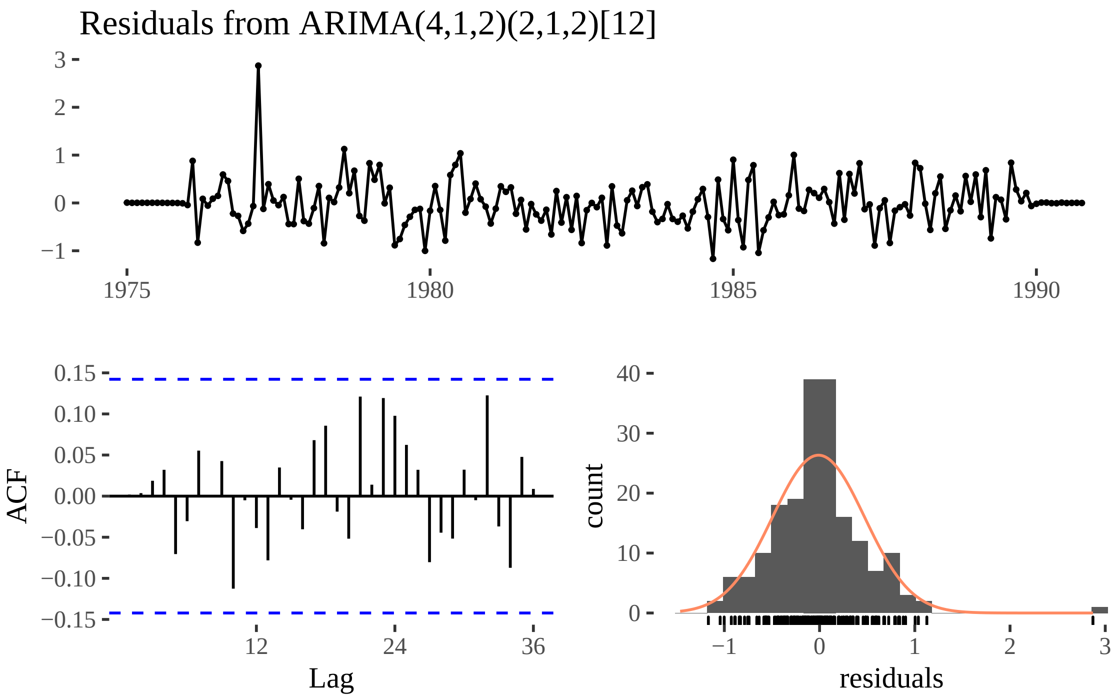



```
##
##  Ljung-Box test
```

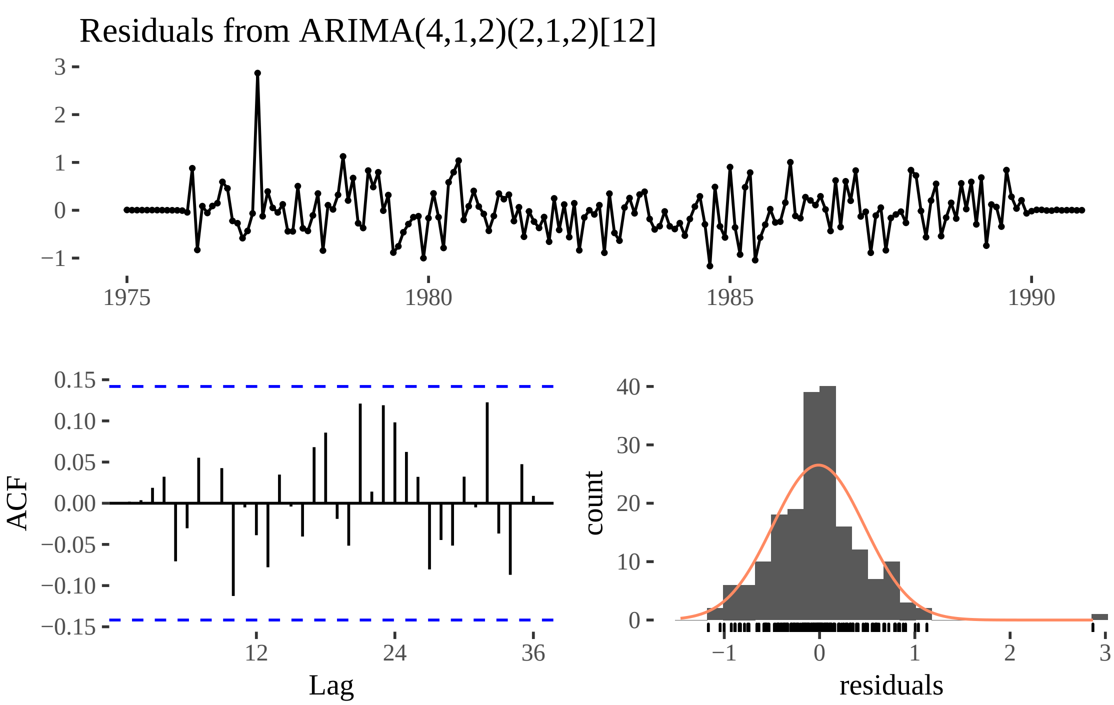
```
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.618, df = 14, p-value = 0.1801
##
## Model df: 10.   Total lags used: 24
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.676, df = 14, p-value = 0.1777
##
## Model df: 10.   Total lags used: 24
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.756, df = 14, p-value = 0.1745
##
## Model df: 10.   Total lags used: 24
```



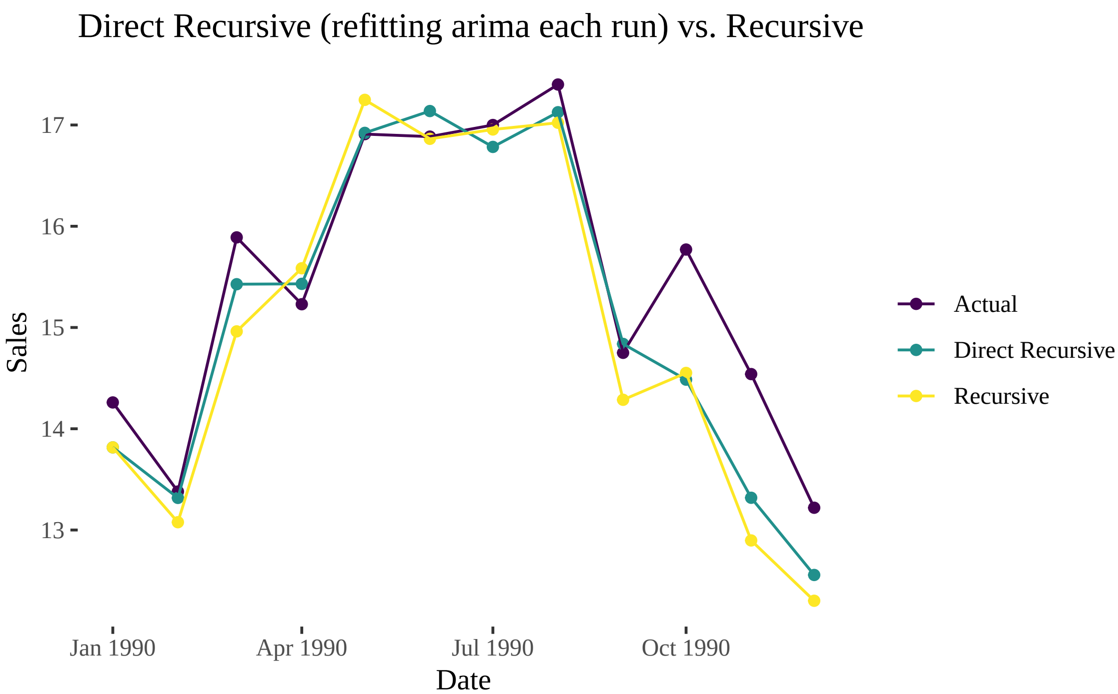

```
##
##  Ljung-Box test
```

```
##
## data:  Residuals from ARIMA(4,1,2)(2,1,2)[12]
## Q* = 18.831, df = 14, p-value = 0.1715
##
## Model df: 10.   Total lags used: 24
```

Part 2

Plot the Recursive and DirRec along with the actuals. Use ylim=c(12.5, 17) to get a good visual of the plot differences.

```
test_predictions <- test %>%
  as_tsibble() %>%
  mutate(direct_recursive = model_refit$predictions,
         recursive_model = recursive_forecast %>% as_tibble() %>% pull(`Point Forecast`))

test_predictions %>%
  rename(actual = value) %>%
  gather(key = type, value = value) %>%
  ggplot(aes(index, value, color = type)) +
  geom_point() +
  geom_line() +
  scale_color_viridis_d(name = NULL, labels = c("Actual", "Direct Recursive", "Recursive")) +
  labs(title = "Direct Recursive (refitting arima each run) vs. Recursive",
       x = "Date",
       y = "Sales")
```



Part 3

Calculate the MSE for 1990 - which of the two approaches take larger computation time and why? Direct Recursive (DR) is better from . Computationally it is more intensive, since it has to refit the model with new data.

```
mse_recursive = mean((test_predictions$recursive_model - test_predictions$value)^2)
mse_direct_recursive = mean((test_predictions$direct_recursive - test_predictions$value)^2)

glue::glue("MSE recursive: {mse_recursive}")
```

```
## MSE recursive: 0.565002004429162
```

```
glue::glue("MSE direct recursive: {mse_direct_recursive}")
```

```
## MSE direct recursive: 0.352798601820935
```