TIME SERIES MODELLING - Number of Divorces

```
# load the relevant libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
library(TTR)
```

```
## Warning: package 'TTR' was built under R version 4.4.2
```

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.4.2

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

```
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 4.4.2
```

```
library(readxl)
```

```
# load the file
file_path = 'data/Vital statistics in the UK.xlsx'
sheet_name = 'Divorce'

divorce <- read_excel(file_path, sheet = sheet_name, skip=5)
head(divorce)
```

```
## # A tibble: 6 x 5
##    Year `United Kingdom` `England and Wales` Scotland `Northern Ireland`
##   <dbl> <chr>            <chr>               <chr>    <chr>
## 1  2021 :                111934              :        2040
## 2  2020 109874           102438              5929     1507
## 3  2019 117733           107599              7777     2357
## 4  2018 100241           90871               7297     2073
## 5  2017 110524           101669              6766     2089
## 6  2016 118046           106959              8515     2572
```

We are gonna model the time series of divorces in United Kingdom

```
# remove any observations that have ":" in United Kingdom
divorce <- divorce %>%
  filter(`United Kingdom` != ":")

head(divorce)
```

```
## # A tibble: 6 x 5
##    Year ‘United Kingdom‘ ‘England and Wales‘ Scotland ‘Northern Ireland‘
##   <dbl> <chr>            <chr>               <chr>    <chr>
## 1  2020 109874           102438              5929     1507
## 2  2019 117733           107599              7777     2357
## 3  2018 100241           90871               7297     2073
## 4  2017 110524           101669              6766     2089
## 5  2016 118046           106959              8515     2572
## 6  2015 112390           101055              8975     2360
```

Some EDA

```
str(divorce)
```

```
## tibble [50 x 5] (S3: tbl_df/tbl/data.frame)
##  $ Year             : num [1:50] 2020 2019 2018 2017 2016 ...
##  $ United Kingdom   : chr [1:50] "109874" "117733" "100241" "110524" ...
##  $ England and Wales: chr [1:50] "102438" "107599" "90871" "101669" ...
##  $ Scotland         : chr [1:50] "5929" "7777" "7297" "6766" ...
##  $ Northern Ireland : chr [1:50] "1507" "2357" "2073" "2089" ...
```

Looks like the number of divorces aren't numeric, so we have to change the type to numeric

```
# change the dtype to numeric and sort the dates in ascending order
divorce <- divorce %>%
  select(Year, `United Kingdom`) %>%
  mutate(`United Kingdom` = as.numeric(`United Kingdom`)) %>%
  arrange(Year)  # Sort the year from 1971 to 2020

str(divorce)
```

```
## tibble [50 x 2] (S3: tbl_df/tbl/data.frame)
##  $ Year          : num [1:50] 1971 1972 1973 1974 1975 ...
##  $ United Kingdom: num [1:50] 79588 124911 113531 121103 129278 ...
```

```
# get the summary statistics
summary(divorce)
```

```
##       Year      United Kingdom
##  Min.   :1971   Min.   : 79588
##  1st Qu.:1983   1st Qu.:127359
##  Median :1996   Median :154570
##  Mean   :1996   Mean   :145875
##  3rd Qu.:2008   3rd Qu.:164557
##  Max.   :2020   Max.   :180493
```

```
# check for missing values
sum(is.na(divorce))
```

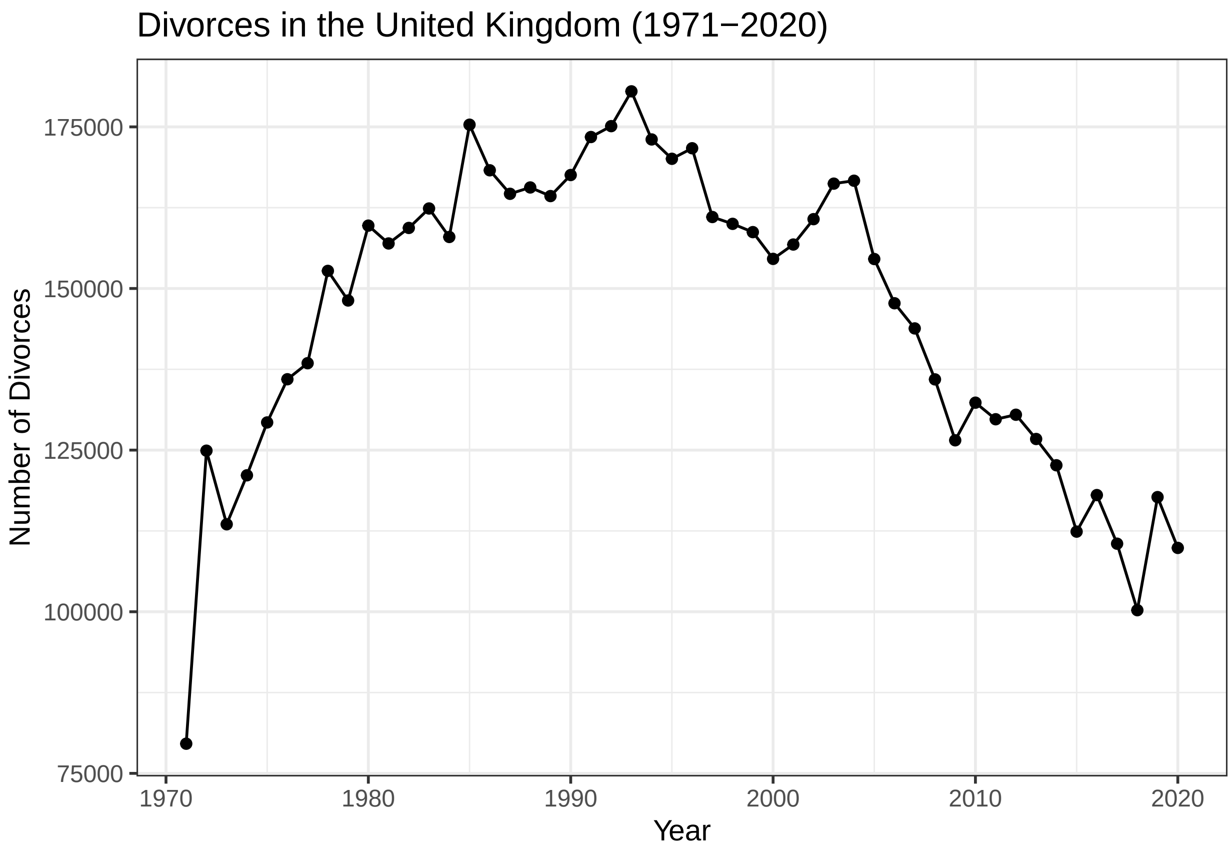
```
## [1] 0
```

```
# check for duplicates
sum(duplicated(divorce))
```

```
## [1] 0
```

No missing values and duplicates

```
divorce %>%
  ggplot(aes(x=Year, y=`United Kingdom`))+
  geom_point()+
  geom_line()+
  labs(title="Divorces in the United Kingdom (1971-2020)", y="Number of Divorces")+
  theme_bw()
```



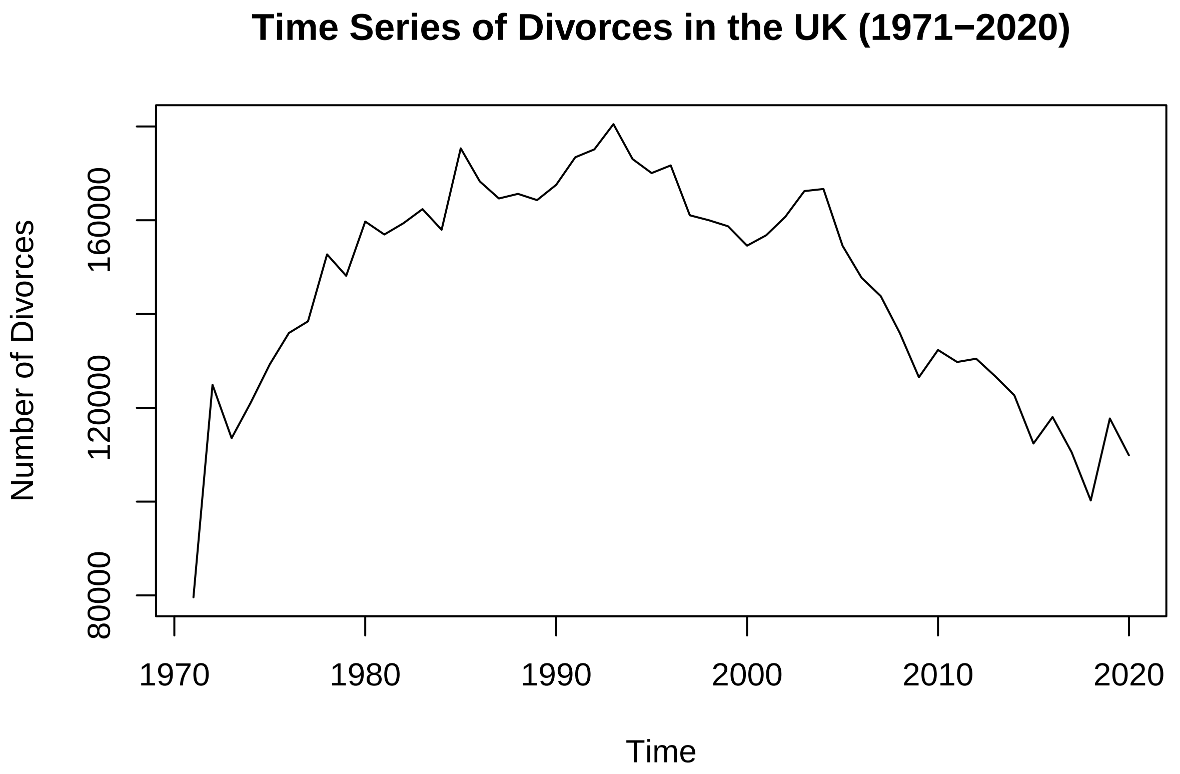
Convert Divorce to a time series

```
# Frequency is year by year -> 1, start = 1971
divorce_ts = ts(divorce$`United Kingdom`, frequency = 1, start = c(1971))
divorce_ts
```

```
## Time Series:
## Start = 1971
## End = 2020
## Frequency = 1
##  [1]  79588 124911 113531 121103 129278 135960 138445 152724 148144 159727
## [11] 156963 159365 162374 157968 175334 168283 164644 165629 164295 167551
## [21] 173430 175112 180493 173054 170062 171690 161055 159997 158720 154581
## [31] 156798 160725 166218 166669 154559 147717 143825 135942 126520 132338
## [41] 129773 130473 126719 122656 112390 118046 110524 100241 117733 109874
```

Plot the time series

```
plot.ts(divorce_ts, main='Time Series of Divorces in the UK (1971-2020)',
        ylab='Number of Divorces')
```

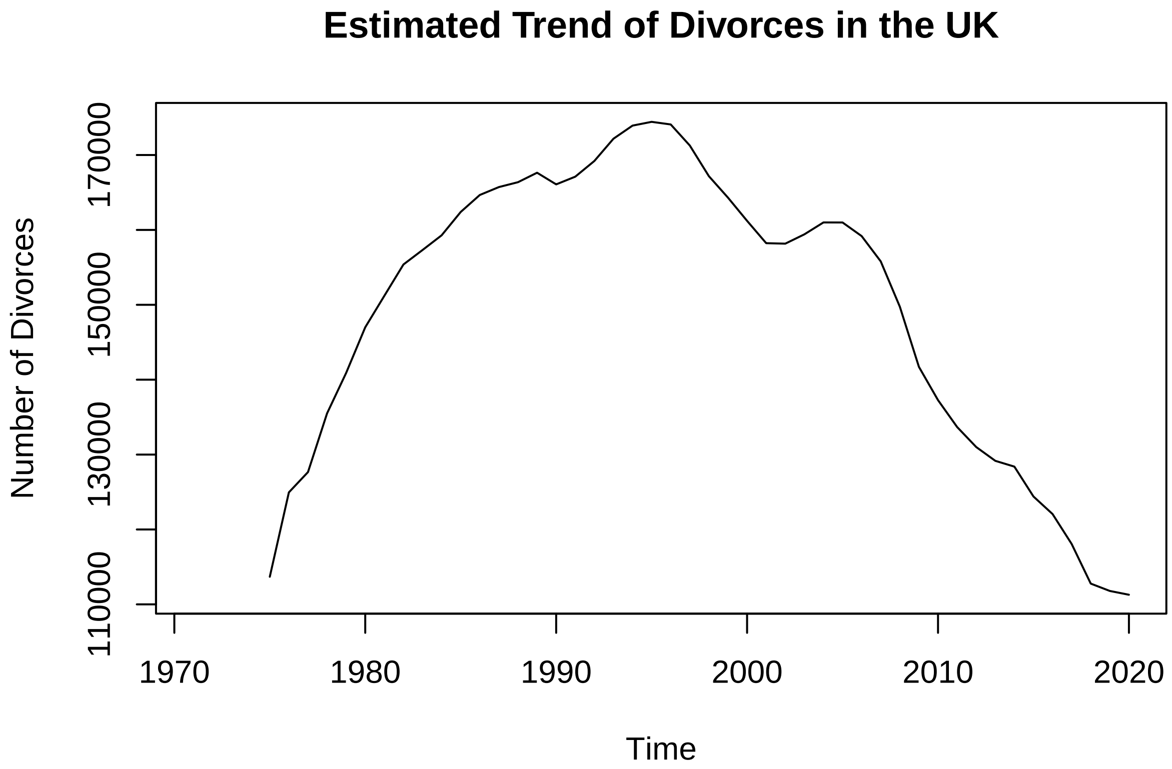


Observations: No seasonality, but there is a trend and some random noise. Can probably be described by an additive model.

Since there is no seasonality, we apply the decompose function, but we can extract and view the trend with a Simple Moving Average.

Simple Moving Average – to estimate the trend

```
sma_divorce <- SMA(divorce_ts, n=5)
plot.ts(sma_divorce, main="Estimated Trend of Divorces in the UK",
        ylab='Number of Divorces')
```



Observations: The SMA trend shows the number of divorces in the United Kingdom rose steadily from the 1970s to about 1995, and then started decreasing.

MODEL 1

FORECASTING - EXPONENTIAL SMOOTHING Since the time series shows a clear trend and no seasonal components, we will use Holts Exponential Smoothing with alpha & beta parameters to estimate the time series

```
# get the initial value and slope
initial_value <- divorce_ts[1]
slope <- divorce_ts[2] - initial_value
print(c(initial_value, slope))
```

```
## [1] 79588 45323
```

```
# fit a predictive model on the time series with Holt Winters
divorce_forecast_hw <- HoltWinters(divorce_ts, l.start = initial_value,
                                   b.start = slope, gamma = FALSE)
divorce_forecast_hw
```

```
## Holt-Winters exponential smoothing with trend and without seasonal component.
##
## Call:
## HoltWinters(x = divorce_ts, gamma = FALSE, l.start = initial_value,     b.start = slope)
##
## Smoothing parameters:
```

```
##  alpha: 0.7163539
##  beta : 0.9942052
##  gamma: FALSE
##
## Coefficients:
##         [,1]
## a 112489.524
## b   1579.476
```

Alpha (0.7163539): This is the smoothing parameter for the level component. A value close to 1 indicates that the model gives more weight to recent observations, making it highly responsive to recent changes

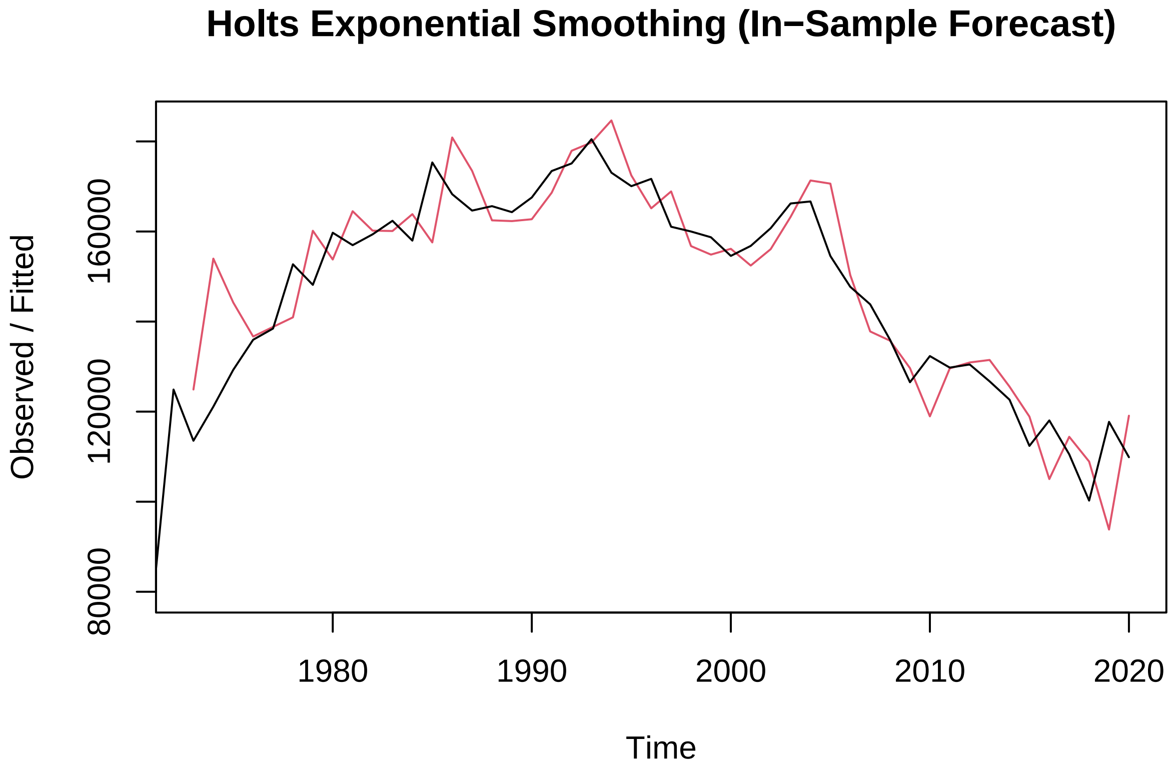
Beta (0.9942052): This is the smoothing parameter for the trend component. Very high value suggests that the estimate of the trend component is based on the most recent observations

```
# sum of squares errors
divorce_forecast_hw$SSE
```

```
## [1] 4296849564
```

Plot the in-sample forecast

```
plot(divorce_forecast_hw, main='Holts Exponential Smoothing (In-Sample Forecast)')
```



The in-sample forecasts looks like a decent fit, but it is slightly off in a few areas.

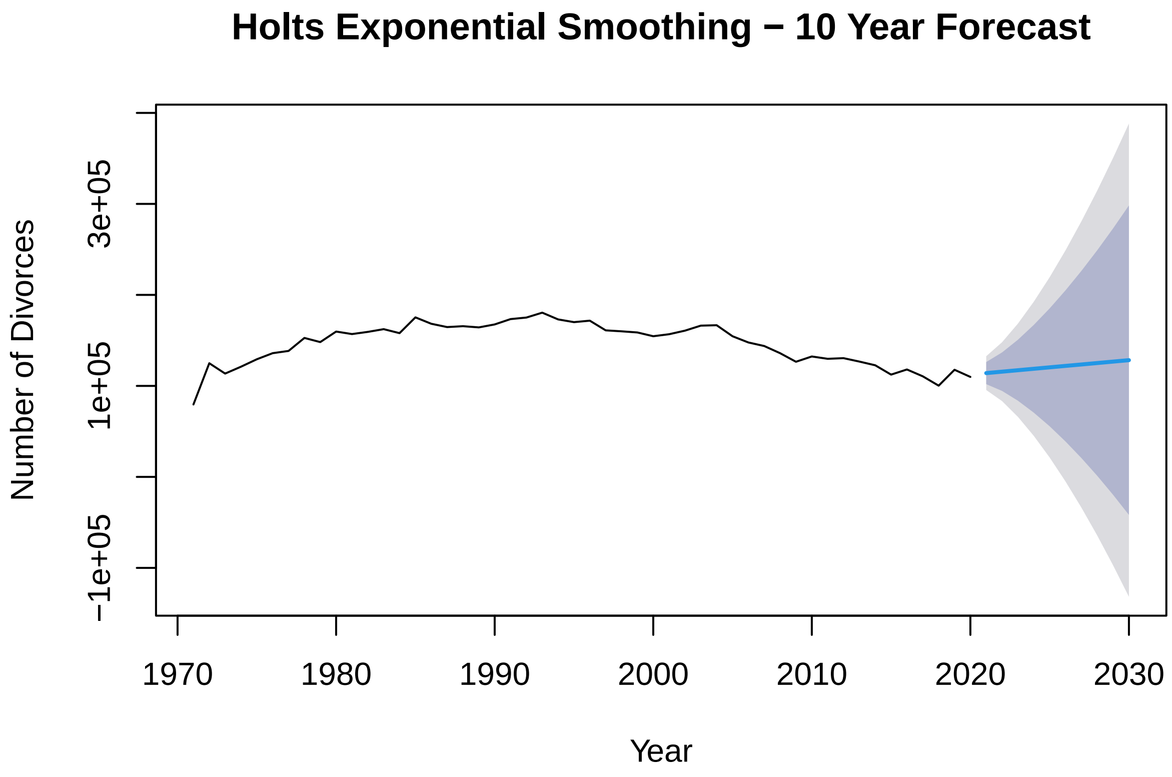
Plot a 10 year forecast of the Holt model

```
divorce_forecast_hw_10 <- forecast(divorce_forecast_hw, h=10)
divorce_forecast_hw_10
```

```
##      Point Forecast       Lo 80    Hi 80       Lo 95    Hi 95
## 2021       114069.0 101928.021 126210.0   95500.975 132637.0
## 2022       115648.5  94477.260 136819.7   83269.896 148027.1
## 2023       117228.0  83705.600 150750.3   65959.939 168496.0
## 2024       118807.4  70604.501 167010.4   45087.414 192527.4
## 2025       120386.9  55602.043 185171.8   21307.010 219466.8
## 2026       121966.4  38936.555 204996.2   -5016.779 248949.5
## 2027       123545.9  20766.097 226325.6  -33642.220 280733.9
## 2028       125125.3   1206.831 249043.8  -64391.659 314642.3
## 2029       126704.8 -19650.360 273060.0  -97126.105 350535.7
## 2030       128284.3 -41731.409 298300.0 -131732.280 388300.8
```

The ten year forecast show a slight increase in the number of divorces yearly from 2021 to 2030.

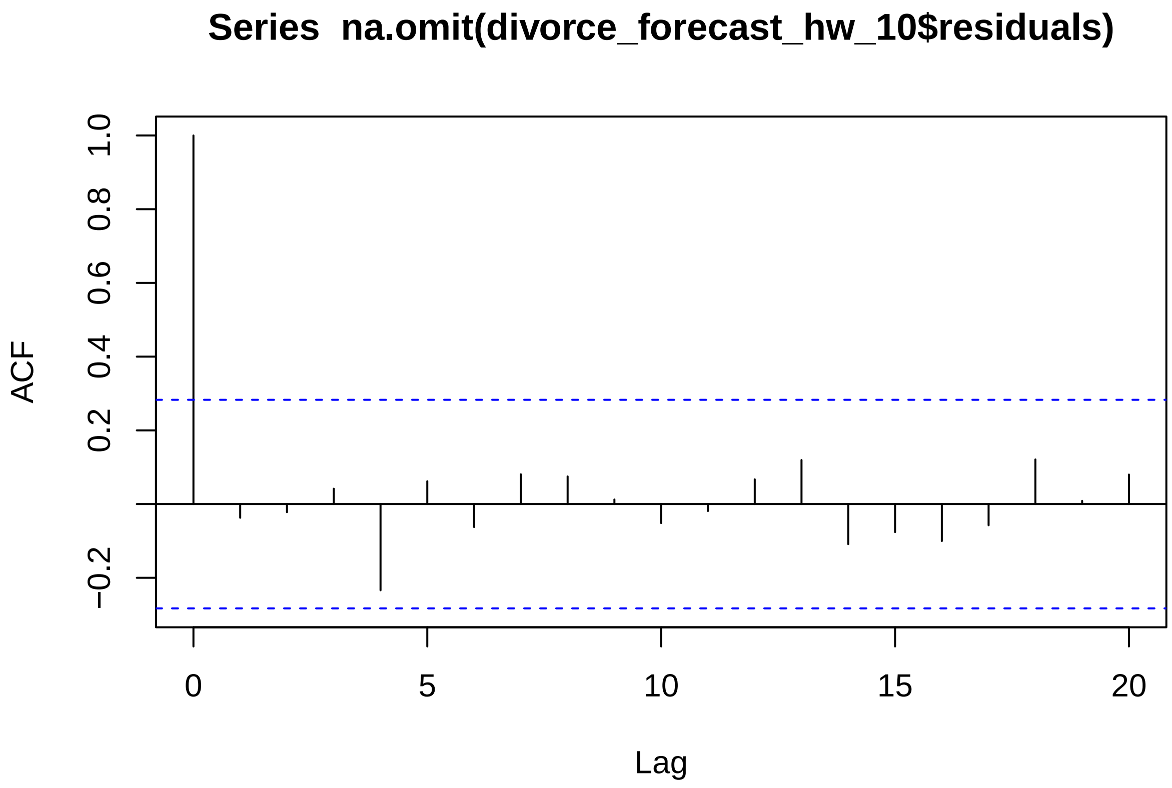
```
plot(divorce_forecast_hw_10, main="Holts Exponential Smoothing - 10 Year Forecast",
     xlab="Year", ylab="Number of Divorces")
```



The ten year forecast show a slight increase in the number of divorces yearly from 2021 to 2030.

Check whether the predictive model is good by checking the auto correlations

```r
# plot the acf correlogram
acf(na.omit(divorce_forecast_hw_10$residuals), lag.max=20, na.action=na.pass)
```



The ACF correlogram shows no significant spikes that exceed the bounds at any lag, suggesting that there are no significant auto correlations.

Ljung-Box test

```r
# Ljung box test
Box.test(na.omit(divorce_forecast_hw_10$residuals), lag=20, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  na.omit(divorce_forecast_hw_10$residuals)
## X-squared = 9.8094, df = 20, p-value = 0.9715
```

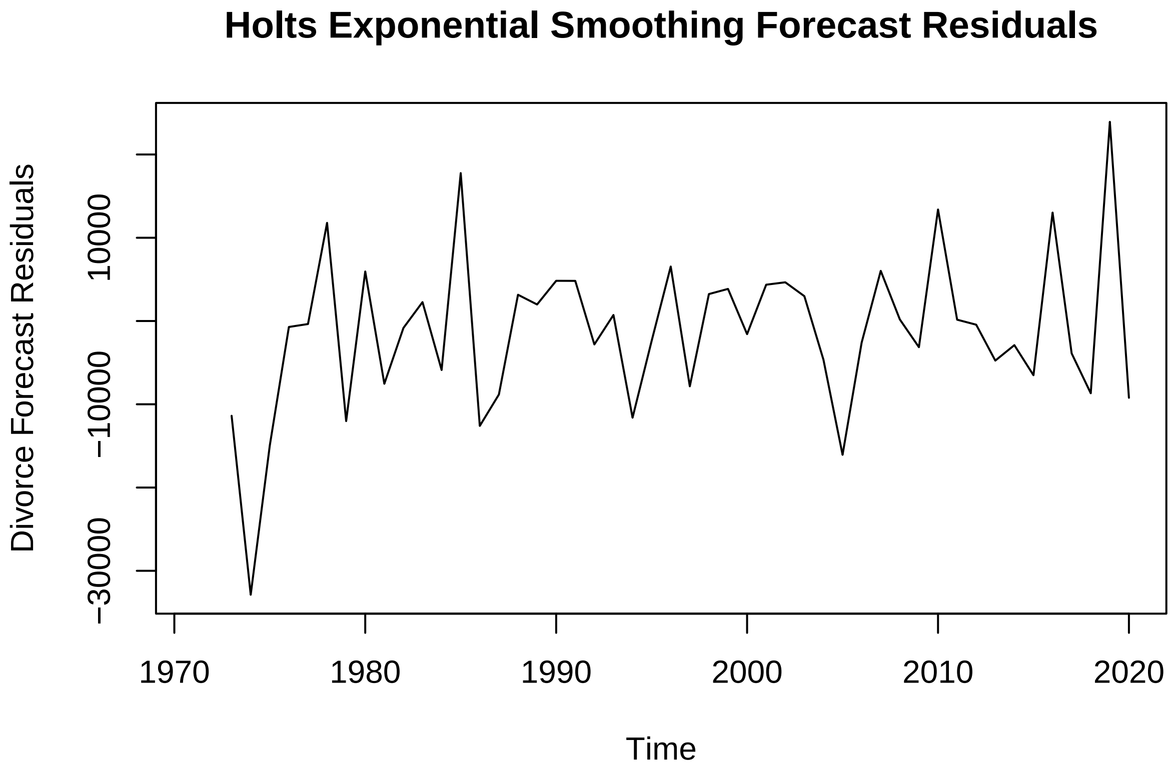
The p-value for Ljung-Box test is greater than 0.05, we fail to the null hypothesis that the residuals are uncorrelated. This further confirms that the Holt model has no significant auto correlations.

Check if forecast errors have constant variance over time, and are normally distributed with mean zero

```r
# Function to plot the histogram distribution of the forecast errors in red, overlaid with a normal dis
# Copied from the book "A little book of R for time series" by Avril Coghlan
# Coghlan, A. (2018). A little book of R for time series. Creative Commons Attribution 3.0 License
```

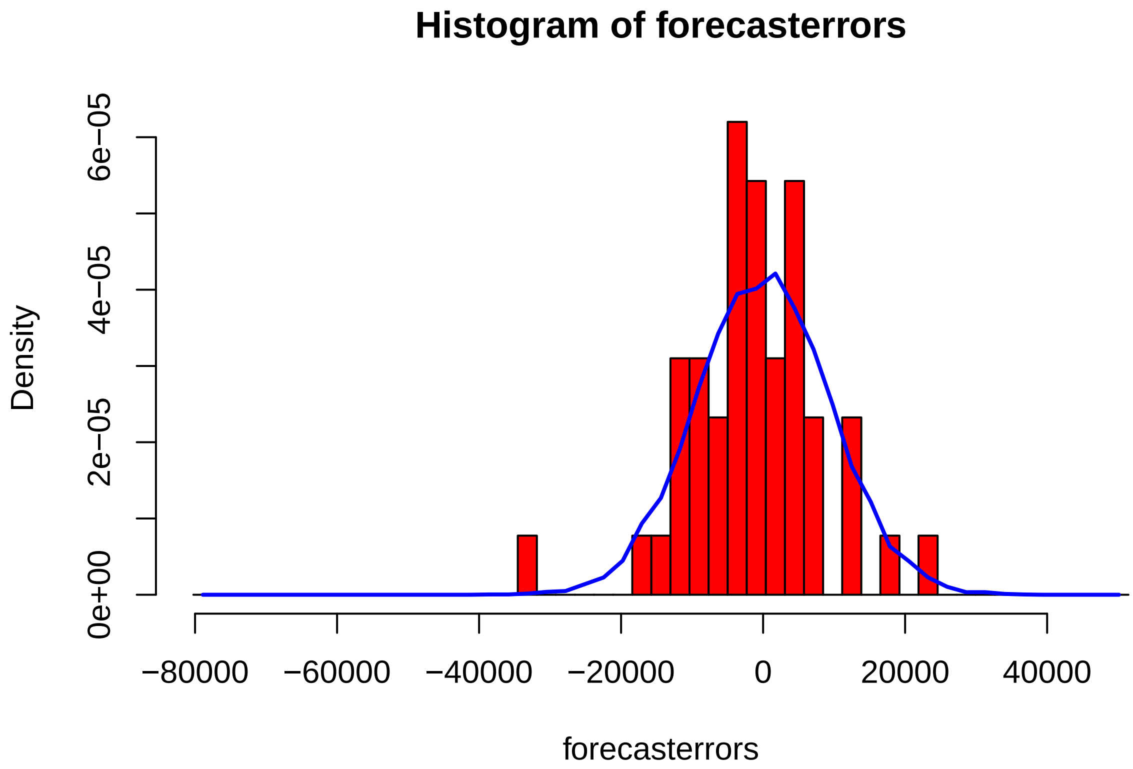
```r
plotForecastErrors <- function(forecasterrors)
{
# make a histogram of the forecast errors:
mybinsize <- IQR(forecasterrors)/4
mysd <- sd(forecasterrors)
mymin <- min(forecasterrors) - mysd*5
mymax <- max(forecasterrors) + mysd*3
# generate normally distributed data with mean 0 and standard deviation 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)
}
```

```r
# plot time series of the residuals
plot.ts(divorce_forecast_hw_10$residuals, ylab="Divorce Forecast Residuals",
        main="Holts Exponential Smoothing Forecast Residuals")
```



The plot of the forecast residuals show that the variance appear to be roughly constant over time.

```
# plot the histogram distribution of the forecast errors overlaid by a normal distribution curve
divorce_forecast_hw_10$residuals <-
  divorce_forecast_hw_10$residuals[!is.na(divorce_forecast_hw_10$residuals)]
plotForecastErrors(divorce_forecast_hw_10$residuals)
```



The distribution is roughly centered around 0 and it looks normally distributed.

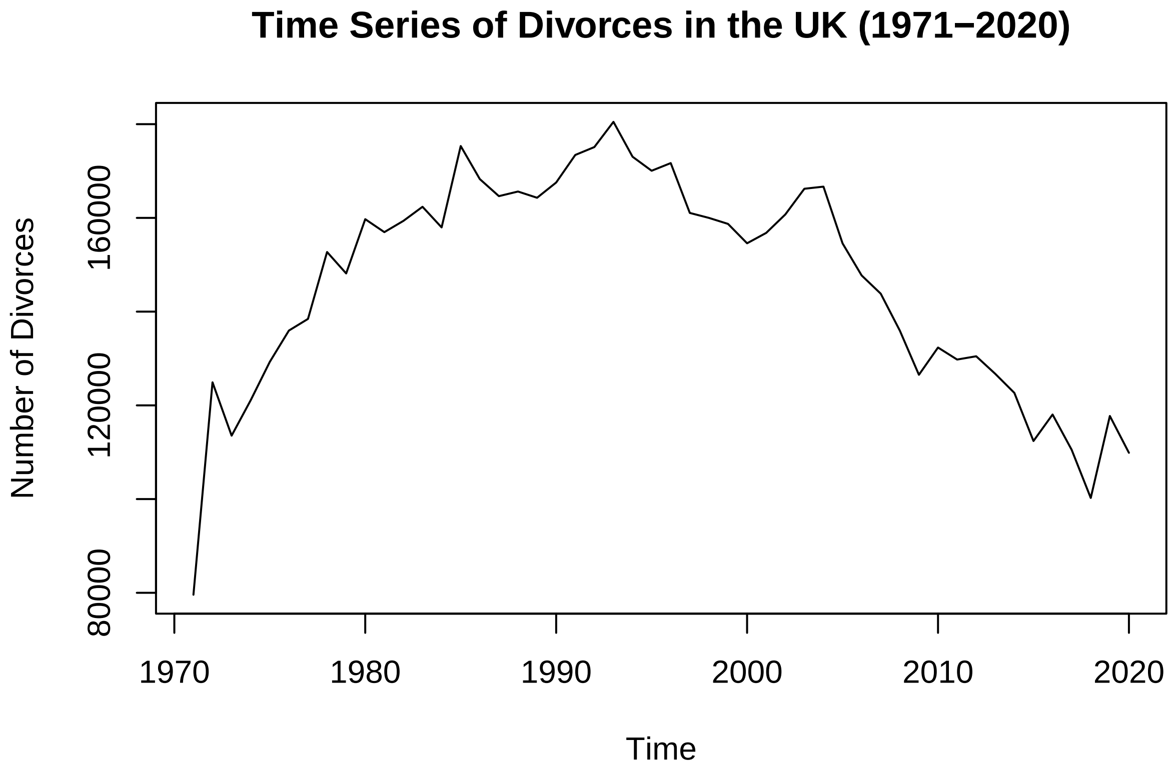
Therefore, it is plausible that the forecast errors are normally distributed with mean zero and constant variance.

Since forecast errors also have no significant auto correlations, and the forecast errors appear to be normally distributed with mean zero and constant variance, the Holt Exponential Smoothing model seems to provide an adequate predictive model for the number of divorces in the United Kingdom.

ARIMA MODELS

1. Check for stationarity

```
plot.ts(divorce_ts, ylab='Number of Divorces',
        main='Time Series of Divorces in the UK (1971-2020)')
```



The trend shows that the series is clearly not stationary.

But to confirm, we run Augmented Dickey-Fuller (ADF) test Null Hypothesis (H0): The time series has a unit root, this means that it is non-stationary.

Alternative Hypothesis (H1): The time series does not have a unit root, this means that it is stationary.

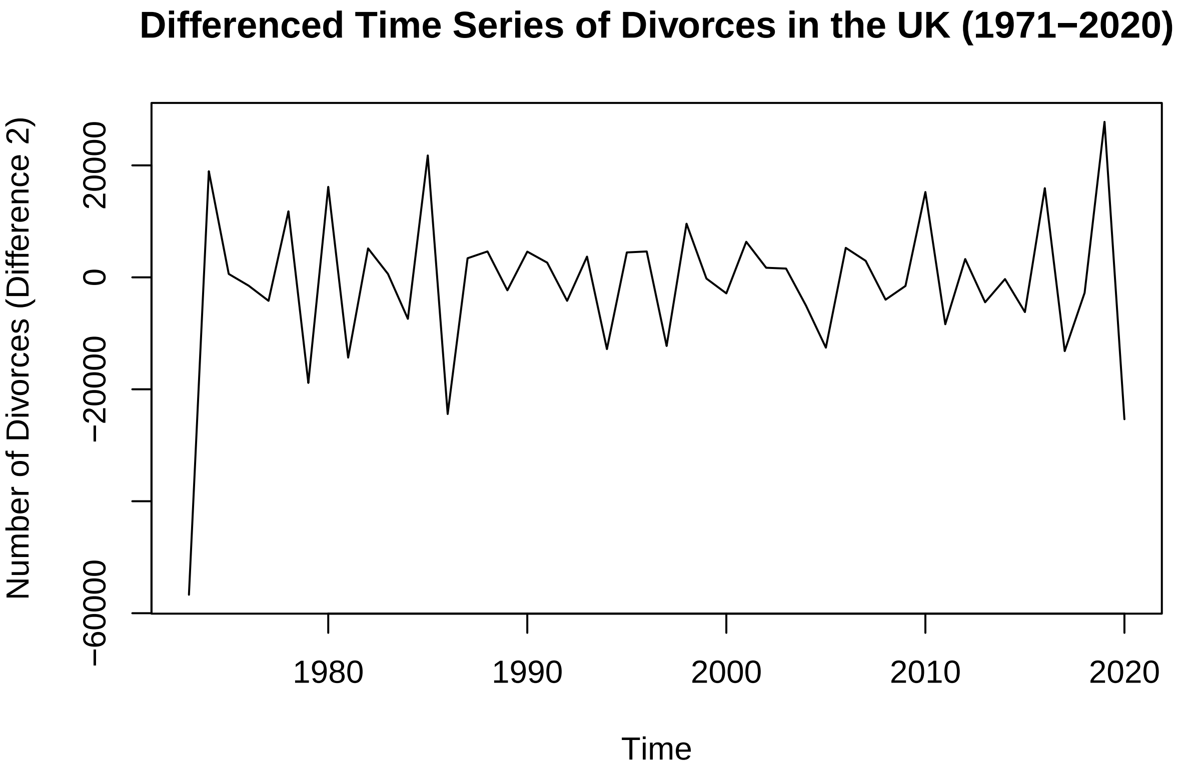
```
# run the ADF test
adf.test(divorce_ts)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  divorce_ts
## Dickey-Fuller = -2.5017, Lag order = 3, p-value = 0.373
## alternative hypothesis: stationary
```

p-value is 0.373 which is > 0.05 so we fail to reject the null hypothesis. Indicating that the series is not stationary.

So we apply differencing

```
# apply differencing of 2 orders
divorce_ts_diff2 <- diff(divorce_ts, differences = 2)
plot.ts(divorce_ts_diff2, ylab='Number of Divorces (Difference 2)',
        main='Differenced Time Series of Divorces in the UK (1971-2020)')
```



The time series appear to be stationary in mean and variance, but we can confirm it with the ADF test

```
# run the ADF test
adf.test(divorce_ts_diff2)
```

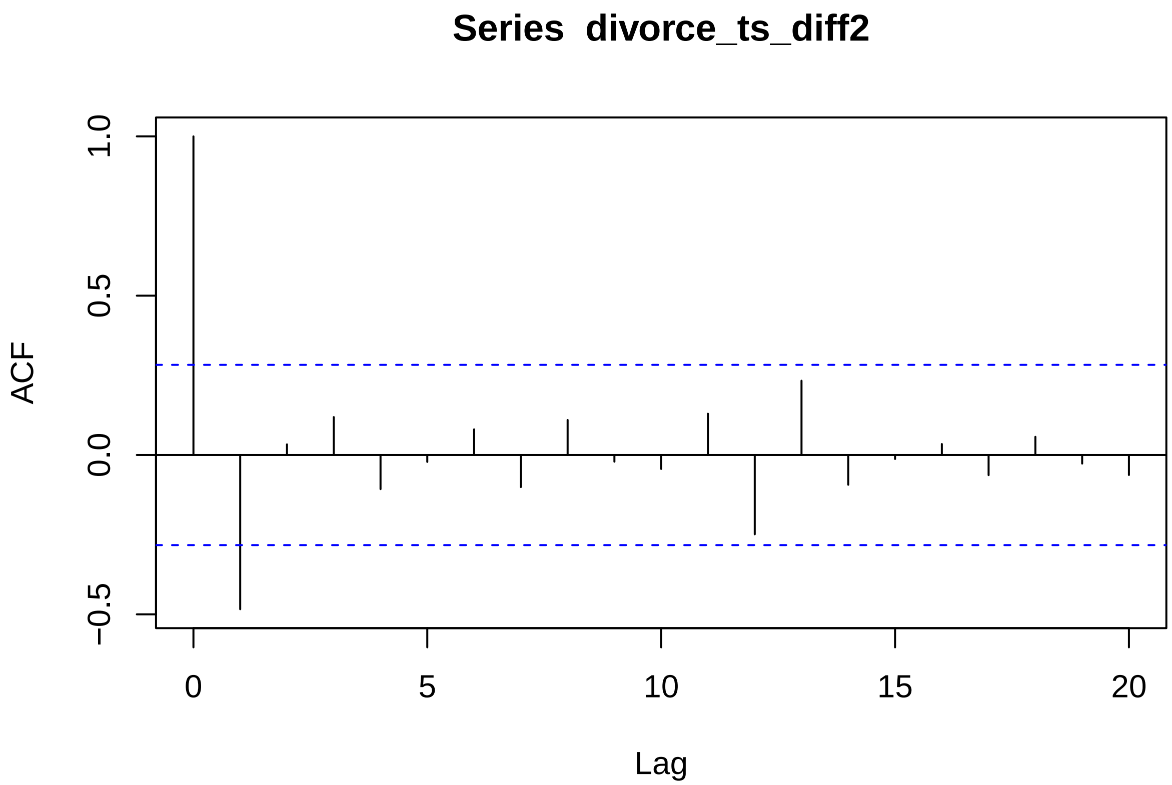
```
## Warning in adf.test(divorce_ts_diff2): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  divorce_ts_diff2
## Dickey-Fuller = -4.2344, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

p-value is less than 0.05 so we reject the null hypothesis and accept the Alternative Hypothesis that the time series is Stationary.

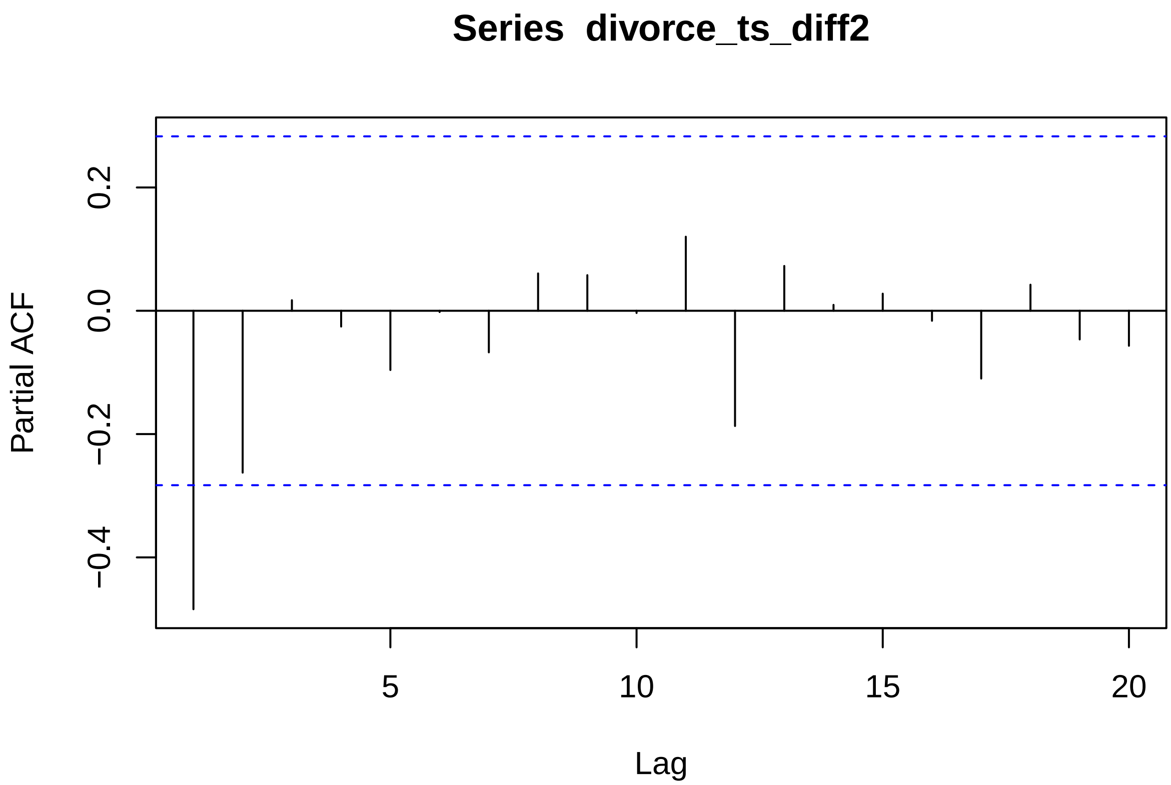
In order to decide on which ARIMA model to use, Plot the correlograms

```
# plot the ACF
acf(divorce_ts_diff2, lag.max = 20)
```



The ACF plot shows that there is 1 spike at lag 1 that exceeds the significance bounds, but all other lags autocorrelations do not exceed the bounds, so ARMA(0,1) seems plausible.

```r
# Plot the PACF
pacf(divorce_ts_diff2, lag.max = 20)
```



The PACF plots show 1 auto correlation that passes the significance bounds at lag 1, but the other auto correlations for lages 2-20 do not exceed the bounds. ARMA(1,0) seems like a good model.

So we can pick either ARMA(0,1) OR ARMA(1,0) for our second model

MODEL 2

ARMA(1,0) model (with p=1, q=0) can be modelled using ARIMA(1,2,0) model (with p=1, d=2, q=0) d = 2 (differencing of 2 orders)

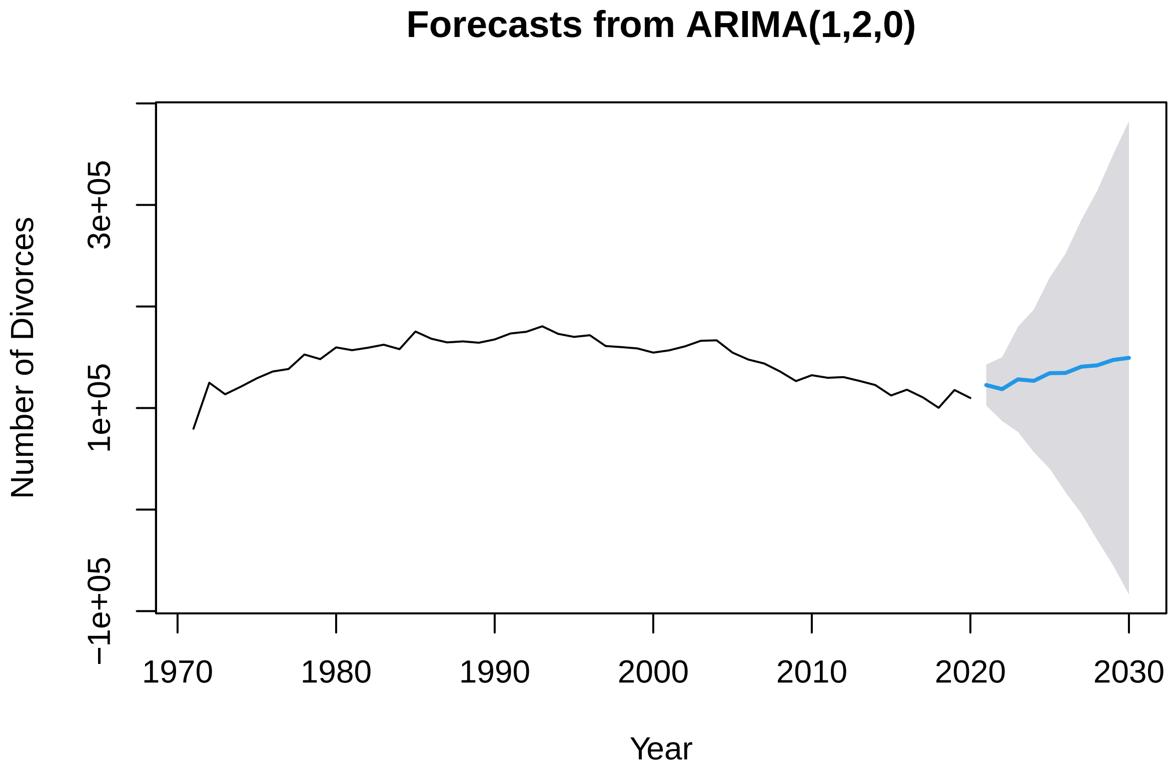
```
# Modelling with ARIMA(1,2,0)
divorce_arima_1 <- arima(divorce_ts, order = c(1,2,0))
divorce_arima_1
```

```
##
## Call:
## arima(x = divorce_ts, order = c(1, 2, 0))
##
## Coefficients:
##           ar1
##       -0.8128
## s.e.   0.1288
##
## sigma^2 estimated as 106387654:  log likelihood = -512.23,  aic = 1028.46
```

X_t - mu = Beta1 * (X_t-1 - mu) + Z_t, • where X_t is the stationary time series we are studying (the time series of number of divorces in the UK), • mu is the mean of time series X_t, • Beta1 is the parameter to be estimated, in this case, it is ar1 = -0.8128, • and Z_t is white noise with mean zero and constant variance.

10 year forecast for 95% confidence level

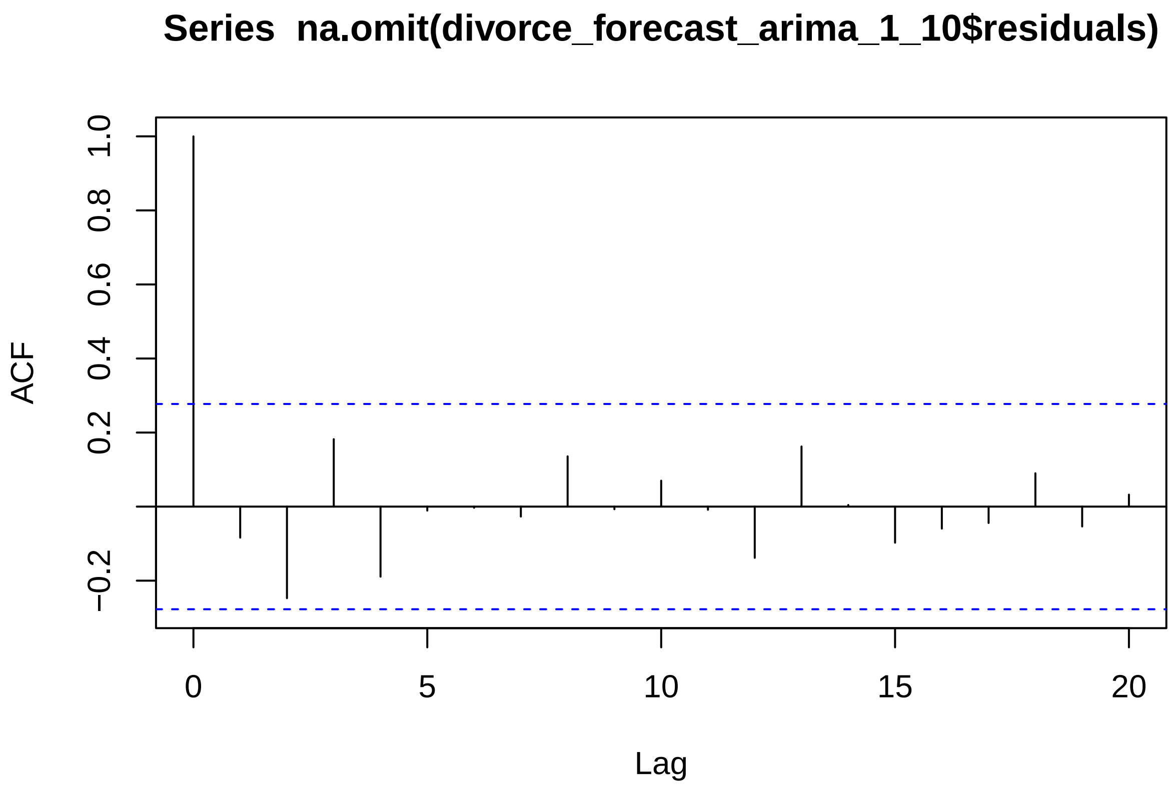
```
# plot the 10 year forecast
divorce_forecast_arima_1_10 <- forecast(divorce_arima_1, h=10, level = c(95))
plot(divorce_forecast_arima_1_10, xlab="Year", ylab="Number of Divorces")
```



The forecast also shows a steady increase in the number of divorces from 2021 to 2030

Check the forecast residuals if they are correlated

```
# plot the acf correlogram
acf(na.omit(divorce_forecast_arima_1_10$residuals), lag.max=20, na.action=na.pass)
```



The correlofram shows that there are auto correlations at any lags from 1-20 that exceed the significance bounds.

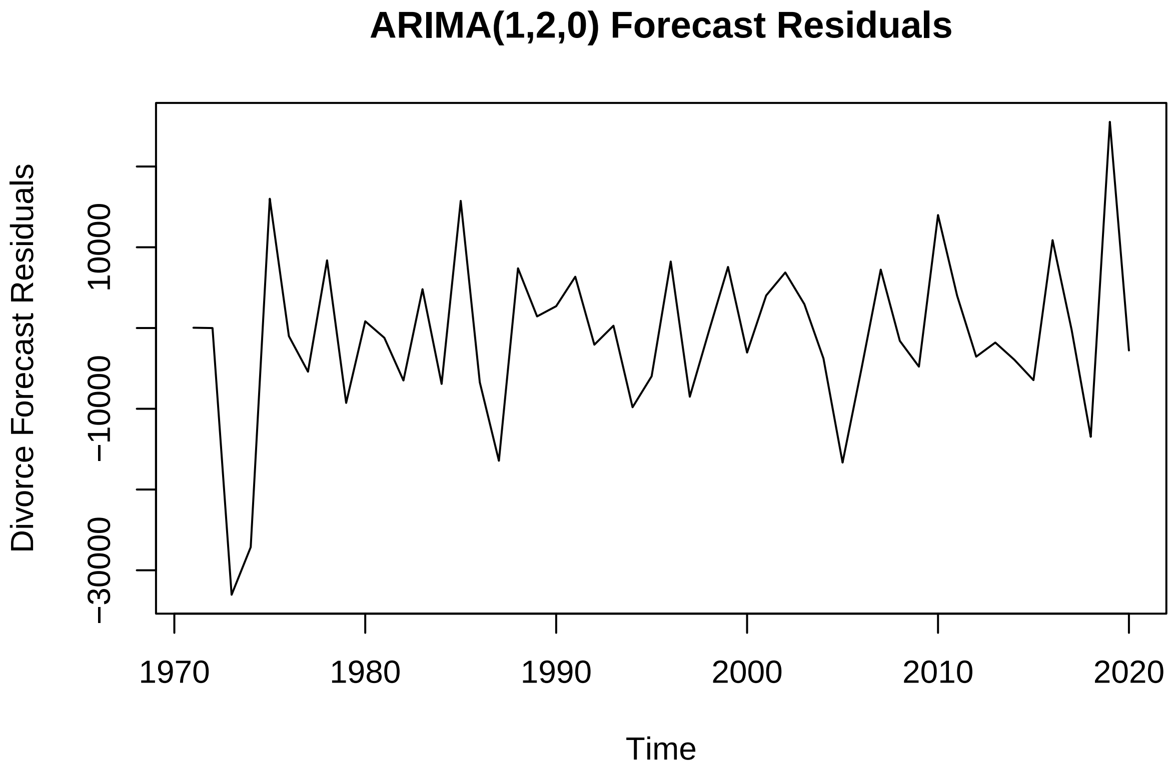
Ljung-Box test

```
# Ljung box test
Box.test(na.omit(divorce_forecast_arima_1_10$residuals), lag=20, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  na.omit(divorce_forecast_arima_1_10$residuals)
## X-squared = 14.358, df = 20, p-value = 0.8119
```

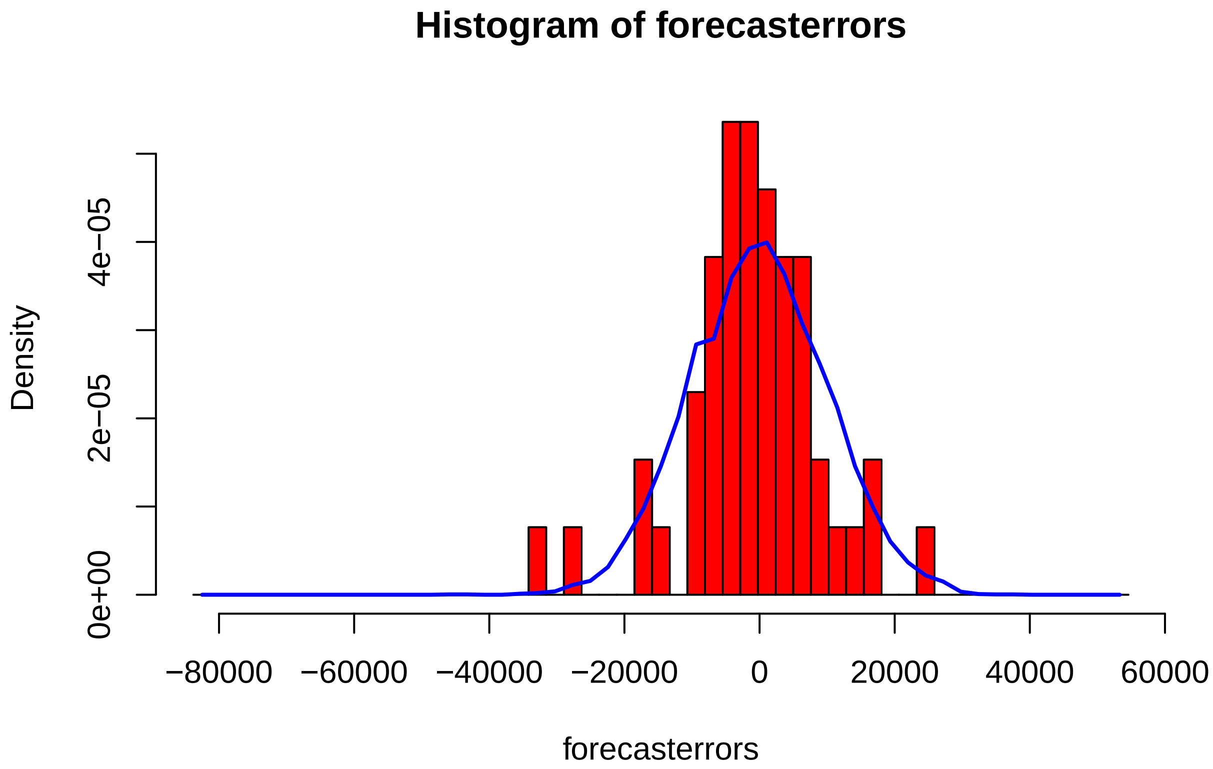
p-value = 0.8119 > 0.05, which further confirms that there are no significant auto correlations for lags 1-20.

```
# plot time series of the residuals
plot.ts(divorce_forecast_arima_1_10$residuals, ylab="Divorce Forecast Residuals",
        main="ARIMA(1,2,0) Forecast Residuals")
```



The plot of the forecast residuals show that the variance appears to be roughly constant over time.

```
# plot the histogram distribution of the forecast errors overlaid by a normal distribution curve
divorce_forecast_arima_1_10$residuals <-
  divorce_forecast_arima_1_10$residuals[!is.na(divorce_forecast_arima_1_10$residuals)]
plotForecastErrors(divorce_forecast_arima_1_10$residuals)
```



The histogram distribution of the forecast errors is roughly centered around 0 and it is approximatedly normally distributed.

Therefore, it is plausible that the forecast errors are normally distributed with mean zero and constant variance.

Since forecast errors also have no significant auto correlations, and the forecast errors appear to be normally distributed with mean zero and constant variance, the ARIMA(1,2,0) seems to provide an adequate predictive model for the number of divorces in the United Kingdom

MODEL 3: Use Auto arima to find best model

```
# use auto arima to find a model
auto.arima(divorce_ts, ic='bic')
```

```
## Series: divorce_ts
## ARIMA(1,2,1)
##
## Coefficients:
##           ar1      ma1
##       -0.5617  -0.8013
## s.e.   0.1701   0.0927
##
## sigma^2 = 78313365:  log likelihood = -504.39
## AIC=1014.78   AICc=1015.33   BIC=1020.4
```

p = 1, q = 1, d = 2

ARIMA(1,2,1)

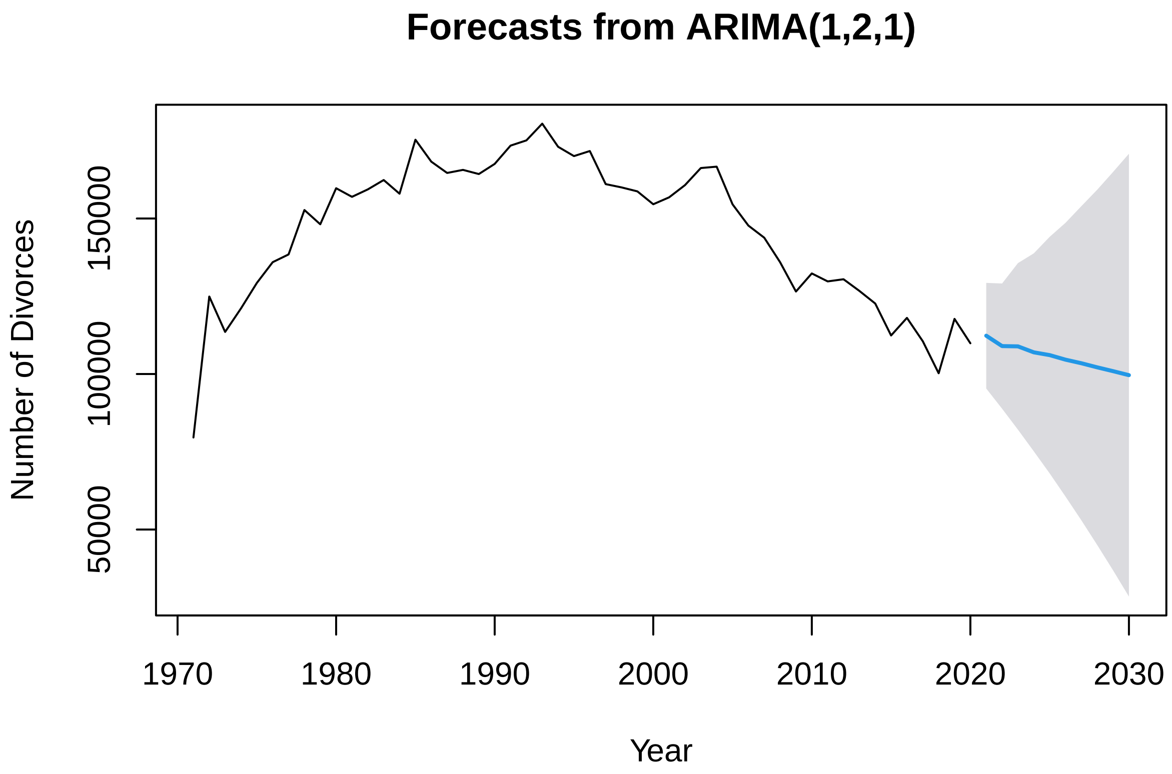
```
# forecasting with ARIMA(1,2,1)
divorce_arima_2 <- arima(divorce_ts, order = c(1,2,1))
divorce_arima_2
```

```
##
## Call:
## arima(x = divorce_ts, order = c(1, 2, 1))
##
## Coefficients:
##           ar1      ma1
##       -0.5617  -0.8013
## s.e.   0.1701   0.0927
##
## sigma^2 estimated as 75050281:  log likelihood = -504.39,  aic = 1014.78
```

beta = ar1 = -0.5617 theta = ma1 = -0.8013

10 year forecast for 95% confidence level

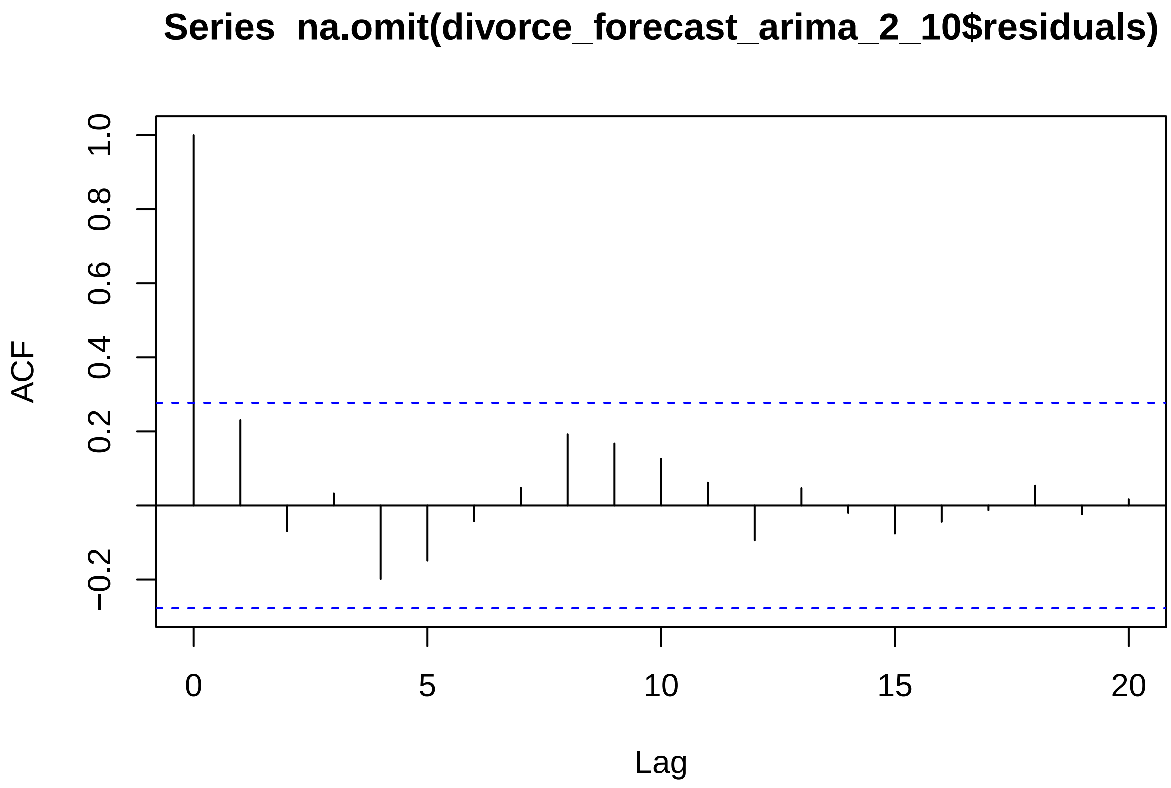
```
# plot the 10 year forecast
divorce_forecast_arima_2_10 <- forecast(divorce_arima_2, h=10, level = c(95))
plot(divorce_forecast_arima_2_10, xlab="Year", ylab="Number of Divorces")
```



The forecast shows the number of divorces decrease steadily from 2021 to 2030.

Check the forecast residuals

```r
# plot the acf correlogram
acf(na.omit(divorce_forecast_arima_2_10$residuals), lag.max=20, na.action=na.pass)
```



The correlogram shows that no auto correlations from lag 1-20 exceeds the significance bounds.

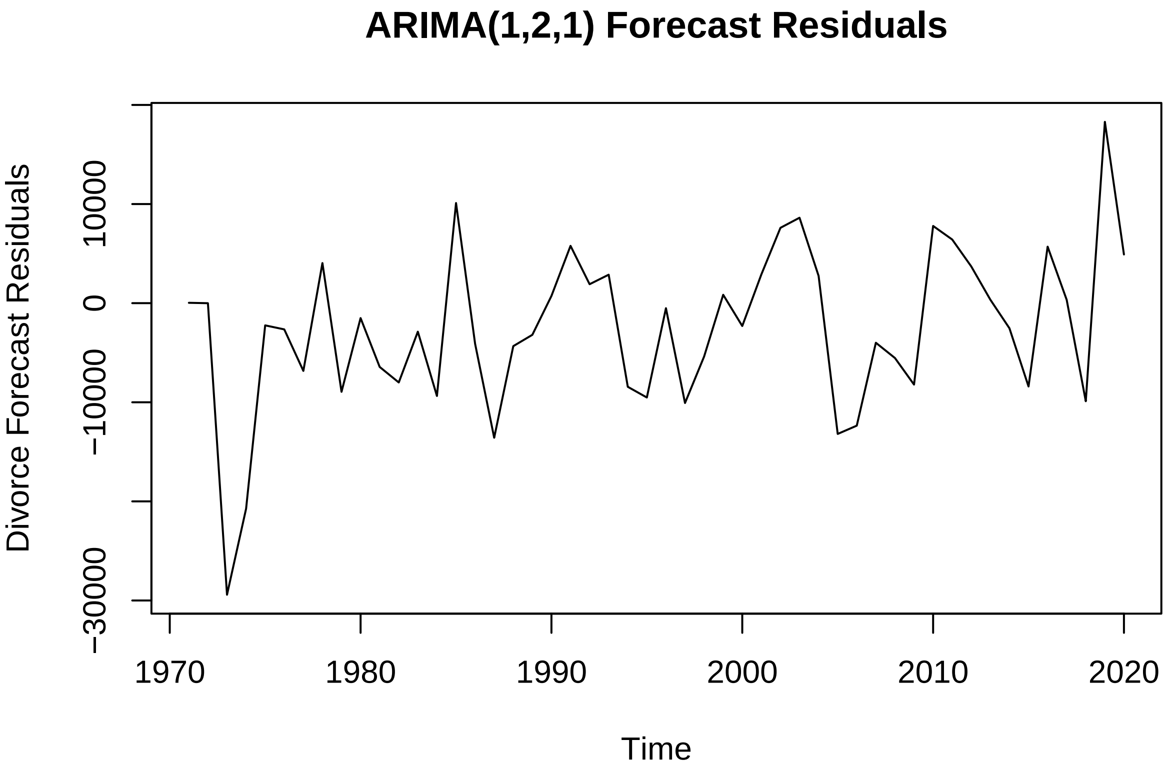
Ljung-Box test

```r
# Ljung box test
Box.test(na.omit(divorce_forecast_arima_2_10$residuals), lag=20, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  na.omit(divorce_forecast_arima_2_10$residuals)
## X-squared = 13.926, df = 20, p-value = 0.8342
```

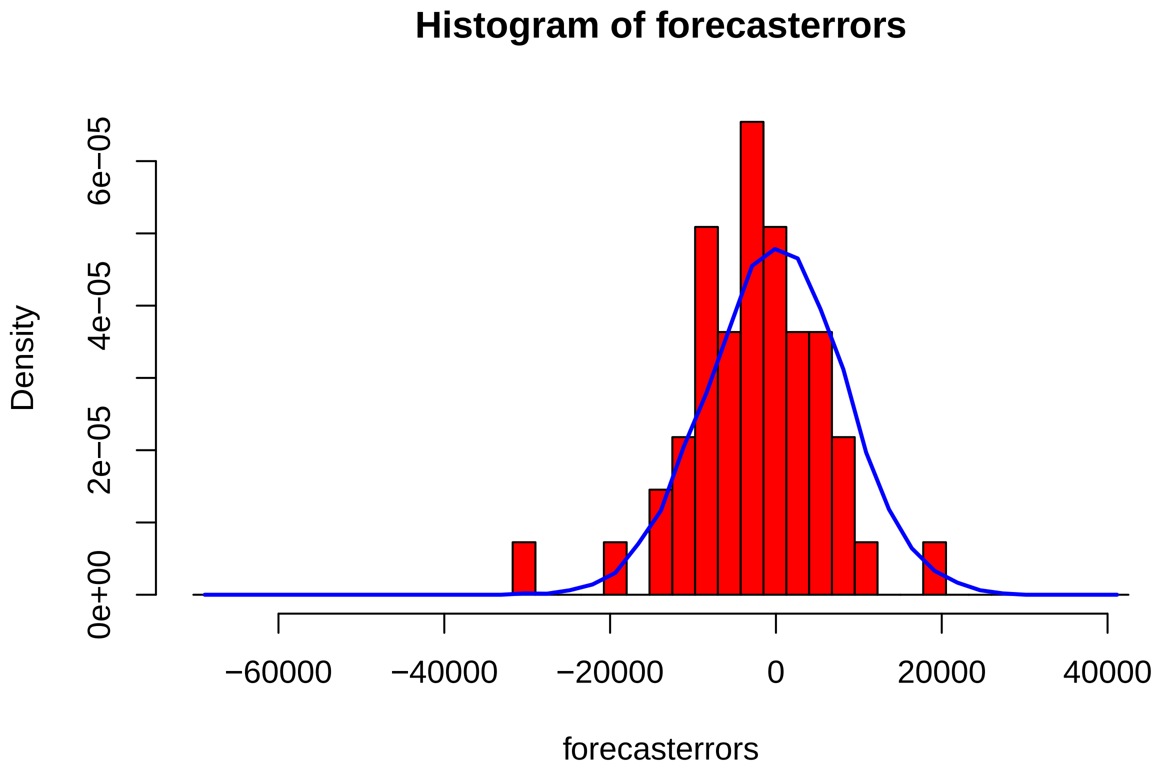
p-value = 0.8342 which is > 0.05. further confirming that there are no significant auto correlations.

```r
# plot time series of the residuals
plot.ts(divorce_forecast_arima_2_10$residuals, main="ARIMA(1,2,1) Forecast Residuals",
        ylab="Divorce Forecast Residuals")
```



The plot of the forecast residuals show that the variance appears to be roughly constant over time.

```r
# plot the histogram distribution of the forecast errors overlaid by a normal distribution curve
divorce_forecast_arima_2_10$residuals <-
  divorce_forecast_arima_2_10$residuals[!is.na(divorce_forecast_arima_2_10$residuals)]
plotForecastErrors(divorce_forecast_arima_2_10$residuals)
```



The histogram distribution of the forecast errors is roughly centered around 0 and it is approximately normally distributed.

Therefore, it is plausible that the forecast errors are normally distributed with mean zero and constant variance.

Since forecast errors also have no significant auto correlations, and the forecast errors appear to be normally distributed with mean zero and constant variance, the ARIMA(1,2,1) seems to provide an adequate predictive model for the number of divorces in the United Kingdom.

```
# Get the metrics of the ARIMA(1,2,0) model
accuracy(divorce_arima_1)
```

```
##                     ME     RMSE     MAE        MPE     MAPE     MASE
## Training set -1041.996 10106.05 7253.71 -0.9027929 5.260159 1.081668
##                     ACF1
## Training set -0.08392984
```

```
# Get the metrics of the ARIMA(1,2,1) model
accuracy(divorce_arima_2)
```

```
##                     ME     RMSE      MAE       MPE     MAPE      MASE     ACF1
## Training set -2578.179 8488.127 6405.803 -1.886425 4.603926 0.9552287 0.230466
```

ARIMA(1,2,1) has lower RMSE, MAPE and AIC so it is the better model.