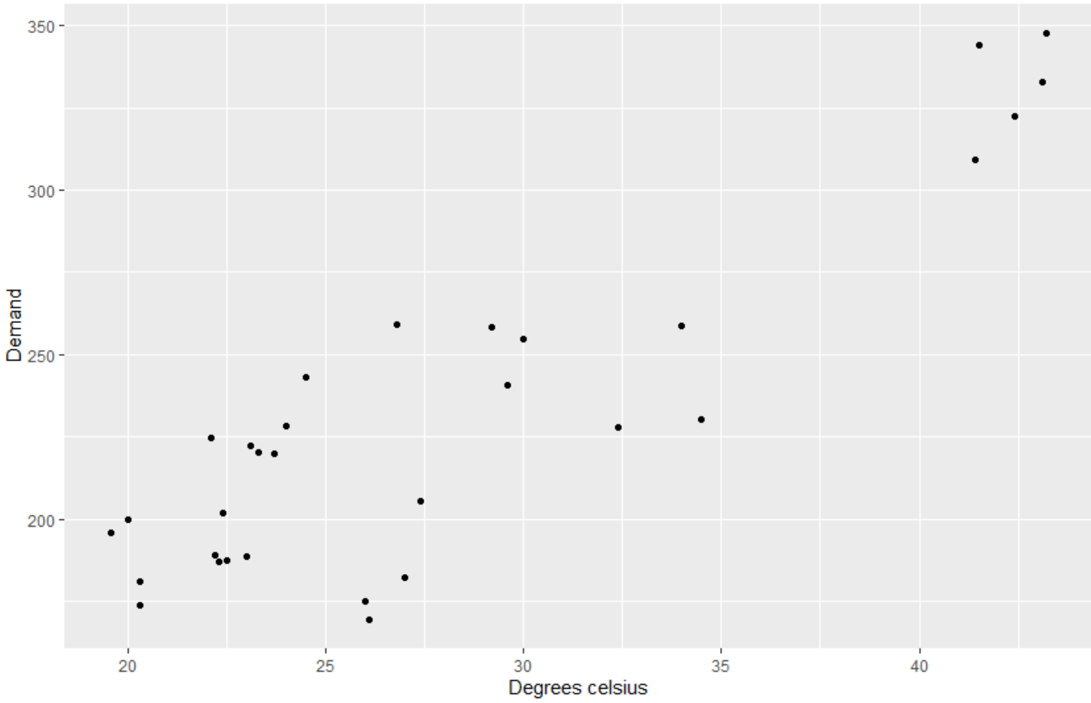
Exercise 1.

The data elecdaily, gives daily electricity demand for Victoria, Australia, during 2014, along with maximum daily temperatures in Melbourne (in degrees Celsius) and an indicator variable taking value 1 on work days, and 0 otherwise.

(i) Create a new dataset for only the month of January (i.e. the first 31 days) using the head function and create a scatter plot of Demand against Temperature.

```
help(elecdaily)

elecdaily1 <- head(elecdaily, 31)

autoplot(elecdaily)

ggplot(as.data.frame(elecdaily1), aes(x=Temperature, y=Demand))+

  geom_point()+xlab('Degrees celsius')+ylab('Demand')
```



(ii) Give a possible explanation of the relationship you see in your scatterplot.

The scatterplot indicating a clear positive linear relationship, where demand increases with temperature.

(iii) Create a simple linear regression model in R with Demand as the forecast variable and Temperature as the predictor variable. Write down the equation of the fitted model.

fit <- tslm(Demand~Temperature, data=elecdaily1)

fit

```
Call:
tslm(formula = Demand ~ Temperature, data = elecdaily1)

Coefficients:
(Intercept)  Temperature
     59.329        6.155
```

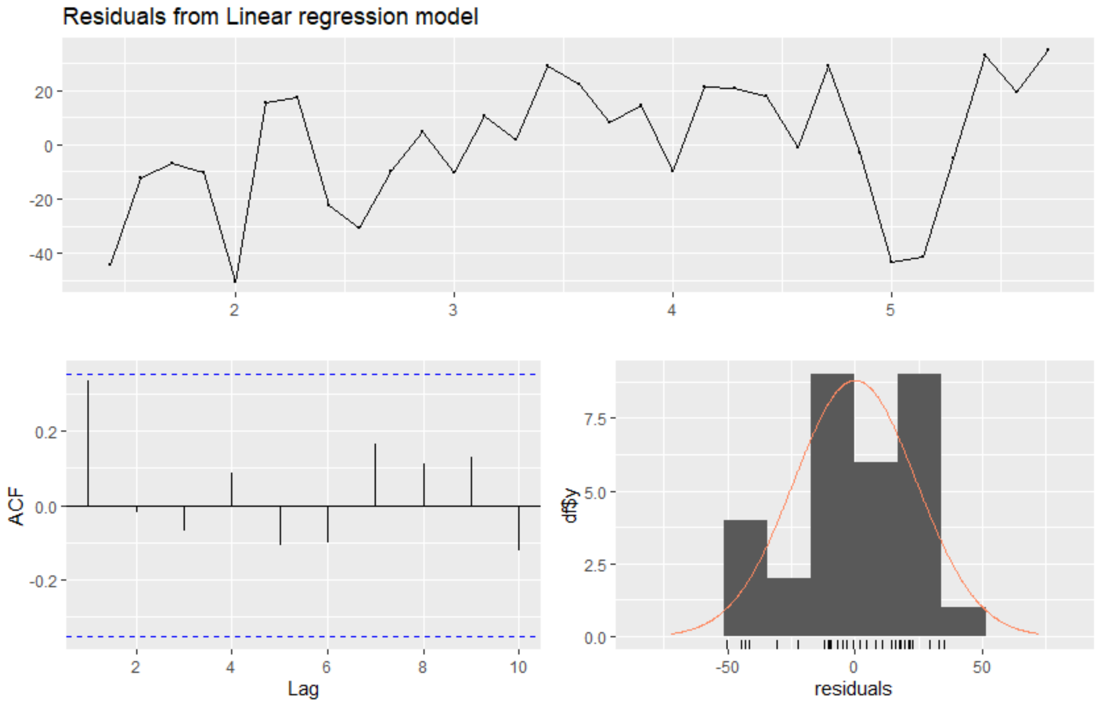
(iv) Is there any evidence of autocorrelation in the residuals? Do the residuals appear to show heteroscedasticity?

checkresiduals(fit)

All these different lags are within the dotted lines -> no evidence there

p-value is 0.3266, so not significant

So, no evidence of autocorrelation




```
        Breusch-Godfrey test for serial correlation of order up to 6

data:  Residuals from Linear regression model
LM test = 6.9378, df = 6, p-value = 0.3266
```

(v) Use your model to forecast the electricity demand if the temperature is 35 degrees or 15 degrees. Would you trust either or both of these forecasts? Explain your answer.

Hint: you can create a new data frame to use in the forecast function by using the code my.data <- data.frame(Temperature = c(15,35)).

```
my.data <- data.frame(Temperature=c(15, 35))
```

```
my.data
```

```
forecast(fit, newdata=my.data)
```

I will trust forecast the electricity demand if the temperature is 35 degrees.

```
         Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
5.857143       151.6601 117.5631 185.7571  98.48456 204.8356
6.000000       274.7677 241.7634 307.7721 223.29619 326.2392
```

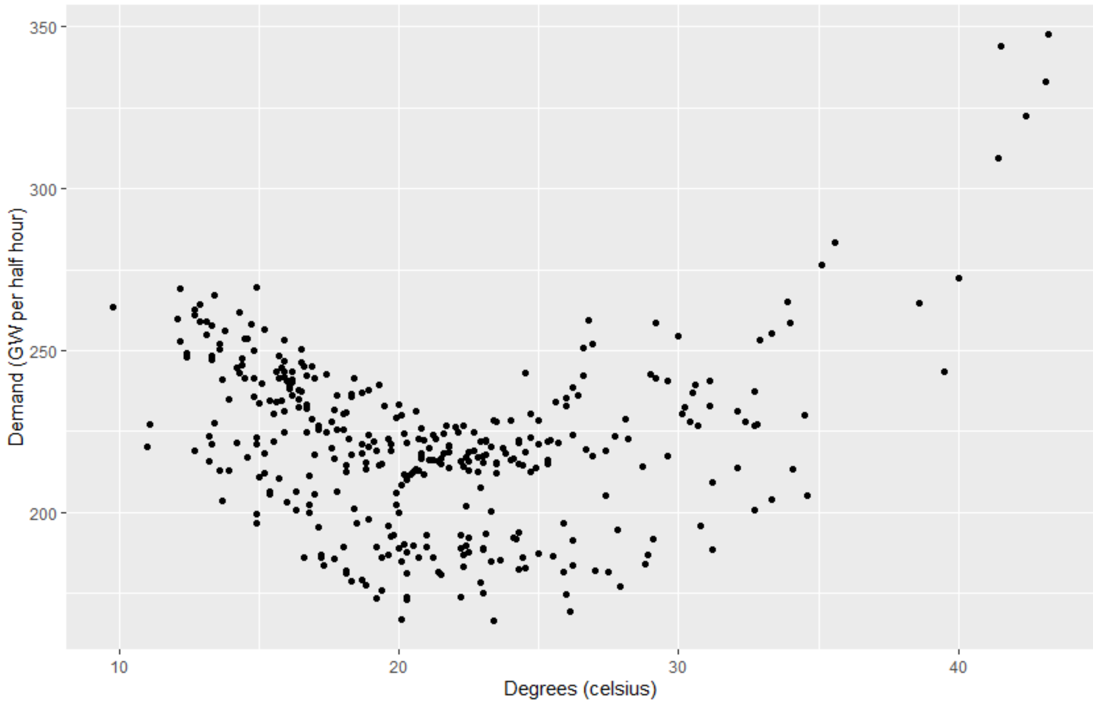
(vi) Plot Demand against Temperature for all available data in elecdaily. What does this say about your model?

```
ggplot(as.data.frame(elecdaily), aes(x=Temperature, y=Demand))+
```

```
  geom_point()+xlab('Degrees (celsius)')+ylab('Demand (GW per half hour)')
```

The data spans roughly between 23 and 43.

If I'm looking to predict between 10 and 23 degrees, it might be wise to use different models.



Exercise 2.

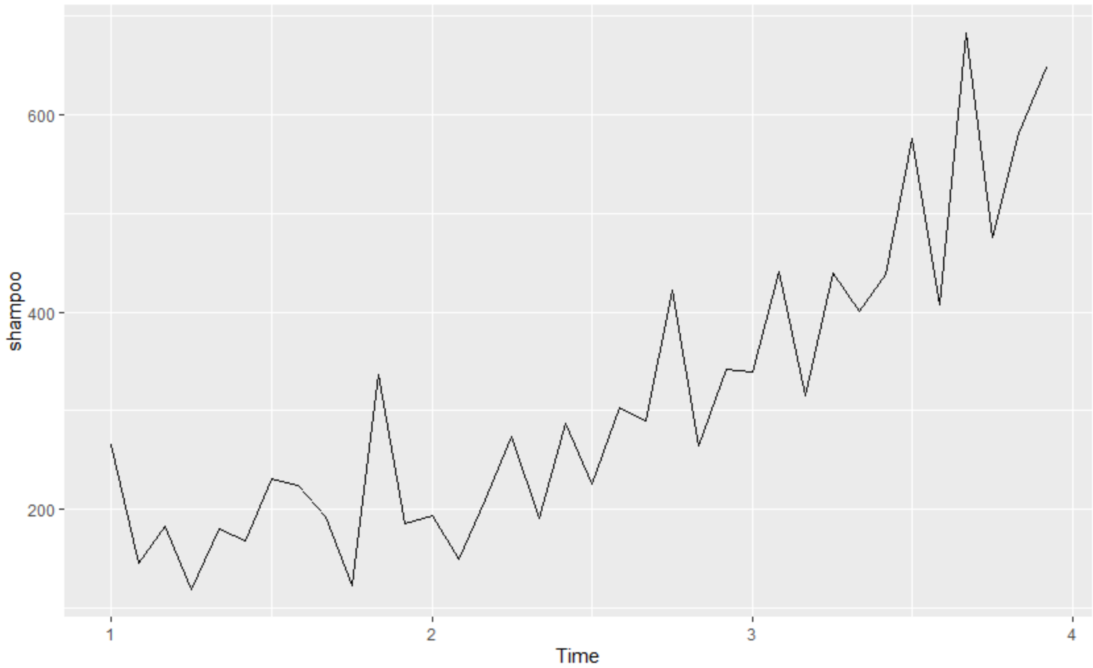
Consider the data shampoo on the sales of shampoo over a three year period.

(i) Plot the data and comment on any patterns you see.

```
help(shampoo)
```

```
shampoo
```

```
autoplot(shampoo)
```



(ii) Fit a simple linear regression model for shampoo with a linear trend predictor variable.

fit.sham <- tslm(shampoo~trend)

fit.sham

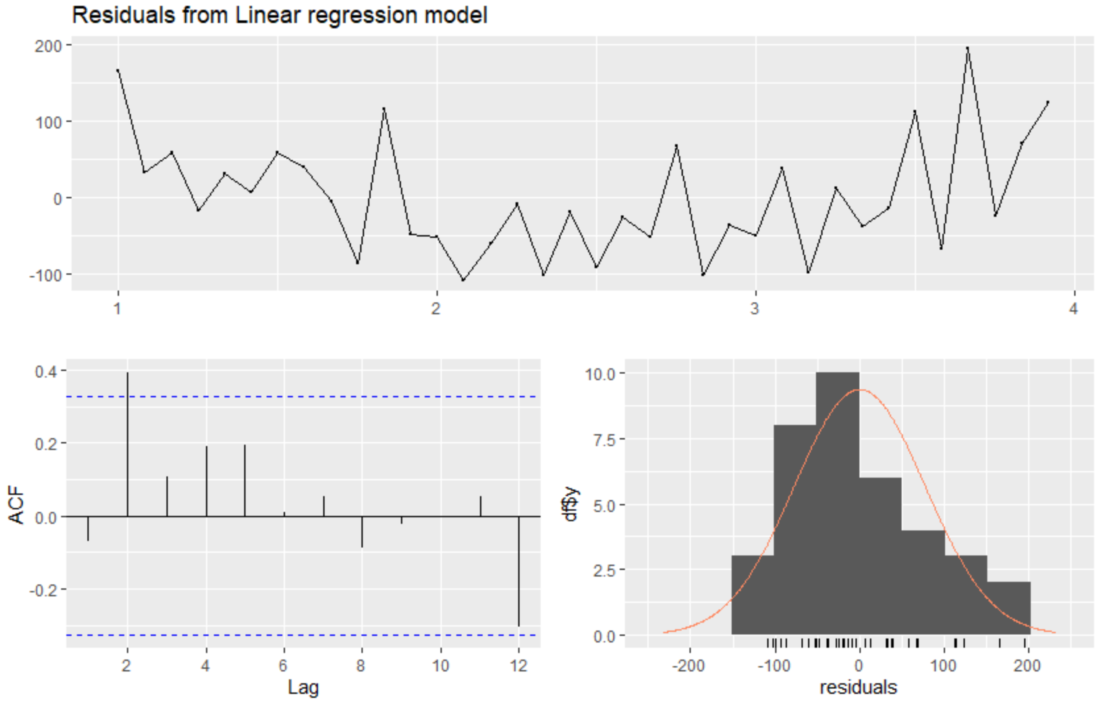
```
Call:
tslm(formula = shampoo ~ trend)

Coefficients:
(Intercept)        trend
      89.14        12.08
```

(iii) Is there evidence of autocorrelation in the residuals?

checkresiduals(fit.sham)

At the lag of 2, there seems to be a bit of autocorrelation.

p-value is 0.1769, so there isn't significant evidence at the 5% level or even a 10% level of autocorrelation.

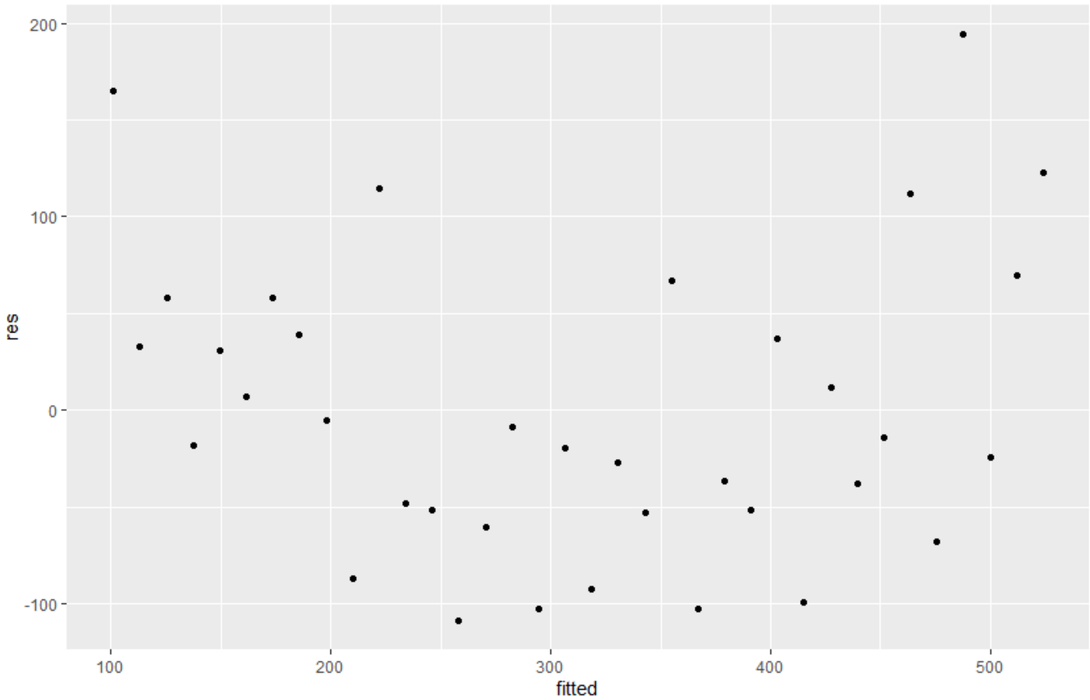



```
        Breusch-Godfrey test for serial correlation of order up to 7

data:  Residuals from Linear regression model
LM test = 10.211, df = 7, p-value = 0.1769
```

(iv) Plot the residuals against time and against the fitted values. Do the plots reveal any problems with the model?

```
ggplot(as.data.frame(cbind(res=residuals(fit.sham), fitted=fitted(fit.sham))),
       aes(x=fitted, y=res))+
  geom_point()
```

There is no heteroscedasticity.



(v) Create a forecast for shampoo sales for the next year, along with 95% and 80% prediction intervals. Plot the forecast alongside the original data. Why should you be wary about trusting the prediction intervals?

```
fc.sham <- forecast(fit.sham, h=12)
```

```
fc.sham
```

```
      Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
Jan 4       536.0629 427.6792 644.4465 367.5317 704.5940
Feb 4       548.1419 439.2843 656.9995 378.8738 717.4101
Mar 4       560.2210 450.8667 669.5753 390.1805 730.2615
Apr 4       572.3001 462.4266 682.1735 401.4524 743.1478
May 4       584.3792 473.9645 694.7938 412.6898 756.0685
Jun 4       596.4582 485.4805 707.4359 423.8935 769.0230
Jul 4       608.5373 496.9751 720.0994 435.0637 782.0109
Aug 4       620.6164 508.4486 732.7841 446.2012 795.0316
Sep 4       632.6954 519.9014 745.4895 457.3063 808.0846
Oct 4       644.7745 531.3337 758.2154 468.3797 821.1694
Nov 4       656.8536 542.7459 770.9613 479.4218 834.2854
Dec 4       668.9327 554.1384 783.7270 490.4332 847.4321
```

```
autoplot(fc.sham)
```

If the residuals aren't appearing normally distributed, especially with the left skewness, it could impact the trustworthiness of prediction intervals.

(vi) Create a multiple linear regression model for shampoo using both a linear trend predictor variable and seasonal dummy variables. Calculate the AICc for both models. Which model is better?

```
fit.sham2 <- tslm(shampoo~trend+season)
```

```
fit.sham2
```

Tells me how much I should expect the forecast variable to increase in each of these months with respect to month 1.

Because most of these are negative, that means I should expect them to go down compared to month 1.

```
Call:
tslm(formula = shampoo ~ trend + season)

Coefficients:
(Intercept)        trend      season2      season3      season4      season5      season6      season7      season8  
    113.867       11.754      -33.154      -53.808      -24.628      -56.015      -27.802        7.244      -37.043  
    season9     season10     season11     season12  
     27.536      -32.518        9.895       -4.259  
```

```
CV(fit.sham2)
```

```
CV(fit.sham)
```

```
> CV(fit.sham2)
          CV          AIC         AICc          BIC        AdjR2 
12697.952111   335.983840   355.983840   358.153105     0.633559 
> CV(fit.sham)
          CV          AIC         AICc          BIC        AdjR2 
6677.4129971  318.0898942  318.8398942  322.8404510    0.7221647 
```

The original model has smaller AICc, which smaller is better.

So, the message is, if we add all these dummy variables, it doesn't really give us a better model.

We should just stick with the original model.

Exercise 3.

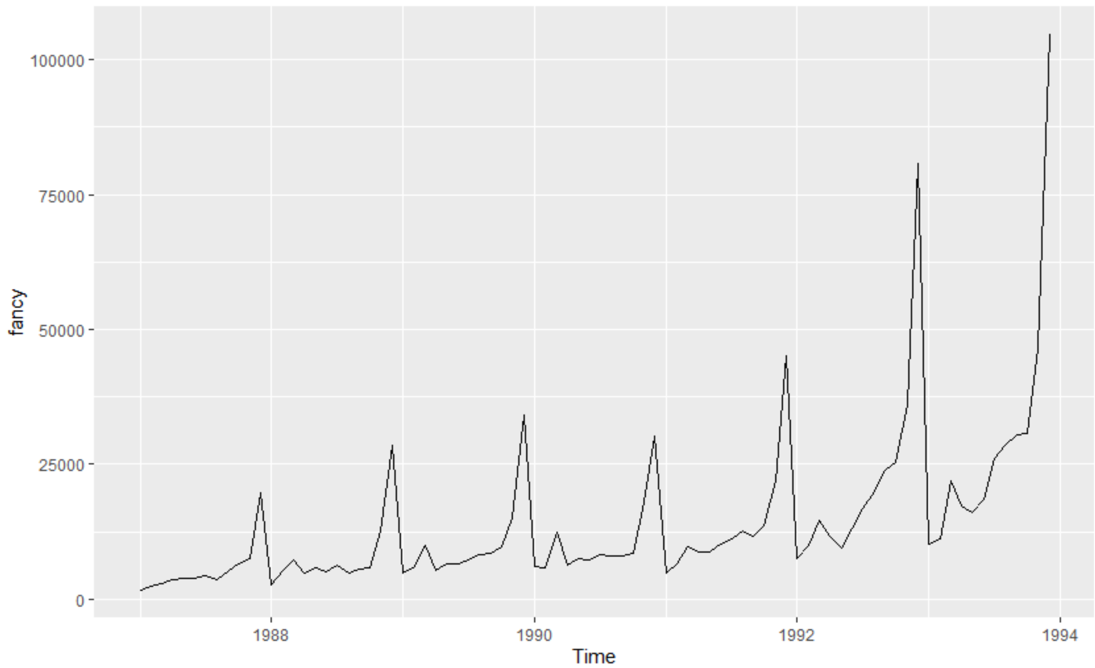
The data set fancy concerns the monthly sales figures of a shop which opened in January 1987 and sells gifts, souvenirs, and novelties. The shop is situated on the wharf at a beach resort town in Queensland, Australia. The sales volume varies with the seasonal population of tourists. There is a large influx of visitors to the town at Christmas and for the local surfing festival, held every March since 1988. Over time, the shop has expanded its premises, range of products, and staff.

(i) Produce a time plot of the data and describe the patterns in the graph. Identify any unusual or unexpected fluctuations in the time series.

```
autoplot(fancy)
```

Very strong seasonality.

The peak is in the end of each year, also have an upward trend and heteroscedasticity, as the variance is increasing as time goes on.



(ii) Explain why it is appropriate to take logarithms of these data before fitting a model.

When I have variance that seems to depend on the level of the data, that's when it's appropriate to take algorithms.

And also, because it's financial data, that would be another reason why it's appropriate to take algorithms.

(iii) Use R to fit a regression model to the logarithms of these sales data with a linear trend, seasonal dummies and a “surfing festival" dummy variable.

Hint: the “surfing festival" dummy variable can be created using the following code.

> festival <- cycle(fancy) == 3

> festival[3] == 0

fancy2 <- log(fancy)

autoplot(fancy2)

festival <- cycle(fancy)==3

festival

```
       Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
1987 FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
1988 FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
1989 FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
1990 FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
1991 FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
1992 FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
1993 FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

festival[3] <- 0

festival

```
     Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1987   0   0   0   0   0   0   0   0   0   0   0   0
1988   0   0   1   0   0   0   0   0   0   0   0   0
1989   0   0   1   0   0   0   0   0   0   0   0   0
1990   0   0   1   0   0   0   0   0   0   0   0   0
1991   0   0   1   0   0   0   0   0   0   0   0   0
1992   0   0   1   0   0   0   0   0   0   0   0   0
1993   0   0   1   0   0   0   0   0   0   0   0   0
```

fit.surf <- tslm(fancy2~trend+season+festival)

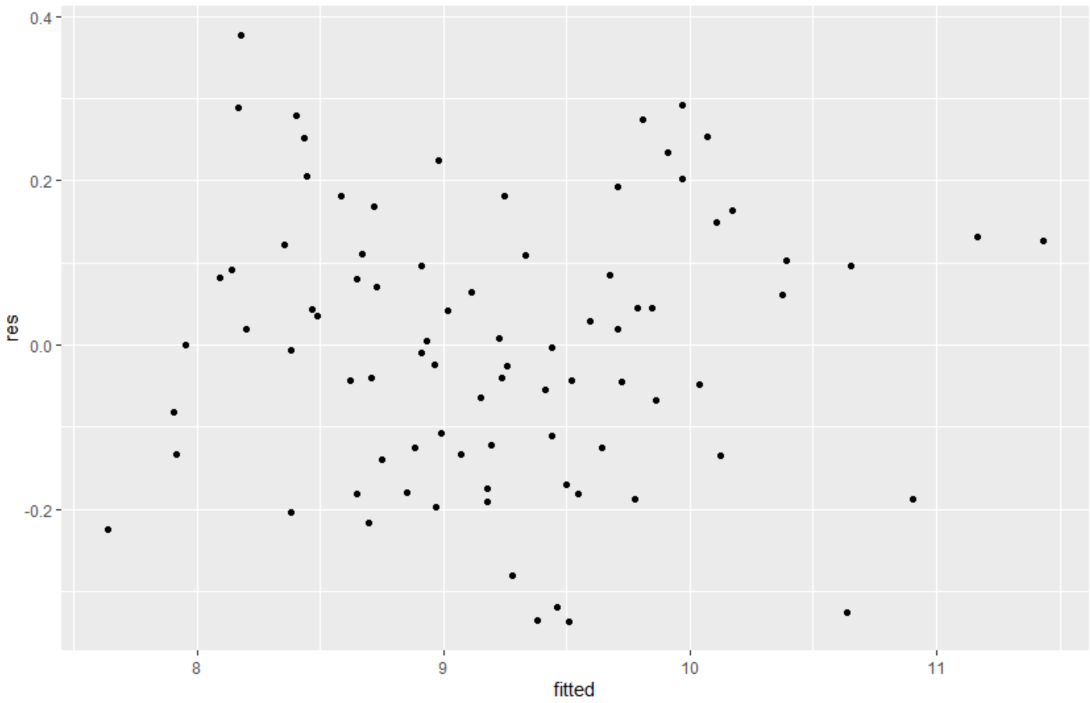
fit.surf

```
Call:
tslm(formula = fancy2 ~ trend + season + festival)

Coefficients:
(Intercept)        trend      season2      season3      season4      season5      season6      season7      season8
    7.61967      0.02202      0.25142      0.26608      0.38405      0.40949      0.44883      0.61045      0.58796
    season9     season10     season11     season12     festival
    0.66933      0.74739      1.20675      1.96224      0.50152
```

(iv) Plot the residuals against time and against the fitted values. Do these plots reveal any problems with the model?

```
ggplot(as.data.frame(cbind(res=residuals(fit.surf), fitted=fitted(fit.surf))), aes(x=fitted, y=res))+
  geom_point()
```

I don't see any obvious correlation between the fitted values and the residuals. (that's good)



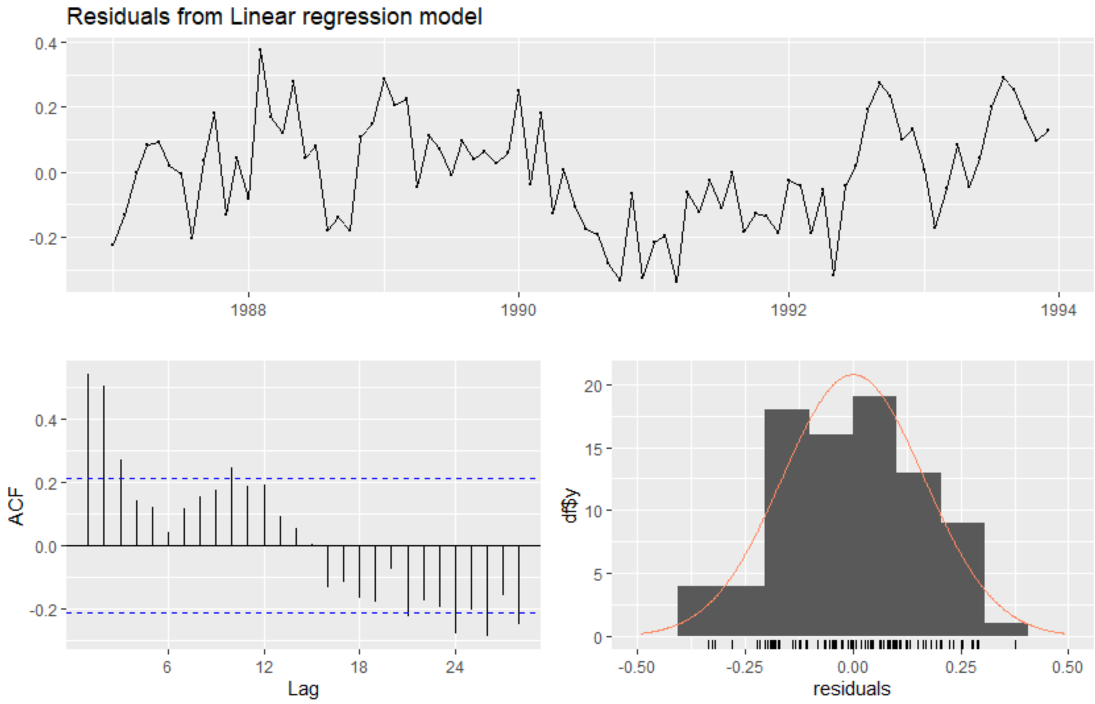
(v) Perform a Breusch-Godfrey test. What does it tell you?

`checkresiduals(fit.surf)`

There's any heteroscedasticity, that's good.

In ACF plot, the lags of 1, 2, and 3 months, got pretty high autocorrelation. (not great)

Having autocorrelation, it doesn't mean the model is wrong, it just means the model could be better somehow.




```
	Breusch-Godfrey test for serial correlation of order up to 17

data:  Residuals from Linear regression model
LM test = 37.954, df = 17, p-value = 0.002494
```

(vi) Notwithstanding your answers to the questions above, create a forecast for monthly sales data in 1994. You will need to produce new data for the dummy variable to use in the forecast. One way to do this is to use the following code.

```
> future.festival <- rep(0, 12)

> future.festival[3] <- 1
```

```
future.festival <- rep(0, 12)

future.festival
```

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

```
future.festival[3] <- 1

future.festival
```

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

```
new.data <- data.frame(festival=future.festival)

fc.surf <- forecast(fit.surf, newdata=new.data, h=12)

fc.surf
```

```
         Point Forecast     Lo 80     Hi 80     Lo 95    Hi 95
Jan 1994       9.491352  9.238522  9.744183  9.101594  9.88111
Feb 1994       9.764789  9.511959 10.017620  9.375031 10.15455
Mar 1994      10.302990 10.048860 10.557120  9.911228 10.69475
Apr 1994       9.941465  9.688635 10.194296  9.551707 10.33122
May 1994       9.988919  9.736088 10.241749  9.599161 10.37868
Jun 1994      10.050280  9.797449 10.303110  9.660522 10.44004
Jul 1994      10.233926  9.981095 10.486756  9.844168 10.62368
Aug 1994      10.233456  9.980625 10.486286  9.843698 10.62321
Sep 1994      10.336841 10.084010 10.589671  9.947083 10.72660
Oct 1994      10.436923 10.184092 10.689753 10.047165 10.82668
Nov 1994      10.918299 10.665468 11.171129 10.528541 11.30806
Dec 1994      11.695812 11.442981 11.948642 11.306054 12.08557
```

(vii) Transform the forecast to obtain predictions for the original (untransformed) data.

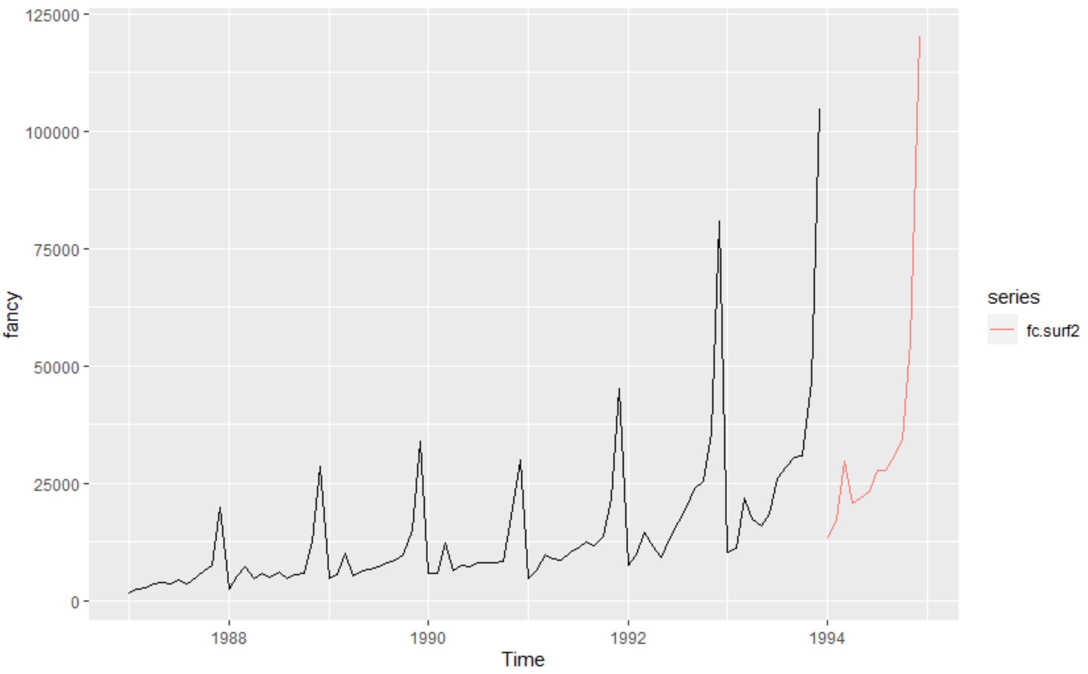
Hint: for a forecast fc, you can extract the predictions using the code fc$mean.

```
fc.surf2 <- exp(fc.surf$mean)
```

```
fc.surf2
```

```
           Jan       Feb       Mar       Apr       May       Jun       Jul       Aug       Sep       Oct       Nov
1994  13244.70  17409.81  29821.65  20774.16  21783.73  23162.27  27831.56  27818.48  30848.42  34095.57  55176.84
           Dec
1994 120067.79
```

```
autoplot(fancy)+autolayer(fc.surf2)
```



(viii) Recall that applying a log transformation to a time series is equivalent to using a Box-Cox transformation with λ = 0. Check your answer to part (vii) by using the tslm function on the original (untransformed) data and specifying λ = 0.

```
fit.surf2 <- tslm(fancy~trend+season+festival, lambda=0)
```

```
fit.surf2
```

```
Call:
tslm(formula = fancy ~ trend + season + festival, lambda = 0)

Coefficients:
(Intercept)        trend      season2      season3      season4      season5      season6      season7      season8
    7.61967      0.02202      0.25142      0.26608      0.38405      0.40949      0.44883      0.61045      0.58796
    season9     season10     season11     season12     festival
    0.66933      0.74739      1.20675      1.96224      0.50152
```

```
fc.surf3 <- forecast(fit.surf2, newdata=new.data, h=12)
```

```
fc.surf3
```

```
         Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
Jan 1994       13244.70 10285.82  17054.73  8969.583  19557.43
Feb 1994       17409.81 13520.45  22418.00 11790.284  25707.73
Mar 1994       29821.65 23129.40  38450.24 20155.412  44123.68
Apr 1994       20774.16 16133.21  26750.16 14068.696  30675.62
May 1994       21783.73 16917.24  28050.15 14752.395  32166.37
Jun 1994       23162.27 17987.81  29825.24 15685.969  34201.95
Jul 1994       27831.56 21613.98  35837.72 18848.111  41096.73
Aug 1994       27818.48 21603.82  35820.87 18839.249  41077.41
Sep 1994       30848.42 23956.87  39722.43 20891.193  45551.50
Oct 1994       34095.57 26478.61  43903.67 23090.230  50346.32
Nov 1994       55176.84 42850.31  71049.28 37366.903  81475.41
Dec 1994      120067.79 93244.59 154607.08 81312.400 177294.90
```

Exercise 4.

Consider the time series writing, which shows the industry sales for printing and writing paper (in thousands of French francs) from 1968 to 1977.

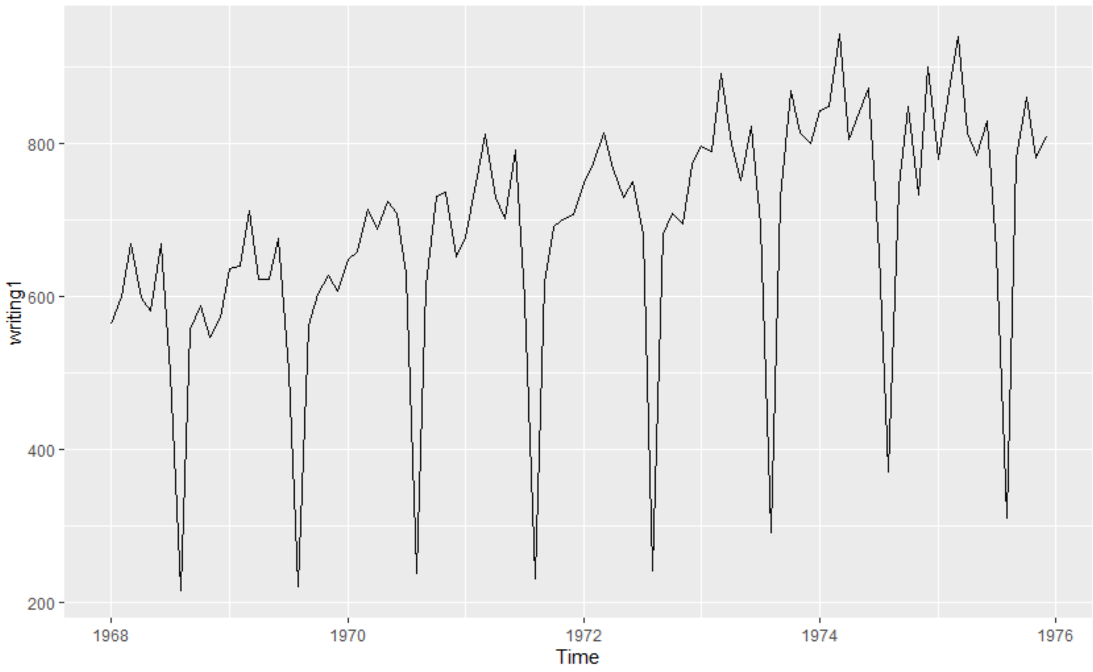
(i) Split the data into a training set from the beginning of 1968 to the end of 1975 and a test set from the beginning of 1976 to the end of 1977.

```
writing

writing1 <- window(writing, end=c(1975, 12))

writing2 <- window(writing, start=c(1976, 1))

writing1

writing2
```

```
> writing1
         Jan     Feb     Mar     Apr     May     Jun     Jul     Aug     Sep     Oct     Nov     Dec
1968 562.674 599.000 668.516 597.798 579.889 668.233 499.232 215.187 555.813 586.935 546.136 571.111
1969 634.712 639.283 712.182 621.557 621.000 675.989 501.322 220.286 560.727 602.530 626.379 605.508
1970 646.783 658.442 712.906 687.714 723.916 707.183 629.000 237.530 613.296 730.444 734.925 651.812
1971 676.155 748.183 810.681 729.363 701.108 790.079 594.621 230.716 617.189 691.389 701.067 705.777
1972 747.636 773.392 813.788 766.713 728.875 749.197 680.954 241.424 680.234 708.326 694.238 772.071
1973 795.337 788.421 889.968 797.393 751.000 821.255 691.605 290.655 727.147 868.355 812.390 799.556
1974 843.038 847.000 941.952 804.309 840.307 871.528 656.330 370.508 742.000 847.152 731.675 898.527
1975 778.139 856.075 938.833 813.023 783.417 828.110 657.311 310.032 780.000 860.000 780.000 807.993
> writing2
          Jan      Feb      Mar      Apr      May      Jun      Jul      Aug      Sep      Oct      Nov      Dec
1976  895.217  856.075  893.268  875.000  835.088  934.595  832.500  300.000  791.443  900.000  781.729  880.000
1977  875.024  992.968  976.804  968.697  871.675 1006.852  832.037  345.587  849.528  913.871  868.746  993.733
```

(ii) Plot the sales data for the training set. Propose an appropriate regression model based on the patterns you see in the plot.

autoplot(writing1)



fit.writing <- tslm(writing1~trend+season)

fit.writing

```
Call:
tslm(formula = writing1 ~ trend + season)

Coefficients:
(Intercept)        trend      season2      season3      season4      season5      season6      season7      season8
    589.981        2.804       25.361       94.936        8.262       -5.587       39.367     -113.587     -465.646
    season9     season10     season11     season12
    -73.442        1.095      -35.250      -14.861
```

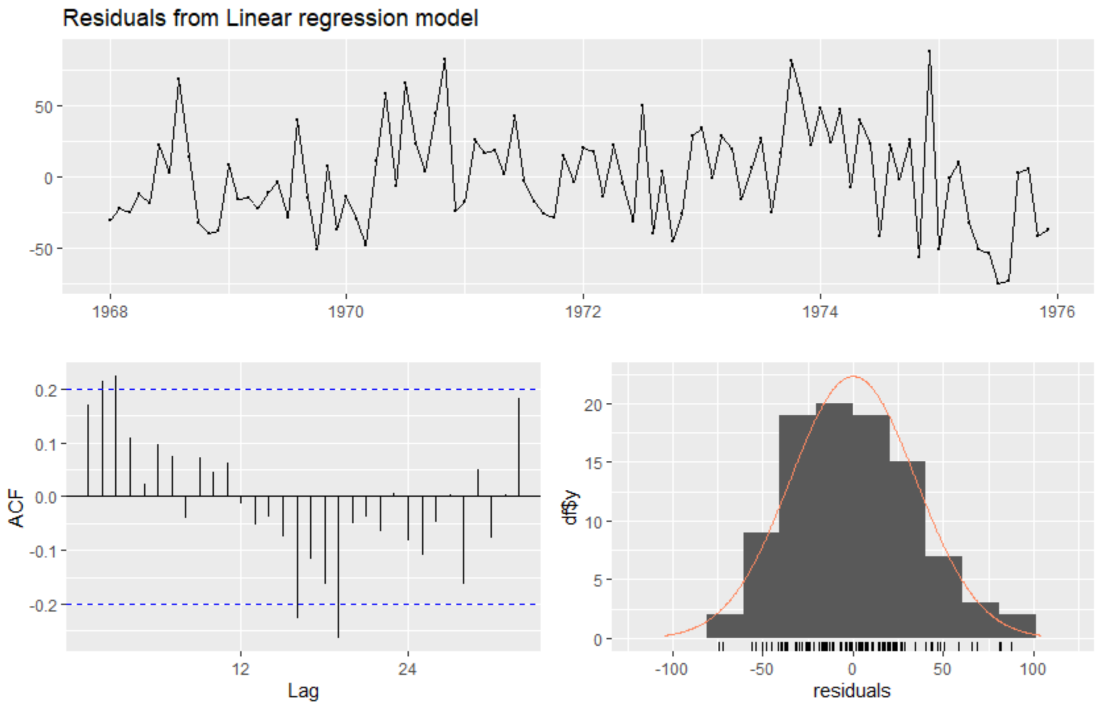
(iii) For your chosen model, check for autocorrelation and seasonality in the residuals. Do the residuals appear to be normally distributed?

```
checkresiduals(fit.writing)
```

There's no heteroscedasticity observed in the plot of residuals against time.

The presence of a few points just outside the blue dotted lines may not be a significant concern, especially given the non-significant p-value (0.2011) from the series test for autocorrelation.

Additionally, the observation of residuals appearing somewhat normally distributed is a favorable aspect.



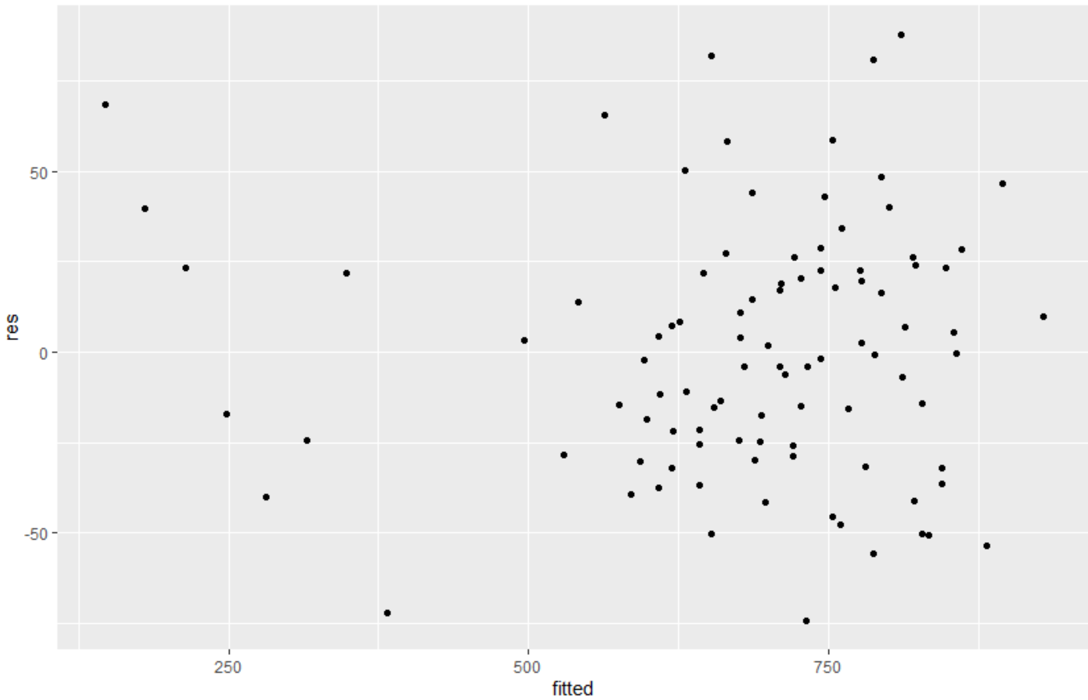

```
        Breusch-Godfrey test for serial correlation of order up to 19

data:  Residuals from Linear regression model
LM test = 23.871, df = 19, p-value = 0.2011
```

(iv) Plot the residuals against time and against the fitted values. Do these plots reveal any problems with the model?

```
ggplot(as.data.frame(cbind(res=residuals(fit.writing), fitted=fitted(fit.writing))), aes(x=fitted, y=res))+
  geom_point()
```

Can't see any clear signs of a trend, so that's good.

And that covers question (iii) and (iv), so no problems with the model.



(v) Create a forecast for 1976 and 1977 using your model, and plot it alongside the full data set from 1968 to 1977.

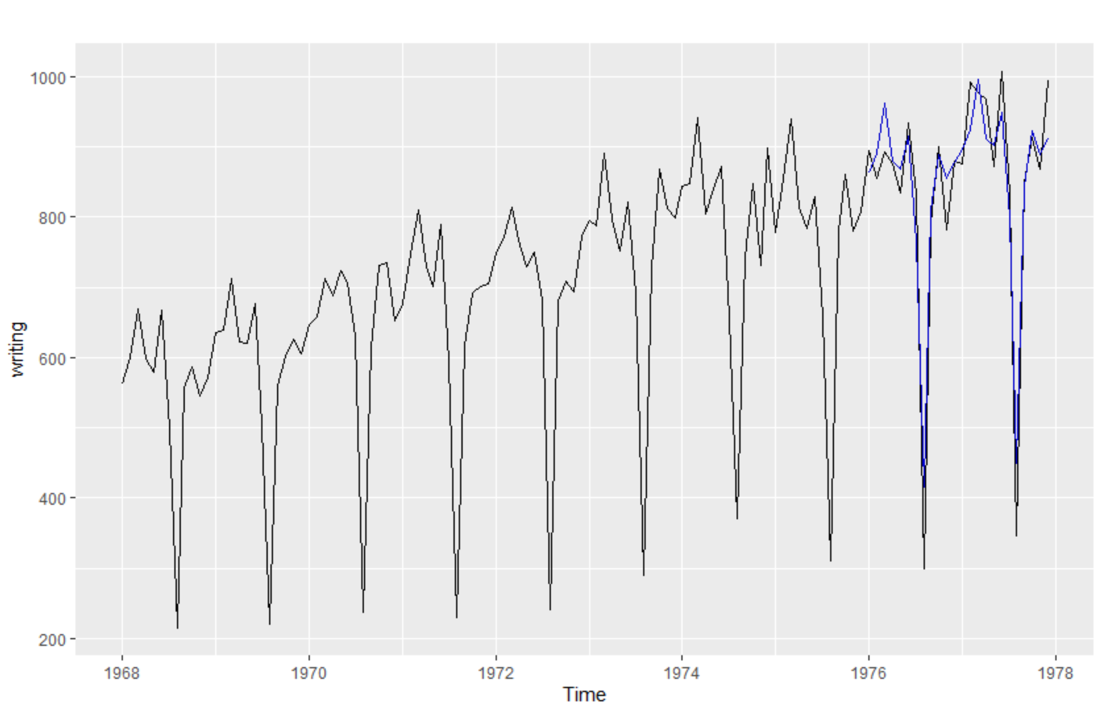
```
fc.writing <- forecast(fit.writing, h=24)
```

```
fc.writing
```

```
         Point Forecast    Lo 80     Hi 80    Lo 95     Hi 95
Jan 1976       861.9835 810.1056  913.8614 782.1103  941.8567
Feb 1976       890.1487 838.2709  942.0266 810.2755  970.0220
Mar 1976       962.5275 910.6496 1014.4054 882.6543 1042.4007
Apr 1976       878.6580 826.7801  930.5359 798.7848  958.5312
May 1976       867.6132 815.7354  919.4911 787.7400  947.4865
Jun 1976       915.3710 863.4931  967.2489 835.4978  995.2442
Jul 1976       765.2211 713.3433  817.0990 685.3479  845.0943
Aug 1976       415.9665 364.0886  467.8444 336.0933  495.8397
Sep 1976       810.9750 759.0971  862.8529 731.1018  890.8482
Oct 1976       888.3156 836.4378  940.1935 808.4424  968.1888
Nov 1976       854.7755 802.8976  906.6534 774.9023  934.6487
Dec 1976       877.9686 826.0908  929.8465 798.0954  957.8418
Jan 1977       895.6333 843.3156  947.9510 815.0829  976.1837
Feb 1977       923.7986 871.4809  976.1163 843.2482 1004.3490
Mar 1977       996.1773 943.8596 1048.4950 915.6269 1076.7277
Apr 1977       912.3078 859.9901  964.6255 831.7574  992.8582
May 1977       901.2631 848.9454  953.5808 820.7127  981.8135
Jun 1977       949.0208 896.7031 1001.3385 868.4704 1029.5712
Jul 1977       798.8710 746.5533  851.1887 718.3206  879.4214
Aug 1977       449.6163 397.2986  501.9340 369.0659  530.1667
Sep 1977       844.6248 792.3071  896.9425 764.0744  925.1752
Oct 1977       921.9655 869.6478  974.2832 841.4151 1002.5159
Nov 1977       888.4253 836.1076  940.7430 807.8749  968.9757
Dec 1977       911.6185 859.3008  963.9362 831.0681  992.1689
```

```
autoplot(writing)+autolayer(fc.writing, PI=FALSE)
```

The forecast looks accurate.



(vi) Check the accuracy of the forecast by comparing the errors to the standard deviation of the forecast variable in the test set. How well does your model perform?

accuracy(fc.writing, writing2)

```
                        ME     RMSE      MAE        MPE     MAPE      MASE       ACF1 Theil's U
Training set -8.881784e-16 34.59253 28.09114 -0.1398254 4.732063 0.6169491  0.1707892        NA
Test set     -4.683735e+00 51.82569 41.10254 -2.5317986 6.426572 0.9027108 -0.2566030 0.1235056
```

sd(writing2)

```
[1] 172.161
```

A test data size of 172, significantly larger than the RMSE of 51.82, indeed supports the notion of a robust forecast.

It's a good forecast