Souvenir Sales Time Series Analysis

Souvenir Sales - Time Series Analysis

The following is a statistical analysis on a monthly time series which collects data about the sales of a souvenir shop in Australia in the period between 1987 and 1992.

The analysis will roughly follow the Box-Jenkins method and will focus on reaching stationarity for the time series, estimating a SARIMA model and predicting future values.

Data exploration

```r
library(tidyverse)
library(tsdl)
library(tseries)
library(tsibble)
library(feasts)
library(forecast)
library(lmtest)
library(ldsr)
```

```r
data <- subset(tsdl, 12, "sales", description = "Queensland")[[1]]
attributes(data)
```

```
## $tsp
## [1] 1987.000 1993.917   12.000
##
## $class
## [1] "ts"
##
## $source
## [1] "Makridakis, Wheelwright and Hyndman (1998)"
##
## $description
## [1] "Monthly sales for a souvenir shop on the wharf at a beach resort town in Queensland, Australia.
##
## $subject
## [1] "Sales"
```

Now that we have imported the time series, let's have a first look at its values.

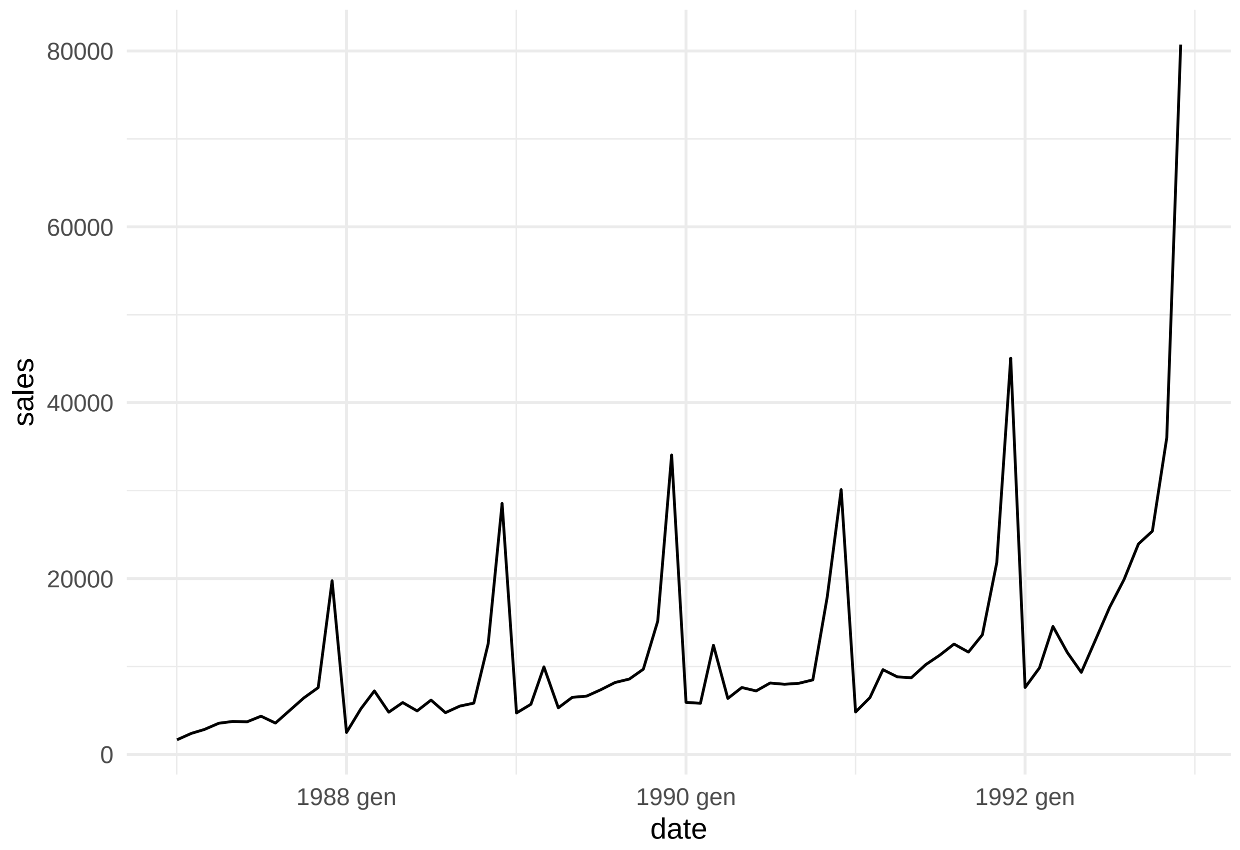
```r
tseries <- data %>% as_tsibble() %>% head(n = 72) %>%
  rename(date = index, sales = value)
tseries
```

```
## # A tsibble: 72 x 2 [1M]
##         date sales
##        <mth> <dbl>
##  1 1987 gen 1665.
##  2 1987 feb 2398.
##  3 1987 mar 2841.
##  4 1987 apr 3547.
##  5 1987 mag 3753.
##  6 1987 giu 3715.
##  7 1987 lug 4350.
##  8 1987 ago 3566.
##  9 1987 set 5022.
## 10 1987 ott 6423.
## # i 62 more rows
```

Check for non-stationarity

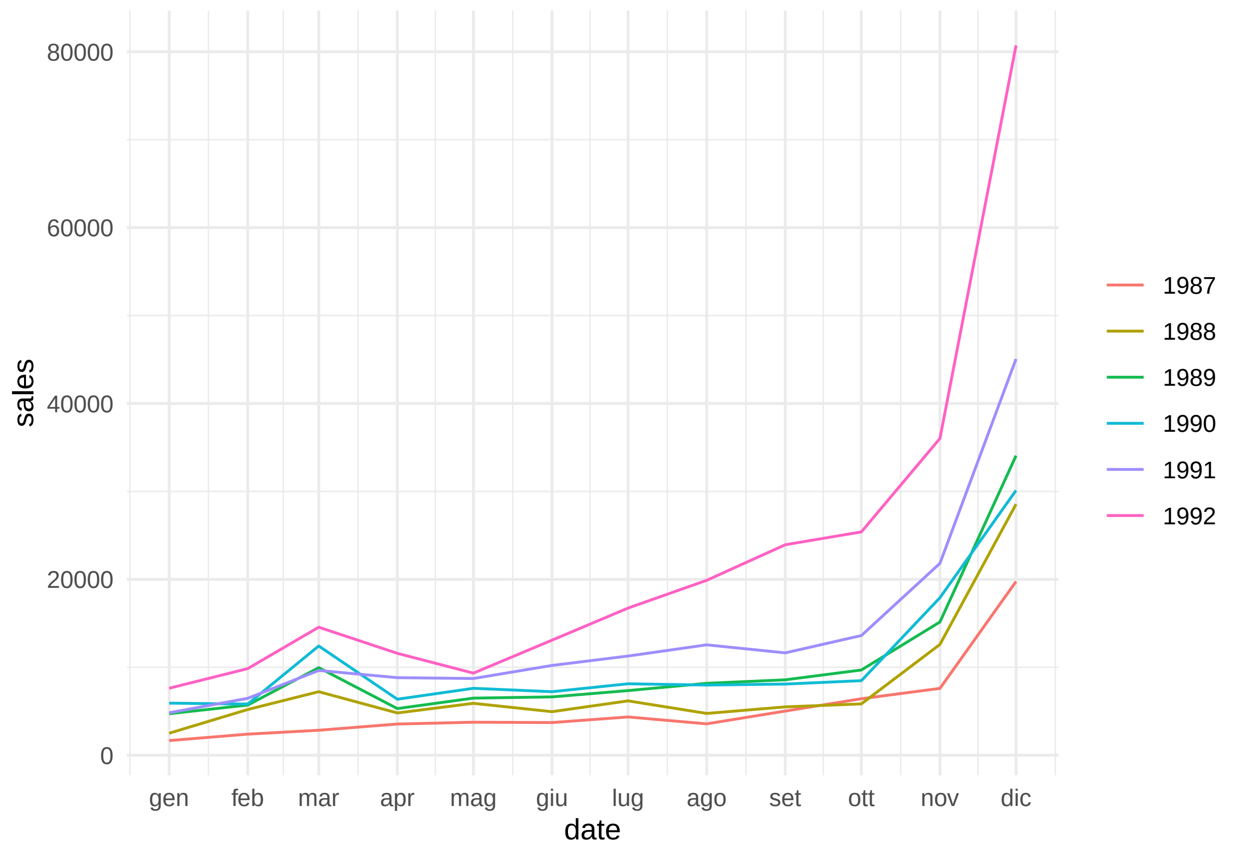
Let's make a plot and see what we can say about it.

```
tseries %>% ggplot(aes(x = date, y = sales)) +
  geom_line() +
  theme_minimal()
```



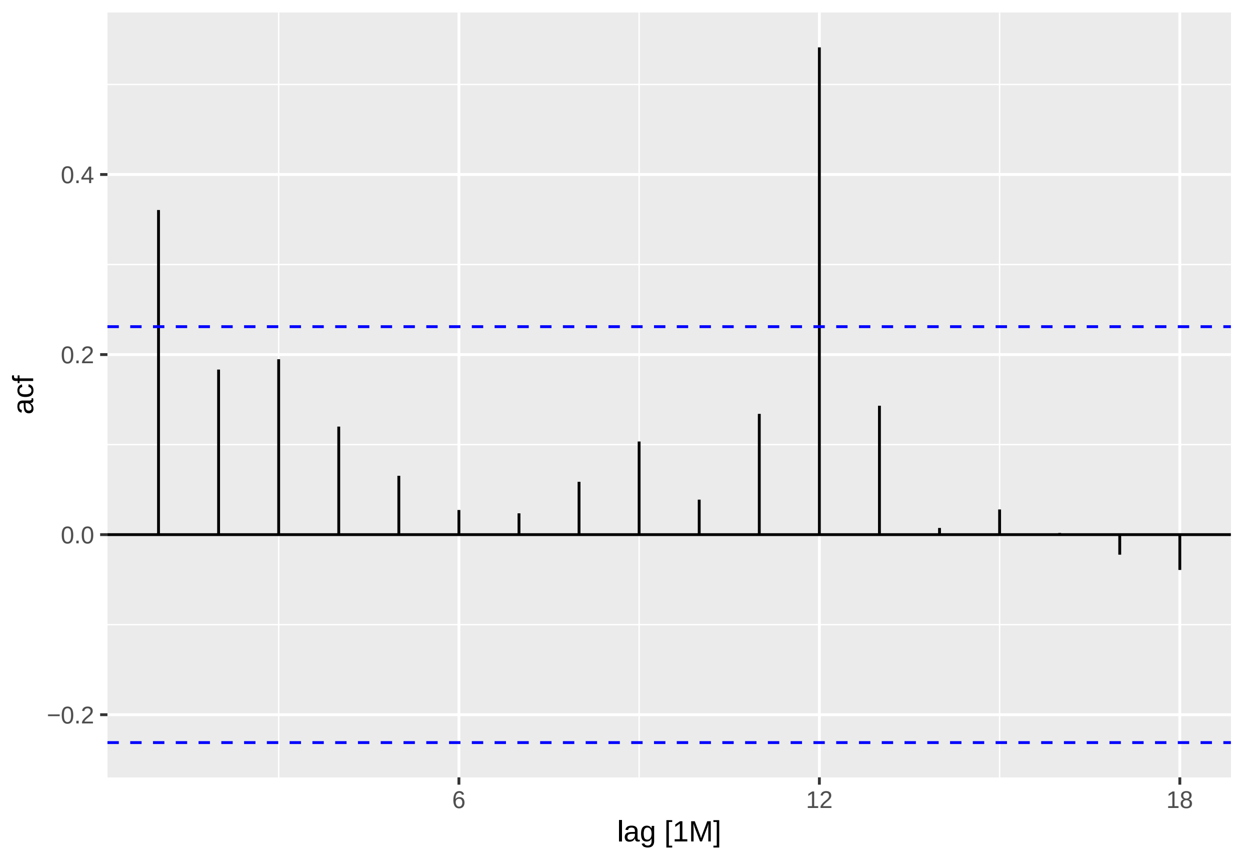
From the plot of the series, we can already grasp that it is not stationary, as the expected value is not constant over time and neither is variance, which tends to increase. In particular, we can speculate the presence of an upward trend and a seasonal effect, which is comprehensible, considering the tourist vocation of the shop.

```r
tseries %>% gg_season(sales) +
  theme_minimal()
```

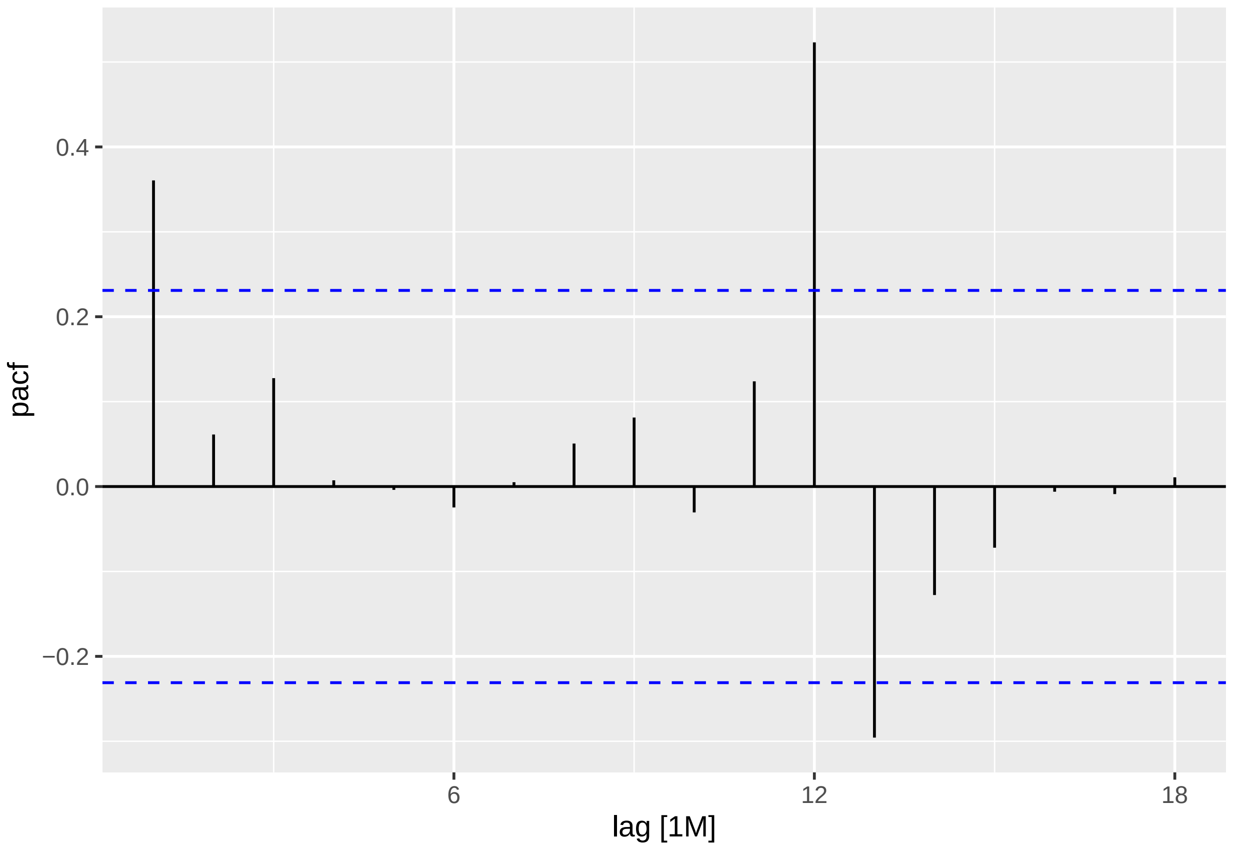


To better appreciate it, this plot shows the data for each year separately. The values of the series are clearly higher for the latest years and there is a recurring peak in the months of March and August, followed by a valley in October.

```r
tseries %>% ACF(sales) %>% autoplot()
```



```
tseries %>% PACF(sales) %>% autoplot()
```



Lastly, these are the *global* and *partial autocorrelation functions* for the series. The slow decay for the ACF suggests, once again, the existence of a trend, while the spikes at lag 12 indicate a probable seasonality.

To formalize our guesses, let's resort to two statistical test: - The **Augmented Dickey-Fuller test** tests the null hypothesis of the presence of a unit root in our time series - The **KPSS test** tests the null hypothesis that our data is stationary

```
tseries %>% as.ts() %>% adf.test()
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  .
## Dickey-Fuller = -1.2891, Lag order = 4, p-value = 0.8648
## alternative hypothesis: stationary
```

```
tseries %>% as.ts() %>% kpss.test()
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  .
## KPSS Level = 0.98879, Truncation lag parameter = 3, p-value = 0.01
```

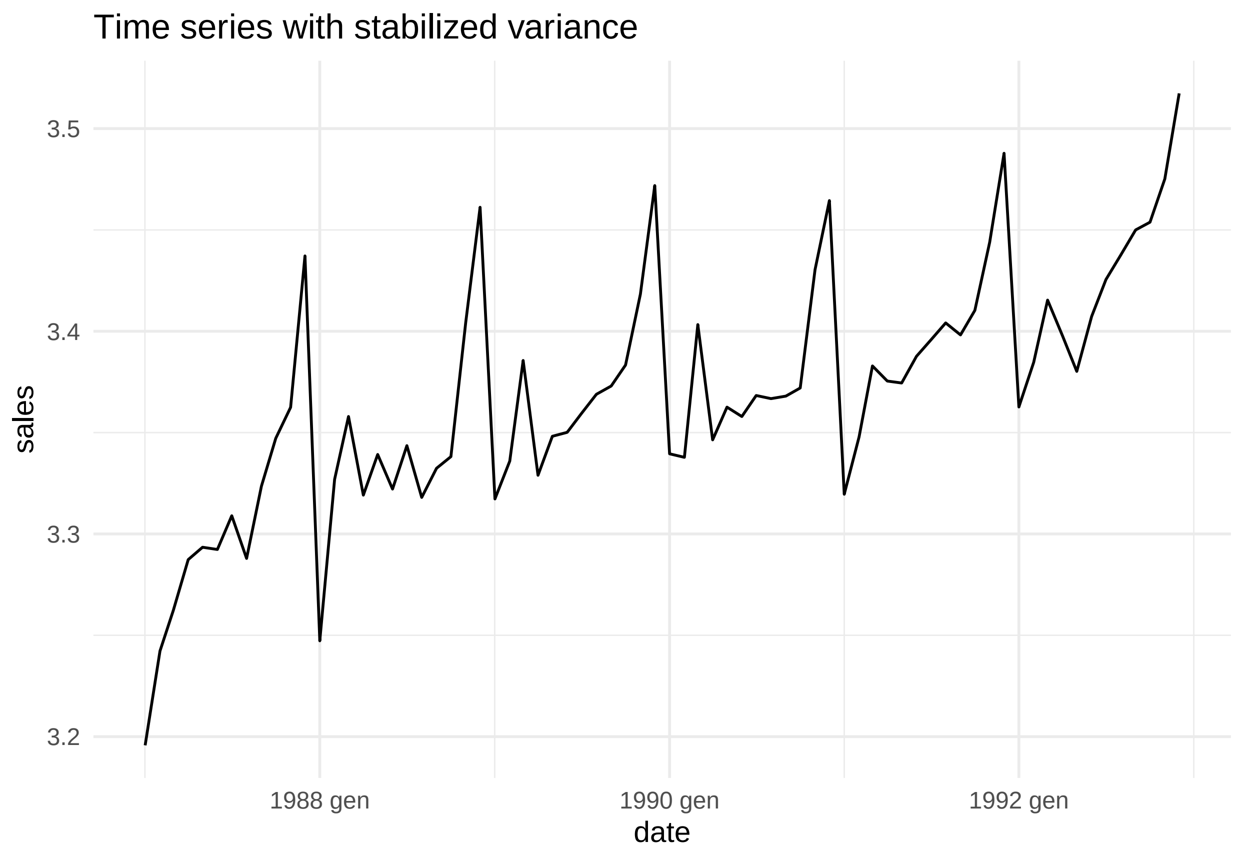
We were expecting to reject H0 for **KPSS** and not be able to reject it for **ADF** and that's exactly what happened, looking at the p-values.

Reach stationarity

In order to obtain stationarity in our time series, we need to perform a series of operations: we are going to stabilize the variance through the *Box-Cox transformation* and then apply differencing to treat trend and seasonality.

```
lambda <- BoxCox.lambda(tseries$sales)
bc <- BoxCox(tseries$sales, lambda = lambda)
tseries.novar <- tseries %>% mutate(sales = bc)
```

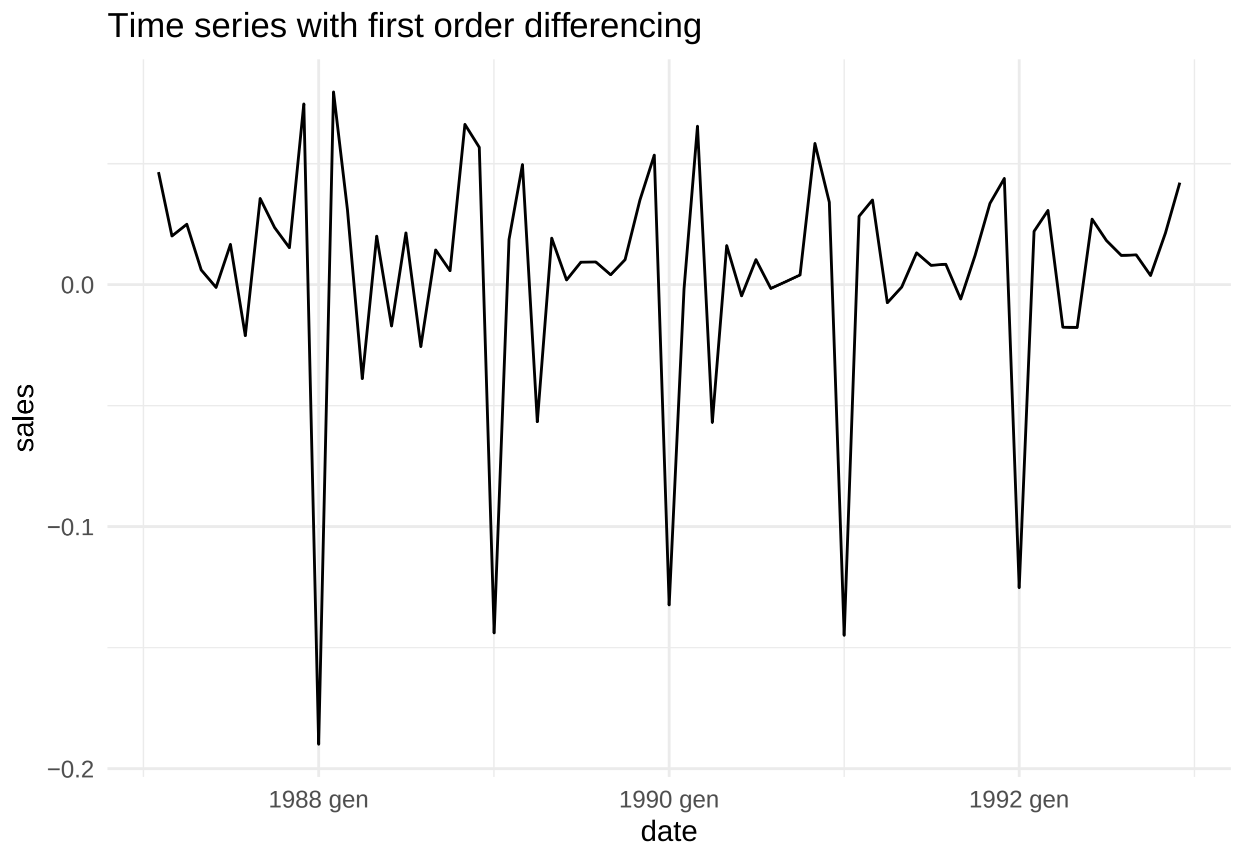
```
tseries.novar %>% ggplot(aes(x = date, y = sales)) +
  geom_line() +
  theme_minimal() +
  ggtitle("Time series with stabilized variance")
```



```
tseries.diff <- tseries.novar %>%
  mutate(date = date, sales = difference(sales, lag = 1, differences = 1)) %>%
  slice_tail(n = -1)
tseries.diff
```

```
## # A tsibble: 71 x 2 [1M]
##        date    sales
##       <mth>    <dbl>
##  1 1987 feb  0.0465
##  2 1987 mar  0.0201
##   3 1987 apr  0.0250
##   4 1987 mag  0.00610
##   5 1987 giu -0.00110
##   6 1987 lug  0.0166
##   7 1987 ago -0.0211
##   8 1987 set  0.0356
##   9 1987 ott  0.0237
##  10 1987 nov  0.0153
## # i 61 more rows
```

```
tseries.diff %>%
  ggplot(aes(x = date, y = sales)) +
  geom_line() +
  theme_minimal() +
  ggtitle("Time series with first order differencing")
```

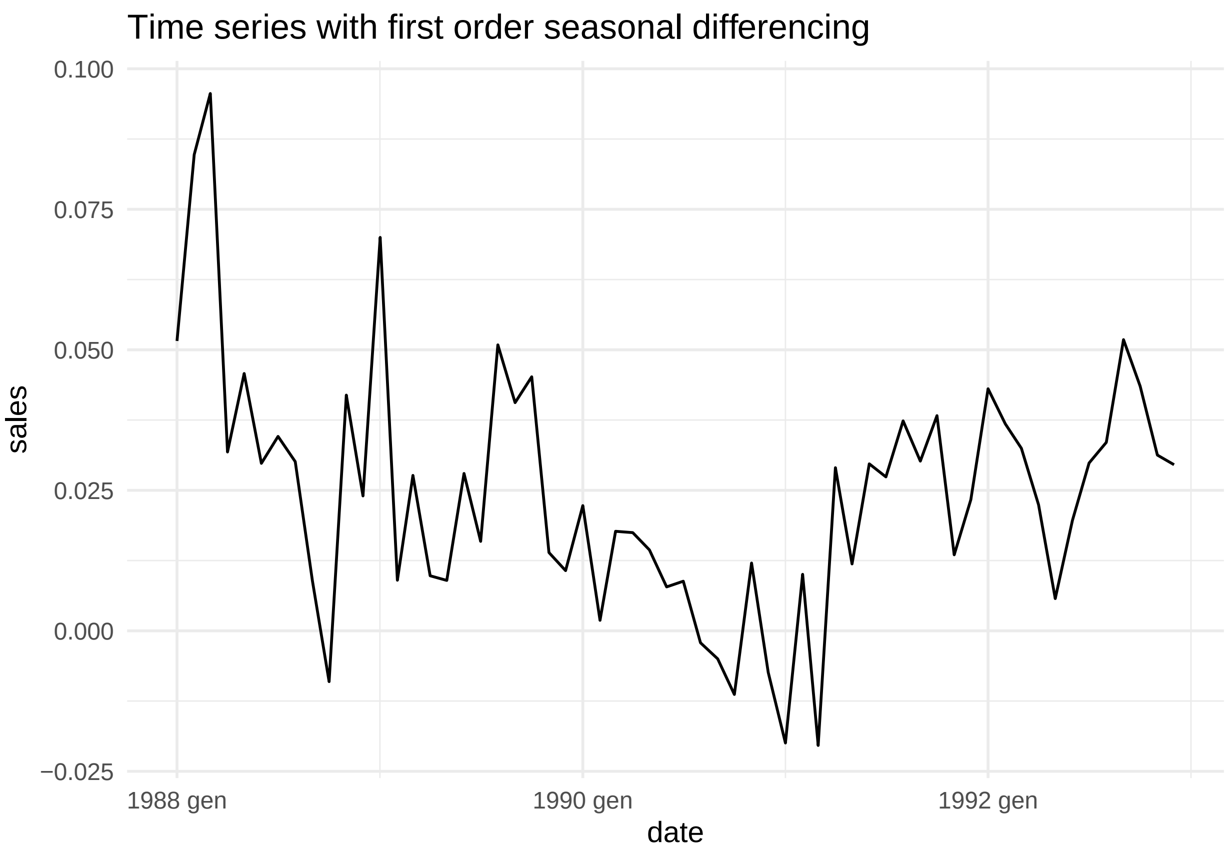


The first order differencing removes trend, but leaves seasonality.

```
tseries.diff.s <- tseries.novar %>%
  mutate(sales = difference(sales, lag = 12, differences = 1)) %>%
  slice_tail(n= -12)
tseries.diff.s
```

```
## # A tsibble: 60 x 2 [1M]
##         date    sales
##         <mth>    <dbl>
##  1 1988 gen  0.0516
##  2 1988 feb  0.0847
##  3 1988 mar  0.0956
##  4 1988 apr  0.0318
##  5 1988 mag  0.0458
##  6 1988 giu  0.0298
##  7 1988 lug  0.0346
##  8 1988 ago  0.0301
##  9 1988 set  0.00886
## 10 1988 ott -0.00904
## # i 50 more rows
```

```
tseries.diff.s %>%
  ggplot(aes(x = date, y = sales)) +
  geom_line() +
  theme_minimal() +
  ggtitle("Time series with first order seasonal differencing")
```

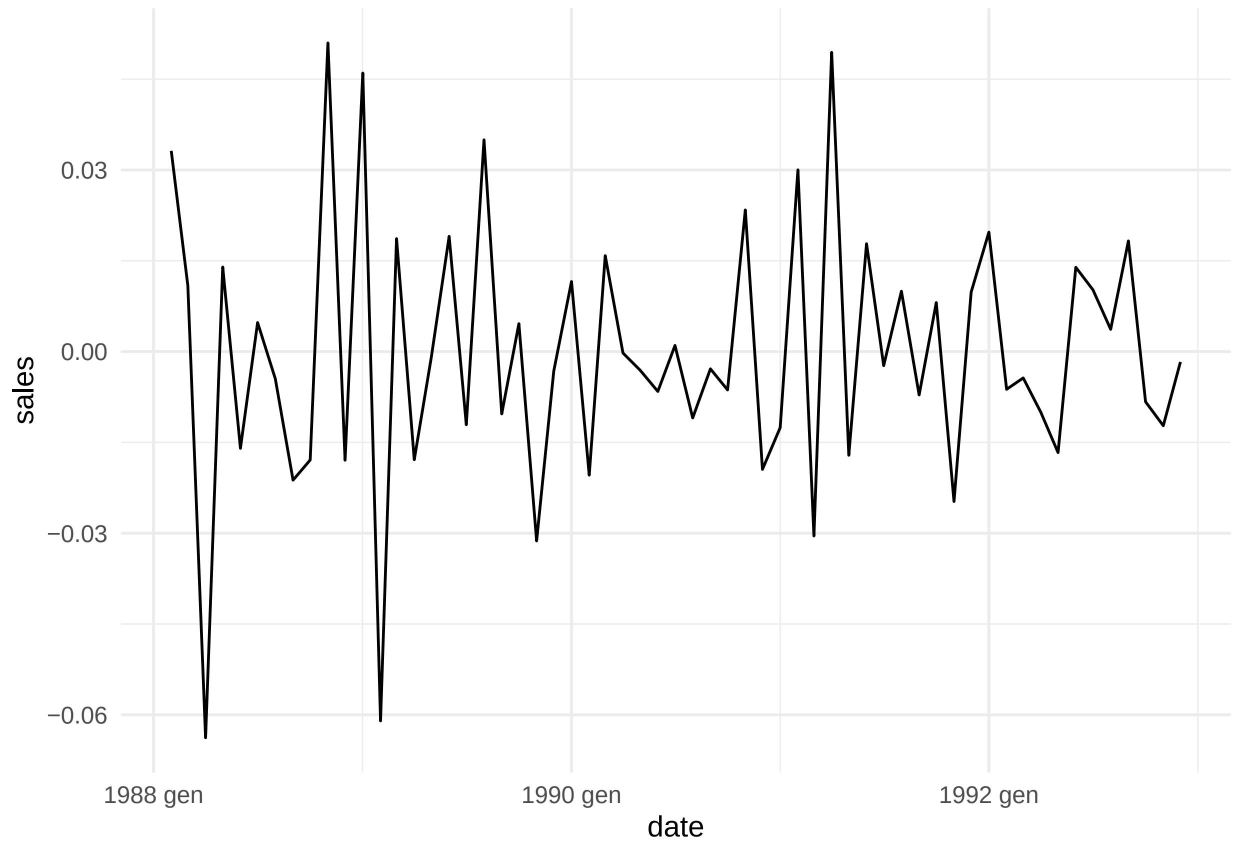


The seasonal differencing removes seasonality, but leaves trend.

```
tseries.diff.final <- tseries.novar %>%
  mutate(sales = difference(sales, lag = 1, differences = 1)) %>%
  slice_tail(n = -1) %>%
  mutate(sales = difference(sales, lag = 12, differences = 1)) %>%
  slice_tail (n = -12)
tseries.diff.final
```

```
## # A tsibble: 59 x 2 [1M]
##         date    sales
##        <mth>    <dbl>
##  1 1988 feb  0.0332
##  2 1988 mar  0.0109
##  3 1988 apr -0.0638
##  4 1988 mag  0.0140
##  5 1988 giu -0.0160
##  6 1988 lug  0.00479
##  7 1988 ago -0.00449
##  8 1988 set -0.0212
##  9 1988 ott -0.0179
## 10 1988 nov  0.0510
## # i 49 more rows
```

```
tseries.diff.final %>%
  ggplot(aes(x = date, y = sales)) +
  geom_line() +
  theme_minimal()
```



The new series should be stationary now. Let's check with our two tests.

```r
tseries.diff.final %>% as.ts() %>% adf.test()
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  .
## Dickey-Fuller = -6.0066, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

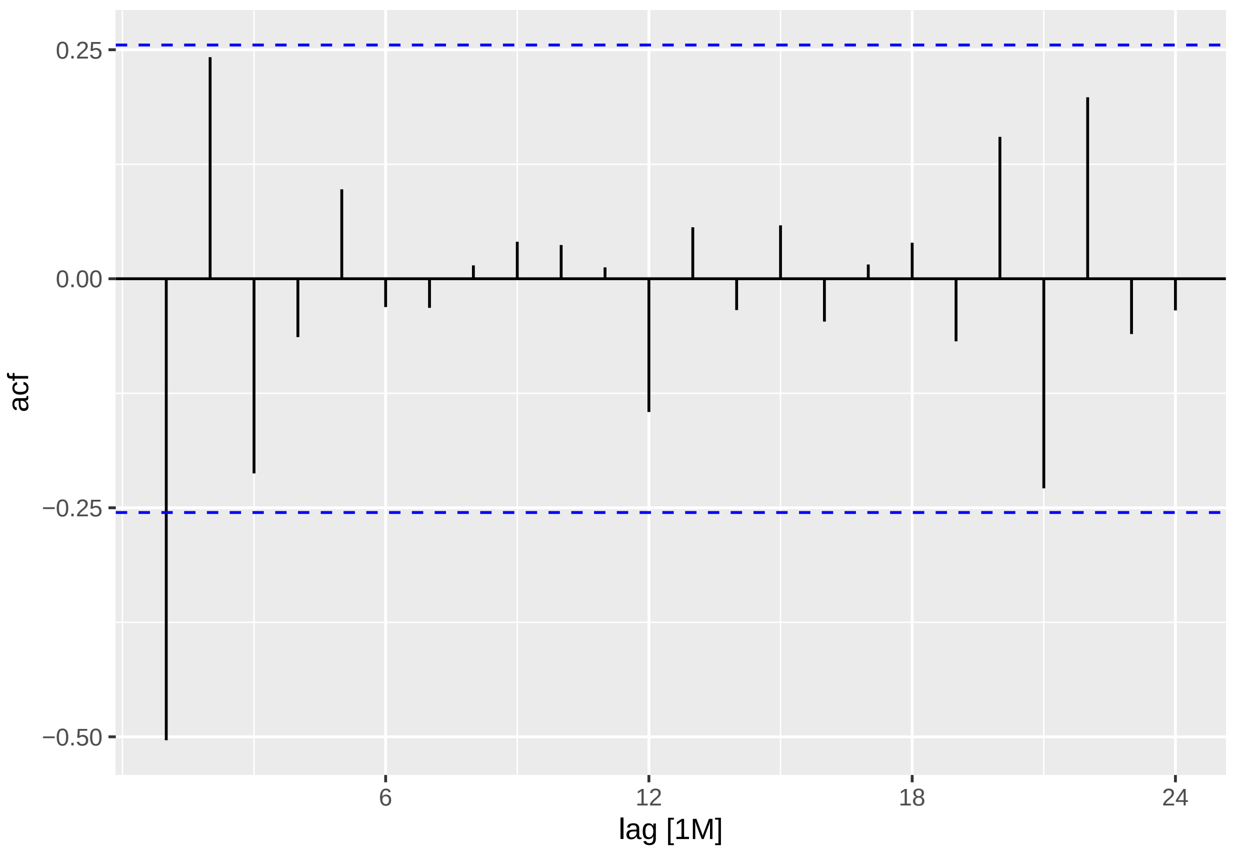
```r
tseries.diff.final %>% as.ts() %>% kpss.test()
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  .
## KPSS Level = 0.068468, Truncation lag parameter = 3, p-value = 0.1
```

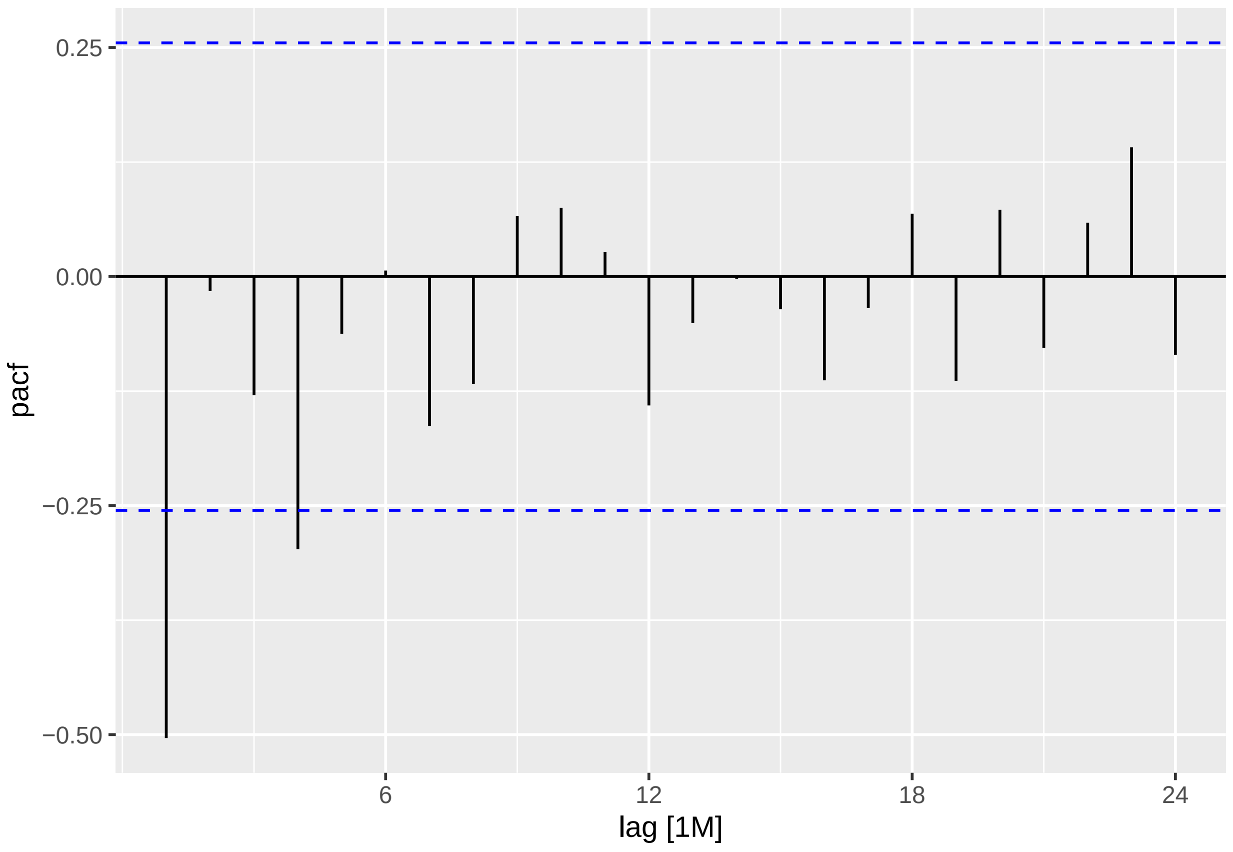
Our conclusions on the null hypothesis have now switched, as we expected.

Let's look at the ACF and PACF.

```r
tseries.diff.final %>% ACF(sales, lag_max = 24) %>% autoplot()
```



```r
tseries.diff.final %>% PACF(sales, lag_max = 24) %>% autoplot()
```



Estimate ARIMA model

At this point, we should be able to estimate the values for the parameters of our ARIMA model by looking at these two plots and the spikes in them. However, the most reliable way to actually determine the parameters is using an objective procedure, for example a stepwise-like, and let a computer do it for us by choosing among many ARIMA models the "best" one, in terms of optimizing a certain indicator.

```r
fit <- tseries %>%
  as.ts() %>%
  auto.arima(start.p = 1,start.q = 1, max.p = 3, max.q = 3,
             start.P = 0, seasonal = T, d = 1, D = 1, trace = T,
             stepwise = T, lambda = "auto")
```

```
##
##  ARIMA(1,1,1)(0,1,1)[12]                    : -3.783953
##  ARIMA(0,1,0)(0,1,0)[12]                    : 12.95279
##  ARIMA(1,1,0)(1,1,0)[12]                    : -3.506572
##  ARIMA(0,1,1)(0,1,1)[12]                    : -4.677704
##  ARIMA(0,1,1)(0,1,0)[12]                    : -1.072194
##  ARIMA(0,1,1)(1,1,1)[12]                    : Inf
##  ARIMA(0,1,1)(0,1,2)[12]                    : Inf
##  ARIMA(0,1,1)(1,1,0)[12]                    : -2.863223
##  ARIMA(0,1,1)(1,1,2)[12]                    : Inf
##  ARIMA(0,1,0)(0,1,1)[12]                    : 11.30466
##  ARIMA(0,1,2)(0,1,1)[12]                    : -2.804183
##  ARIMA(1,1,0)(0,1,1)[12]                    : -5.843042
##  ARIMA(1,1,0)(0,1,0)[12]                    : -0.9084839
##  ARIMA(1,1,0)(1,1,1)[12]                    : Inf
##  ARIMA(1,1,0)(0,1,2)[12]                    : -4.464014
##  ARIMA(1,1,0)(1,1,2)[12]                    : Inf
##  ARIMA(2,1,0)(0,1,1)[12]                    : -3.737659
##  ARIMA(2,1,1)(0,1,1)[12]                    : -1.63897
##
##  Best model: ARIMA(1,1,0)(0,1,1)[12]
```

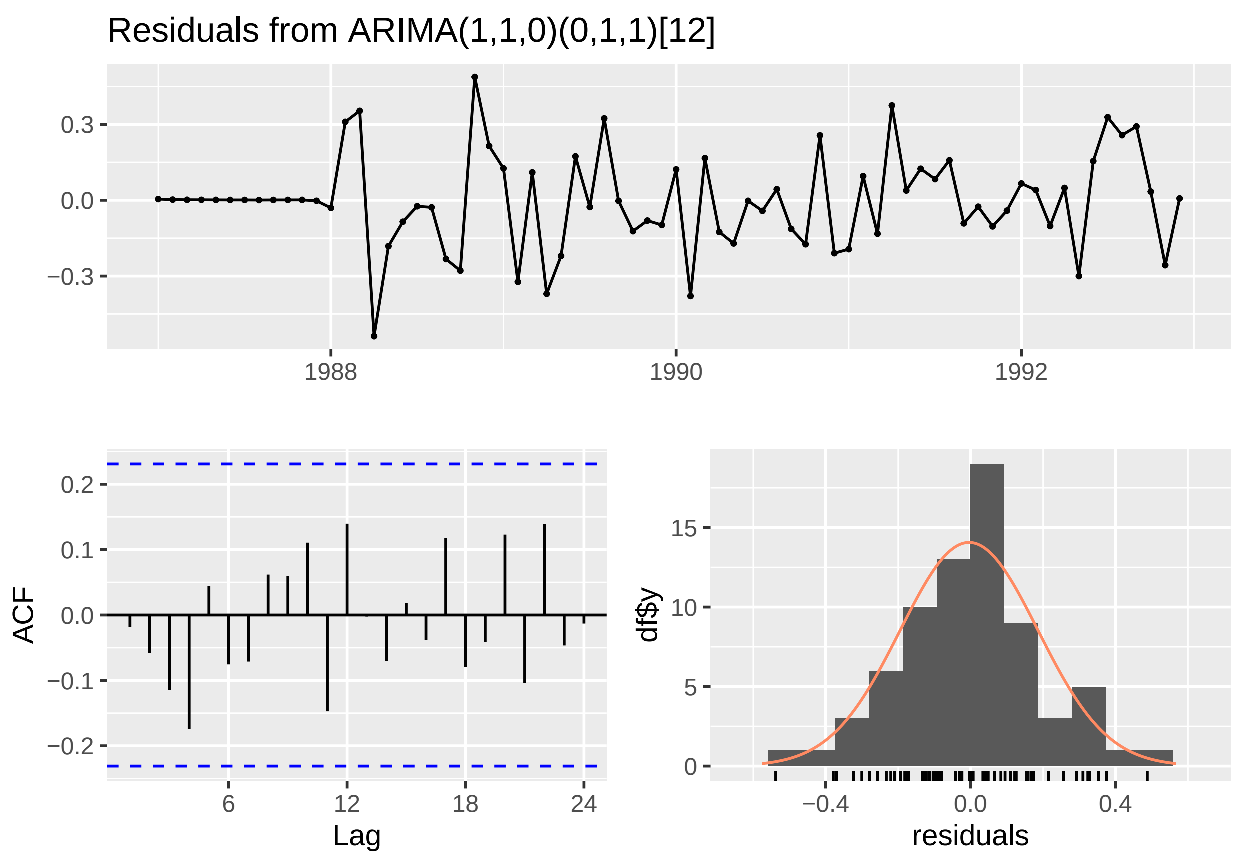
```
coeftest(fit)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1  -0.52834    0.10898 -4.8480 1.247e-06 ***
## sma1 -0.57352    0.22209 -2.5824  0.009812 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The chosen model appears to be *ARIMA(1,1,0)(0,1,1)[12]*. Along with it, we have also got a number of diagnostic tools: we can see that the parameters are significantly different than 0.

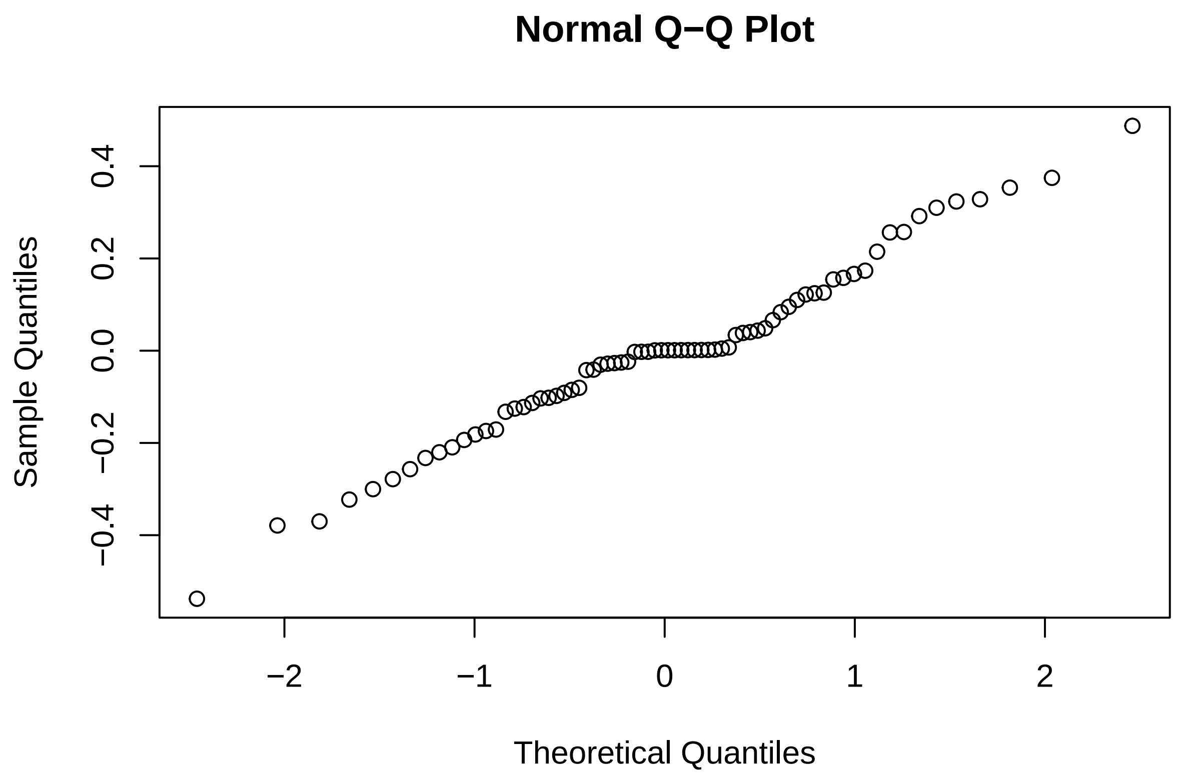
Let's look at other diagnostic measures through some plots.

```
checkresiduals(fit)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0)(0,1,1)[12]
## Q* = 10.467, df = 12, p-value = 0.575
##
## Model df: 2.   Total lags used: 14
```

```
qqnorm(fit$residuals)
```

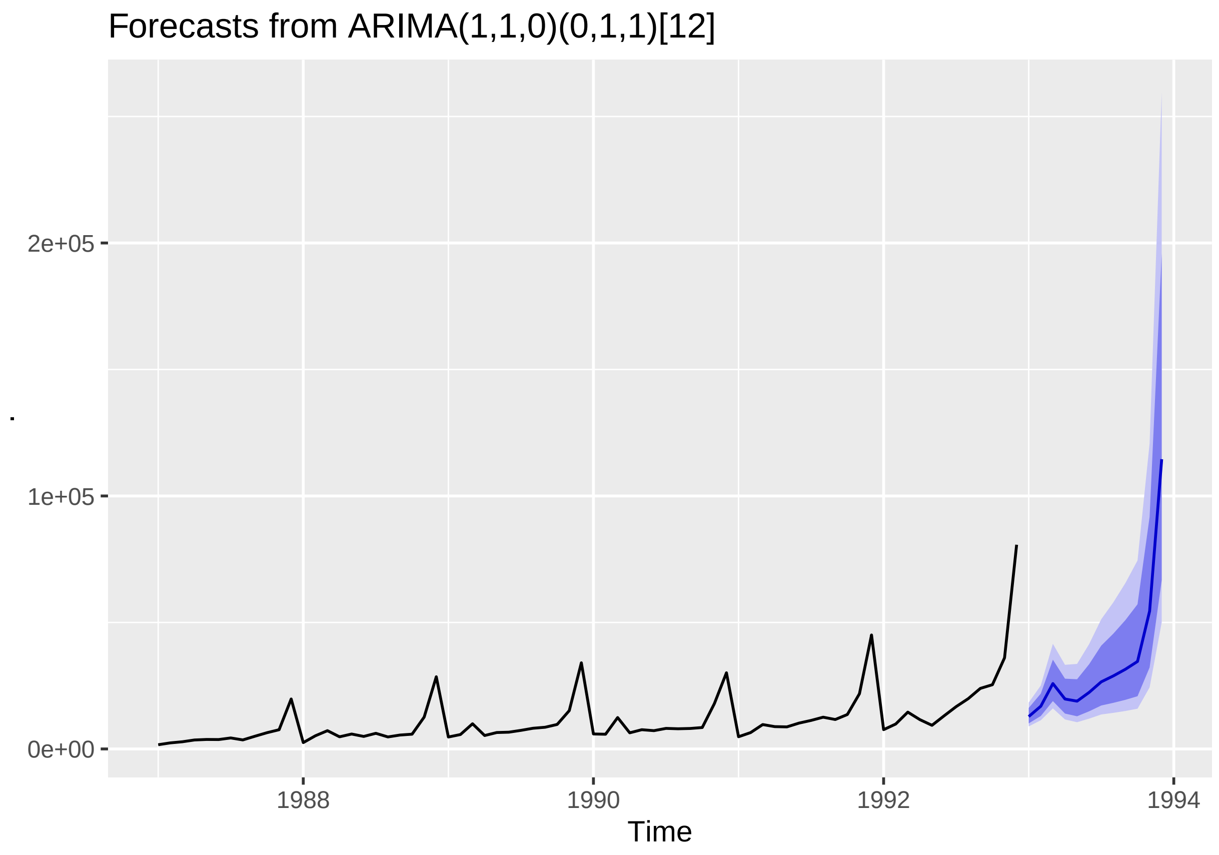


The residuals roughly follow a normal distribution, as deduced from the histogram and the Q-Q plot, but they seem to follow a pattern in their time series, which is not good for our model.

Forecast

For the last step, let's try to forecast some future values, in particular 12 more observations, and plot the result.

```
tseries.forecast <- forecast(fit, h = 12)
tseries.forecast %>% autoplot()
```


Forecasts from ARIMA(1,1,0)(0,1,1)[12]
2e+05
1e+05
0e+00
1988
1990
1992
1994
Time