BAN430 Forecasting
Project 1

Bergen
March 20, 2019

Forecasting Project 1

Group: 2

Candidates: 75 68

NORWEGIAN SCHOOL OF ECONOMICS

Candidates: 75 68

Table of Contents

Statistics Summary **3**

Decompose the series **4**

Components Forecasting **5**

ETS models **7**

tsCV() Function **10**

R-code cross validation 11

References **13**

Candidates: 75 68

Statistics Summary

Our dataset contains 459 observations of one quantitative variable. This time-series data starts from 1957-07 and ends at 1995-08. First, we took a quick look at the data to check if there is any missing values or meaningless values (for example, negative values in monthly production). We observe there is a missing value for monthly production in the last row. We remove it after identified it does not belong to the dataset. Second, we look at the summary of this dataset (Table 1). The median value is 4724 while the Mean value is 5151. We can see from the histogram that this variable has a skewed distribution.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1066    3554    4724    5151    6480   11095
```

Table1: summary of dataset

Furthermore, we plot the whole series, ACF and PACF. As we can see from the plot, this series follows a robust upward trend while also has a strong seasonal effect. The 36 lags are significantly different from zero, which means it may be nonstationary. It took a long time for a shock to die out. There are spikes at lag 12, lag 24 and lag36, indicating seasonality.

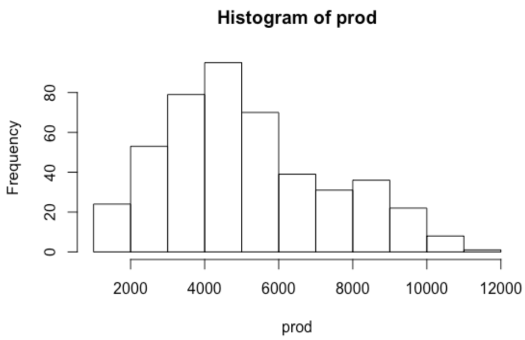

Figure1: Histogram of dataset

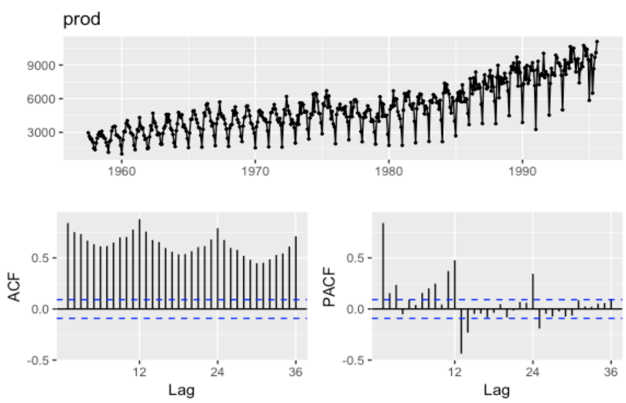

Figure2: Plot and correlogram of dataset

Decompose the series

Here we use `stl()` decomposition to decompose the series. We choose STL decomposition method because it has many advantages comparing other methods; for example, it allows a changing seasonal component and can handle different periods of seasonal effect. So since, as we can see from the plotted time-series, its seasonality is changing over time, the STL decomposition method would be a good decomposition method here.

The `t.window` and `s.window` arguments control the period to extract trend or seasonal features. If the numbers we put into the `t.window` and `s.window` are too small, the decomposed component may have the risk of overfitting. In contrast, it may be too smoothing and omit some of the features if the number is too big. In this case, after trying out several combinations of `t.window` and `s.window`, we choose `t.window` equal to 21 and `s.window` as 7, which allows a trend that is approximately linear while also describe business cycle to a reasonable extent. We can see from the trend plot that the series has a steeper slope since 1985 than the years before, which is correspond to the series. It also shows a flat development at the beginning of the 90s. An `s.window` equal to 7 allows the seasonal component changing over time while not including unnecessary data bump. The remainder component is quite steady and remains small at the beginning period while its volatility.

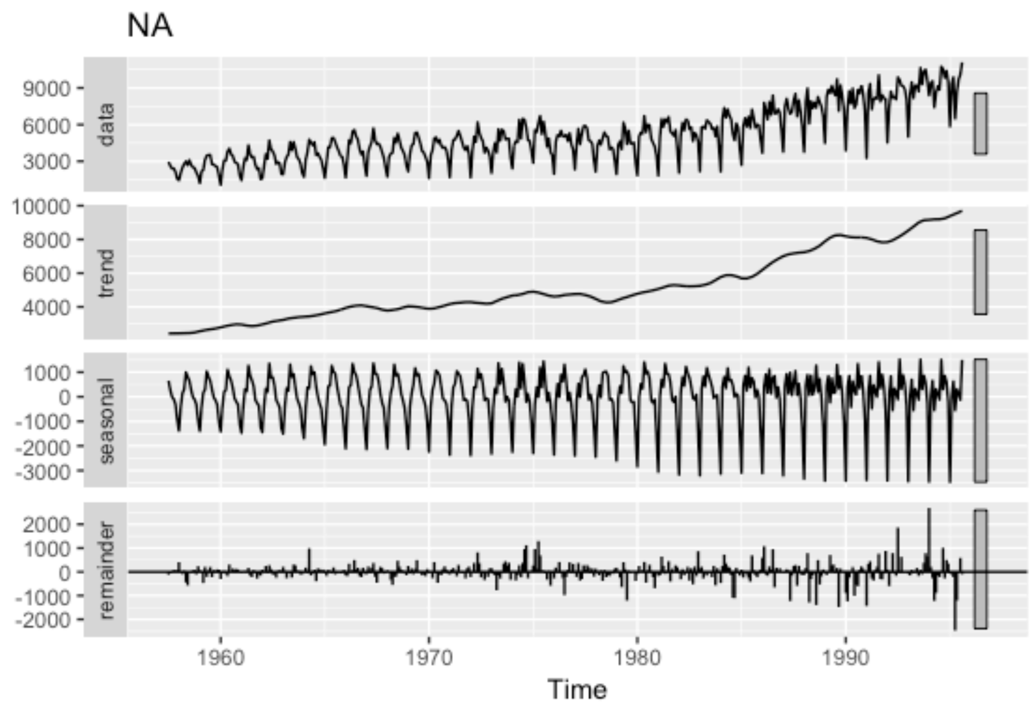

Figure 3: Using STL decompose the series

Components Forecasting

Here we use `stlf` function to forecast the series using its components. We use the same decomposition method as previously. `stlf` function helps us to forecast seasonally adjusted data and then seasonalized it. First, we use the default method ETS, which will be further explained in the next question, to forecast the seasonal adjusted data. Then we use the naive method to forecast the seasonally adjusted series and seasonalized it as the seasonal component is the same as last year.

As we can see from the plot, in general, the forecasted results of both methods are very similar. Both methods have pretty well forecasting of the test data. Both methods predicted a significant drop in 1995 and a small peak afterwards while have some deviation with the actual chocolate production.

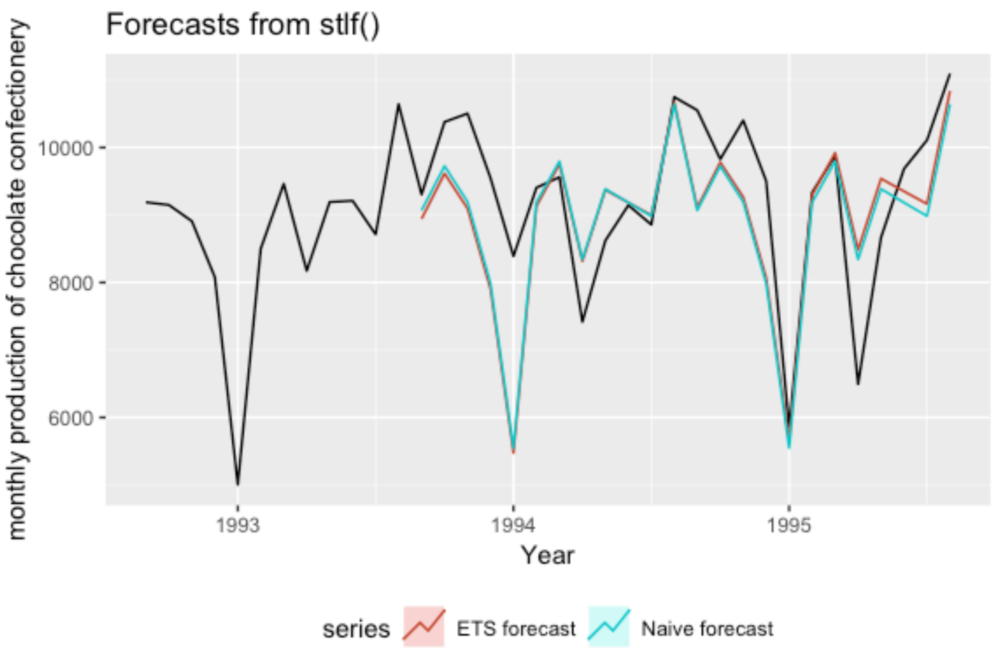

Figure 4: Using stlf() forecast the series

```
##                  ME    RMSE    MAE   MPE MAPE MASE ACF1 Theil's U
## Training set   5.06  400.89 283.23 -0.56 5.95 0.56 0.01        NA
## Test set     348.31 1054.13 758.76  3.06 8.56 1.49 0.48      0.55
```

Table 2: Fitted result for stlf() with ETS method

```
##                  ME    RMSE    MAE   MPE MAPE MASE  ACF1 Theil's U
## Training set  15.29  500.05 352.81 -0.24 7.33 0.69 -0.45        NA
## Test set     382.89 1041.90 771.11  3.46 8.66 1.51  0.46      0.54
```

Table 3: Fitted result for stlf() with naive method

As we have used `stlf()` function to forecast, we used naive and ETS method to forecast seasonally adjusted data and seasonalize it. However, we wonder if there is a better way to forecast each component using different ways according to the series` features?

Here we extract seasonal, trend and remainder components from the STL decomposition and forecast them using different methods. We use `snaive()` function to forecast seasonal component since the seasonal component display a significant seasonal feature. Using `snaive` allows the forecasting to duplicate the data from a seasonal period before. Then we use a random walk with drift model to forecast the trend component since it displays an upward leaning. As for the remainder, stationarity of the series can be observed from the plot. Therefore, we choose to fit it in an ARMA model then use the `forecast()` function to forecast the model.

After forecasting each component, we form the forecasting of observations by integrate the three components forecasting addictively and use test dataset to test its performance. The result turns out quite good. This one has a slightly smaller test RMSE than the two methods using `stlf()` function. Also, the test ME, MAE MPE and other criteria of this method, all are better than the results of the previous two forecasting.

```
##                 ME    RMSE    MAE MPE MAPE ACF1 Theil's U
## Test set 206.35 1018.43 734.68 1.5 8.36 0.49      0.54
```

Table 4: Test result for components forecasting

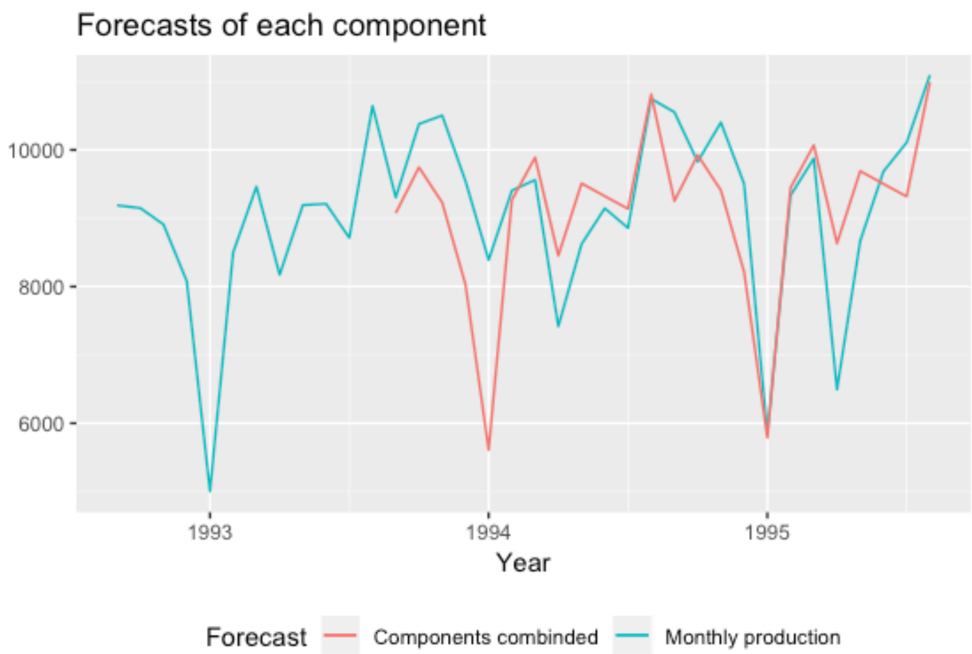

Figure 5: Compare component forecasting with actual data

ETS models

In the ETS model, the three letters stand for Error, Trend and Seasonal separately, in which we can choose from Error{A,M}, Trend{N,A,M} and Seasonal{N,A,M}.

At first, "The point forecasts produced by a model with additive errors and one with multiplicative errors are identical if they use the same smoothing parameter values" *(Rob and George, Forecasting: Principles and Practice)*, however, the model with additive errors has more narrow prediction intervals compared to that with multiplicative errors. As we can see from the figure below, more observations are within the prediction intervals in the model with multiplicative errors than in the model with additive errors. What's more, "models with multiplicative errors are useful when the data are strictly positive" *(Rob and George, Forecasting: Principles and Practice)*, which matches the time series of interest because the monthly productions are all positive. Thus, we can determine that the error type is multiplicative.

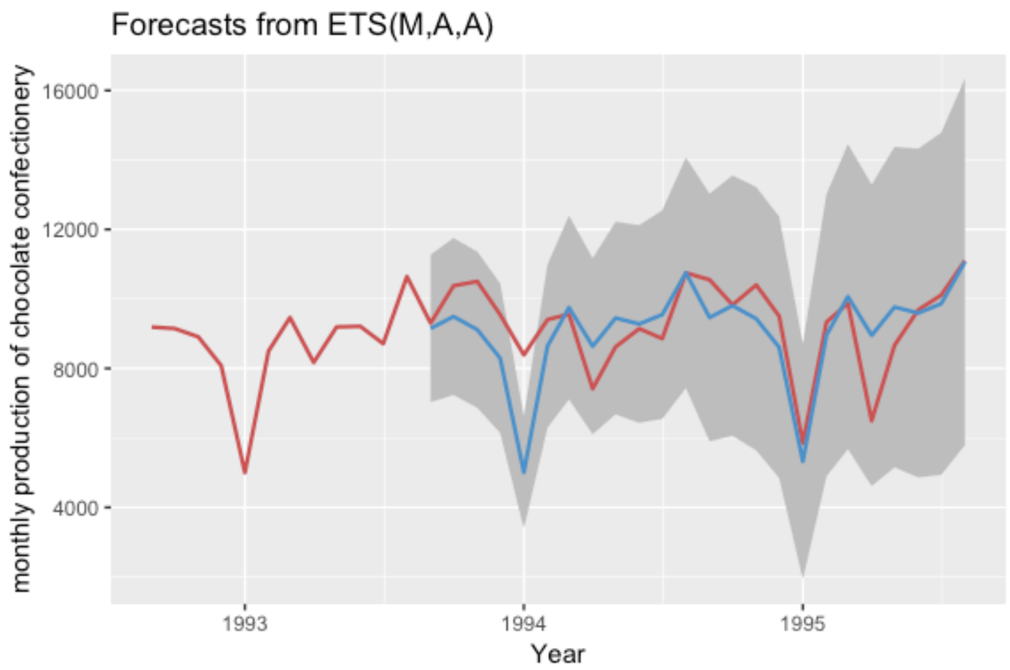

Figure 6: Forecasting from ETS(M,A,A)

Candidates: 75 68

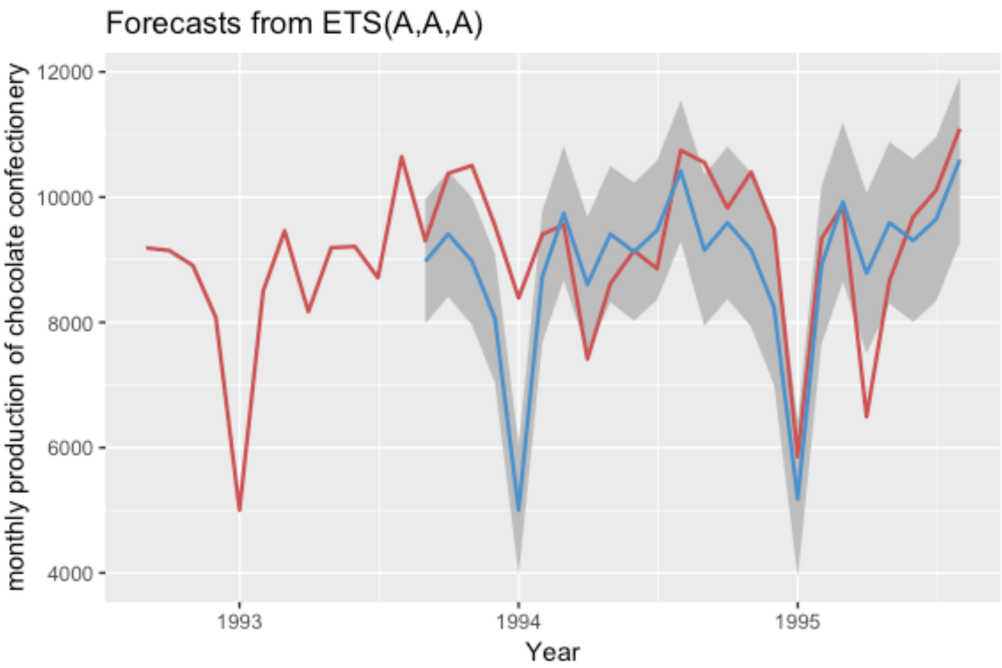

Figure 7: Forecasting from ETS(A,A,A)

Furthermore, because the multiplicative trend methods tend to produce poor forecasts (Rob and George, Forecasting: Principles and Practice) and the series has a strong trend, we can't choose trend type of "N" or "M". Therefore, the second letter of the ETS model Trend=A.

Finally, as we described in the first section, the series has a strong seasonal effect. What's more, the seasonal variations are changing proportional to the level of the series, so the multiplicative seasonal method is preferred (Rob and George, Forecasting: Principles and Practice). Hence, we choose multiplicative season type.

In summary, we select ETS model {M, A, M}, with either damped or non-damped trends. After comparing the accuracy of both methods in test data, ETS {M, A, M} with non-damped trends has better forecast accuracy. It is consistent with the automatically selected model by the `ets()` function.

```
##                   ME    RMSE    MAE  MPE   MAPE MASE ACF1 Theil's U
## Training set   32.30  495.15 374.24 0.05   7.80 0.73 0.07        NA
## Test set      568.45 1267.27 983.73 5.68  11.25 1.93 0.49      0.67
```

Table 5: Fitted result for ETS models {M, A, M} with damped trend

```
##                   ME    RMSE    MAE   MPE   MAPE MASE ACF1 Theil's U
## Training set   -9.61  494.10 376.29 -0.86   7.90 0.74 0.07        NA
## Test set      415.05 1227.79 909.77  4.02  10.57 1.79 0.51      0.65
```

Table 6: Fitted result for ETS models {M, A, M} with non-damped trend

Candidates: 75 68

```
## ETS(M,A,M)
##
## Call:
##  ets(y = train, model = "ZZZ")
##
##   Smoothing parameters:
##     alpha = 0.1813
##     beta  = 1e-04
##     gamma = 0.2697
##
##   Initial states:
##     l = 2318.3175
##     b = 13.6833
##     s = 1.2683 1.3408 1.1432 1.0554 0.8611 0.5303
##            0.6579 0.8658 0.9657 0.9797 1.1047 1.2271
##   sigma:  0.1006
##
##      AIC     AICc      BIC
## 7964.236 7965.707 8033.478
```

Table 7: Automatically selected model

Then we use it to forecast the observations for the period 1993:M9 to 1995:M8 and plot the variable and forecasts.

```
##                   ME    RMSE    MAE   MPE  MAPE MASE ACF1 Theil's U
## Training set   -9.61  494.10 376.29 -0.86  7.90 0.74 0.07        NA
## Test set      415.05 1227.79 909.77  4.02 10.57 1.79 0.51      0.65
```

Table 8: Fitted result for chosen model

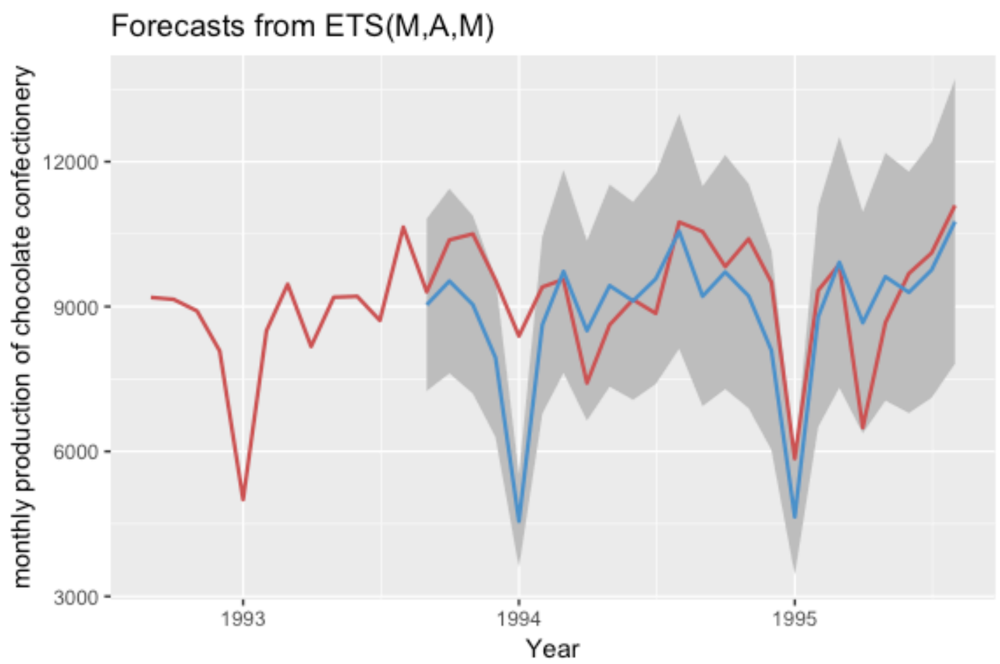

Figure 8: Forecasting of the chosen model

tsCV() Function

At first, we use `tsCV()` function to perform a time series cross-validation in the particular case: 1) forecast two monthes, 2) start at date 1993: M8 and forecast 1993: M10 and so on, 3) use the seasonal naive forecasting method. It returns a matrix with two columns containing forecast errors for forecast horizon 1 and 2. The row name is time t, the first column is prediction f[t+1] and the second column is prediction f[t+2]. The forecast errors are calculated by the actual value y[t+h]-f[t+h].

We use the `head()` function to look at the first 13 rows of the `tsCV()` result and find that the first 11 rows return NAs. Because seasonal naive method make predictions to be equal to the last observation from the same month of the previous year, it returns NA until t is 1958: M6 so that prediction f[t+1] and f[t+2] equal to the value of 1957: M7 and 1957: M8 separately, which are the first two observations of the series. What's more, we also find the second column of each row is identical to the first column of next row, because ft+2 is the same as f(t+1)+1 when using snaive method.

```
##           h=1 h=2
## Jul 1957  NA  NA
## Aug 1957  NA  NA
## Sep 1957  NA  NA
## Oct 1957  NA  NA
## Nov 1957  NA  NA
## Dec 1957  NA  NA
## Jan 1958  NA  NA
## Feb 1958  NA  NA
## Mar 1958  NA  NA
## Apr 1958  NA  NA
## May 1958  NA  NA
## Jun 1958 145  73
## Jul 1958  73 113
```

Table 9: head(cv,n=13L)

We also use the `tail()` function to look at the last few rows and find out there are NAs in the last two rows. Because we don't have observations in 1995: M9 and 1995: M10, it can't calculate forecast errors on these two months.

```
##            h=1  h=2
## Mar 1995 -922   47
## Apr 1995   47  540
## May 1995  540 1251
## Jun 1995 1251  347
## Jul 1995  347   NA
## Aug 1995   NA   NA
```

Table 10: tail(cv)

Since we are only interested in the last two years' predictions, we use `window()` function to make a subset of the two-step ahead forecasts errors and will compare it with the result from the R-code wrote by ourselves. We also compute the RMSE based on the forecast's errors for comparison.

```
##             h=1   h=2
## Aug 1993   115  1230
## Sep 1993  1230  1595
## Oct 1993  1595  1467
## Nov 1993  1467  3386
## Dec 1993  3386   903
## Jan 1994   903    99
## Feb 1994    99  -756
## Mar 1994  -756  -572
## Apr 1994  -572   -66
## May 1994   -66   145
## Jun 1994   145   105
## Jul 1994   105  1248
## Aug 1994  1248  -552
## Sep 1994  -552  -102
## Oct 1994  -102   -44
## Nov 1994   -44 -2537
## Dec 1994 -2537   -71
## Jan 1995   -71   314
## Feb 1995   314  -922
## Mar 1995  -922    47
## Apr 1995    47   540
## May 1995   540  1251
## Jun 1995  1251   347
```

Table 11: The subset we need to make comparison

R-code cross validation

Then we write R-code to reproduce the results of `tsCV()` that we obtained above. The kernel is to write a for loop to apply the forecast function `snaive()` to subsets of the time series using a rolling forecast origin. The for loop will run m=length(test)-(h-1) times, in which test is the last two years in this case, so `length(test)` is 24. The reason why m must be less than `length(test)` is the lack of observations to compute forecasts errors for the last (h-1) times. After each iteration of the loop, the training sets will add one observation while the test sets will move forward in the time horizon.

In order to compute forecasts errors, we need to get predictions first, then subtract the actual value with the prediction to get the forecast error. We create the forecasts errors matrix with the same format of the forecasts errors matrix obtained with `tsCV()`. Then we use `identical()` function to compare the two matrices and get the result True, indicating the results of `tsCV()` are identical with the results of R-code. And for sure the RMSEs are also the same.

```
##            h=1   h=2
## Aug 1993   115  1230
## Sep 1993  1230  1595
## Oct 1993  1595  1467
## Nov 1993  1467  3386
## Dec 1993  3386   903
## Jan 1994   903    99
## Feb 1994    99  -756
## Mar 1994  -756  -572
## Apr 1994  -572   -66
## May 1994   -66   145
## Jun 1994   145   105
## Jul 1994   105  1248
## Aug 1994  1248  -552
## Sep 1994  -552  -102
## Oct 1994  -102   -44
## Nov 1994   -44 -2537
## Dec 1994 -2537   -71
## Jan 1995   -71   314
## Feb 1995   314  -922
## Mar 1995  -922    47
## Apr 1995    47   540
## May 1995   540  1251
## Jun 1995  1251   347

## [1] TRUE

## [1] 1153.908

## [1] 1153.908

## [1] TRUE
```

Table 12: The forecast errors of each run

References

Hyndman, R.J., & Athanasopoulos, G. (2018) Forecasting: principles and practice, 2nd edition, OTexts: Melbourne, Australia. OTexts.com/fpp2. Accessed on 19.March.2019

Petropoulos F, Razbash S, Wang E, Yasmeen F (2019). _forecast: Forecastingfunctions for time series and linearmodels_. R package version 8.5, <URL:http://pkg.robjhyndman.com/forecast>.

Hyndman RJ, Khandakar Y (2008). "Automatic time series forecasting: the forecast package for R." _Journal of Statistical Software_, *26*(3), 1-22. <URL:http://www.jstatsoft.org/article/view/v027i03>.

Adrian Trapletti and Kurt Hornik (2018). tseries: Time Series Analysis and Computational Finance. R package version 0.10-46.