Week #11: Concepts of Time Series

Cheng Peng

4/7/2021

Table of Contents

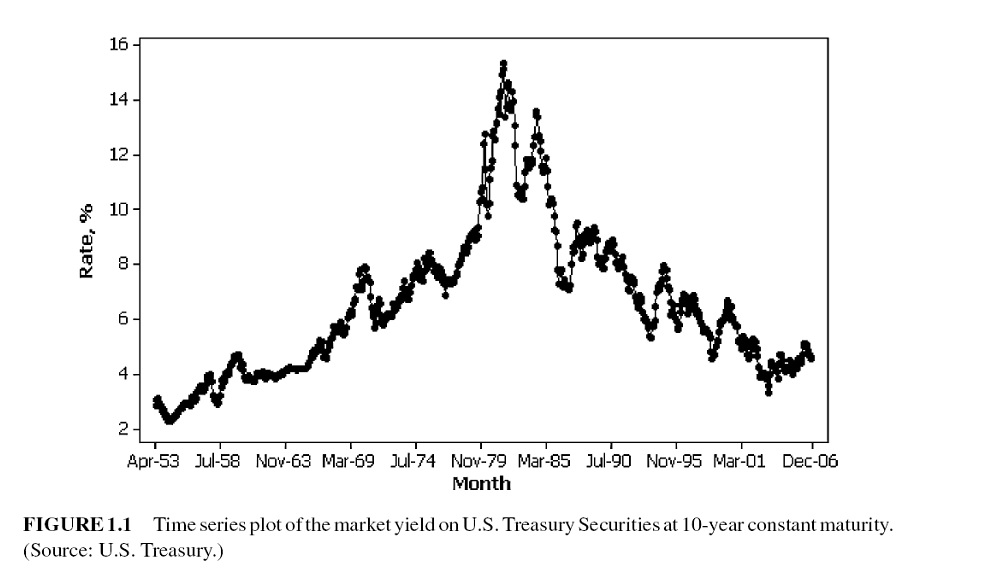
# Concepts of Prediction/Forecasting

A forecast is a prediction of some future event or event. Forecasting problems can be found in many applied areas such as business and industry, economics, finance, environmental sciences, social sciences, political sciences, etc. and can be categorized into

1. Short-term forecasting is which a prediction is only made for a few periods ahead (hours, days, weeks).
2. Medium-term in which one or two years of forecasting will be made.
3. Long-term forecasting in which prediction for several years ahead will be made.

We have discussed different regression models such as linear models and logistic regression models. We also discussed predictions using the fitted regression models. In practice, if the data are collected over time, then the aforementioned regression models are not valid for this type of data since the data values are serially correlated. New models that address time series are needed.

For time-series data, the typical scatter plot of the data values versus the time point at which the data values are observed is shown below.



Many business applications of forecasting utilize daily, weekly, monthly, quarterly, or annual data, but any reporting interval may be used.

The data may be instantaneous, such as the viscosity of a chemical product at the point in time where it is measured; it may be cumulative, such as the total sales of a product during the month; or it may be a statistic that in some way reflects the activity of the variable during the period, such as the daily closing price of a specific stock on the New York Stock Exchange.

* **Methods of forecasting**: There are two major methods for forecasting:
  + Quantitative forecasting methods that make formal use of historical data to build a mathematical/statistical model to project into the future.
  + Qualitative forecasting methods are used when little data are available (new product introduction). In this case, expert opinion is often used.

There are two modes for analyzing time series: frequency domain and time domain approaches. We restrict our discussion within the time domain to avoid more advanced mathematical tools in this class.

* **Statistical forecasting models**:
  + Regression methods.
  + Smoothing methods.
  + Formal time series analysis methods - ARIMA Box-Jenkins methodology.
  + Dynamic regression methods.
* **Some terminology**
* Point forecast or point estimate - a single predicted value (no information on the variation of the predicted value)
* Prediction interval - interval prediction, information on accuracy is available.
* Forecast horizon or lead time - based on the fitted model, the number of periods specified to be predicted.

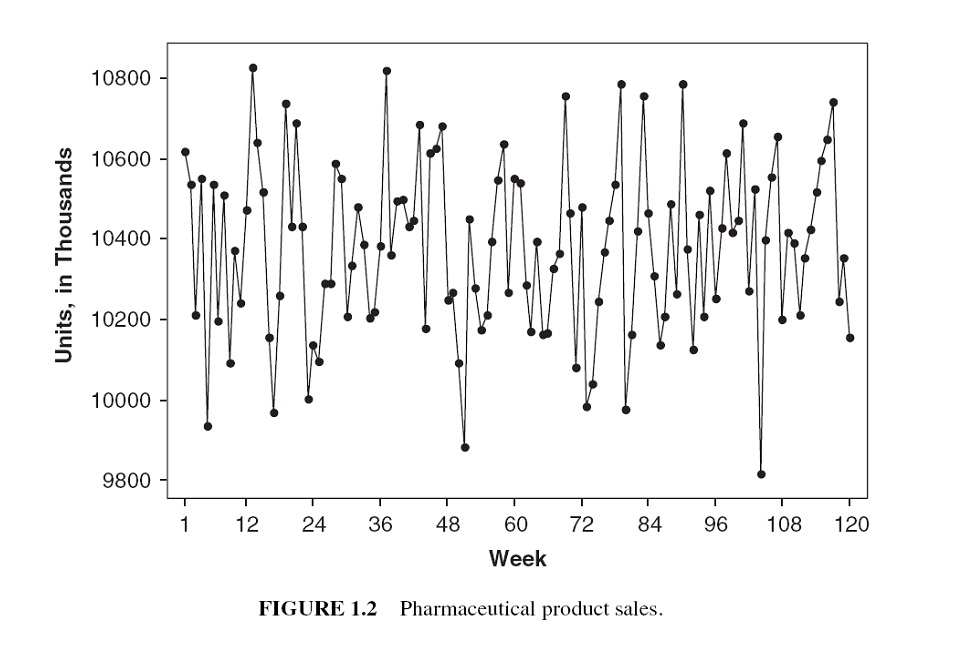
The forecast horizon is the length of time into the future for which forecasts are to be prepared.

# Types of Time Series Data

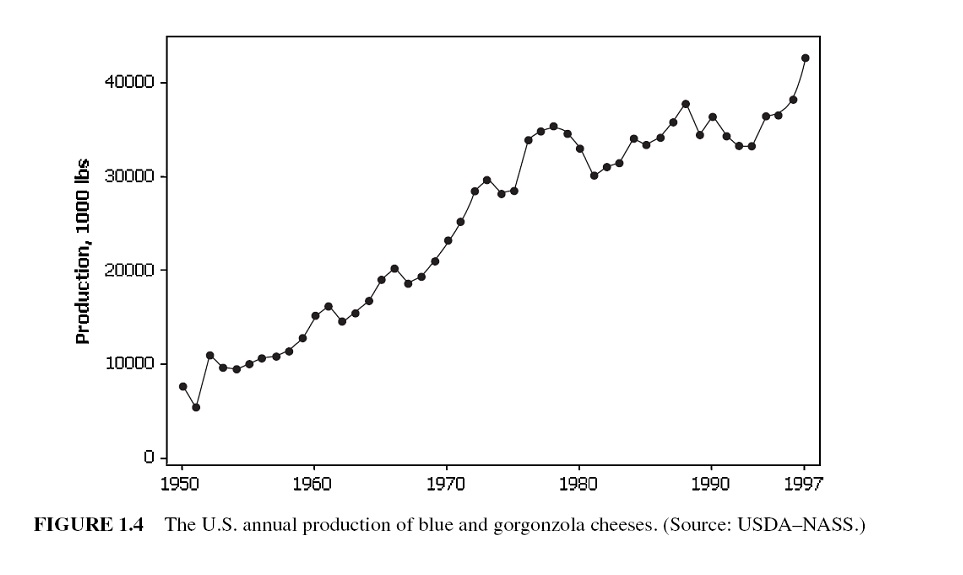
## Uncorrelated data, constant process model: random time series

Random time series is the result of many influences that act independently to yield nonsystematic and non-repeating patterns about some average value.

A purely random series has a constant mean and no systematic patterns. Simply averaging model is often the best and the most accurate way to forecast them.

 A special uncorrelated process: **white noise process**, . That is, a sequence , () is a white noise process if each value in the sequence has a) zero-mean; b) constant conditional variance; and c) is uncorrelated with all other realizations.

## 2. Autocorrelated data

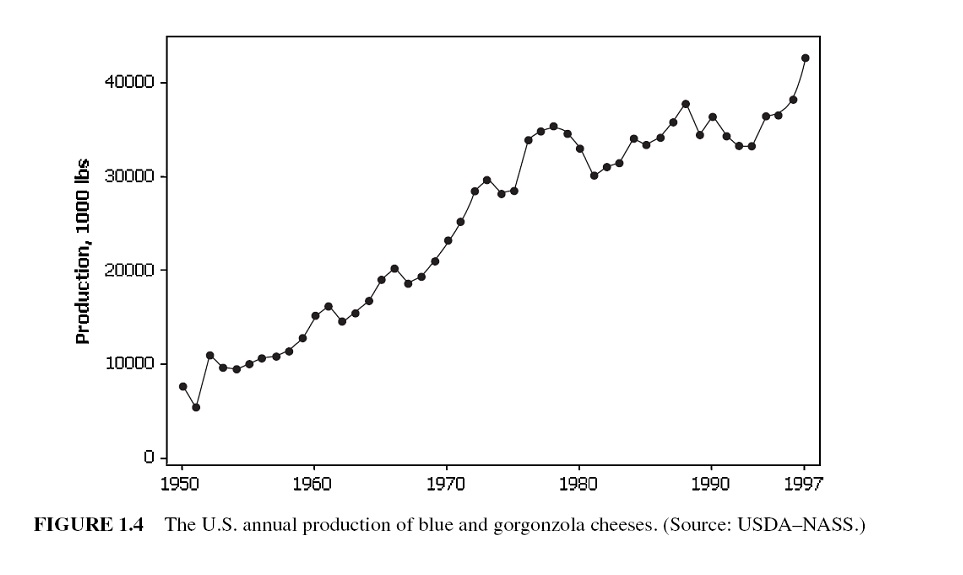


The correlation measures the degree of dependence or association between two variables.

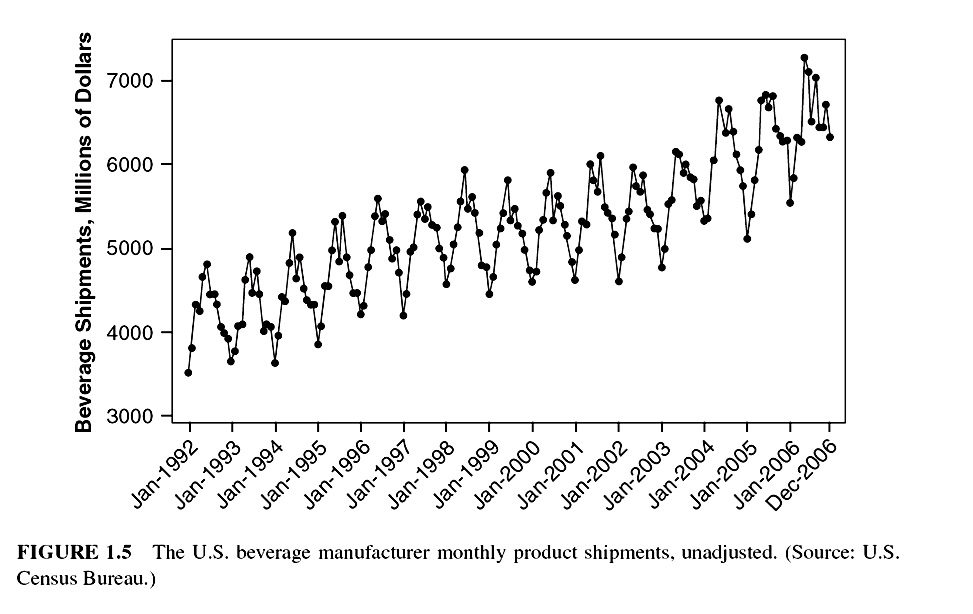
The term auto-correlation means that the value of a series in one time period is related to the value of itself in previous periods.

## Trend Data

A trend is a general increase or decrease in a time series that lasts for approximately seven or more periods. it can be a period-to-period increase or decrease that follows as a straight-line - this kind of pattern is called a linear trend.

 Trends are not necessarily linear because there are a large number of nonlinear causal influences that yield nonlinear series.

## Seasonal Data



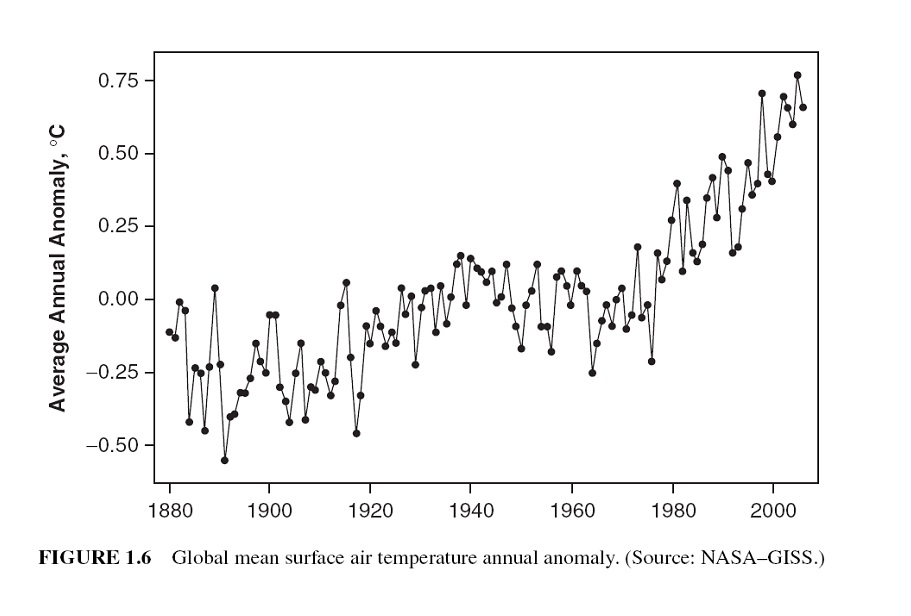
Seasonal series result from events that are periodic and recurrent (e.g., monthly, quarterly, changes recurring each year). The common seasonal influences are climate, human habits, holidays, repeating promotions, and so on.

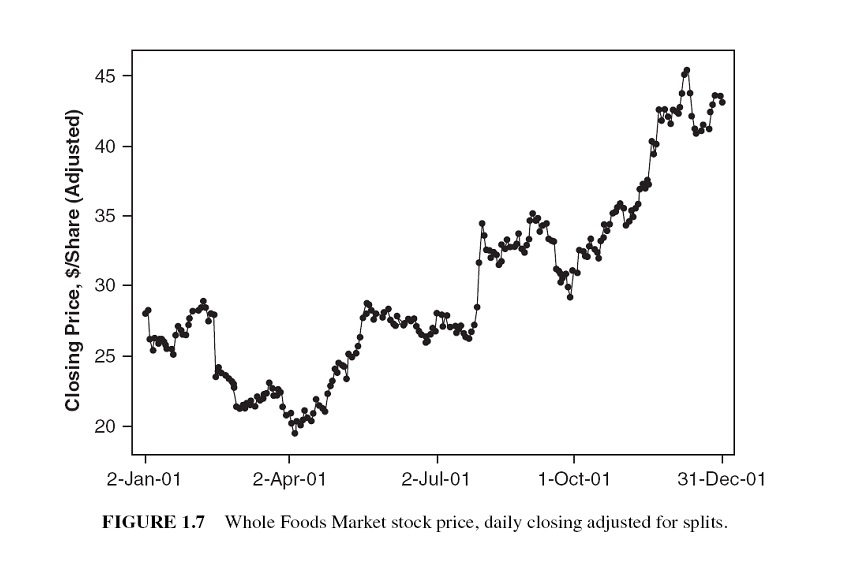
## Nonstationary Data

A stationary time series is one whose properties do not depend on the time at which the series is observed.

More formally, a strictly stationary stochastic process is one where given the joint statistical distribution of is NOT the same as the joint statistical distribution of for all and .

Thus, time series with trends, or with seasonality, are not stationary — the trend and seasonality will affect the value of the time series at different times.



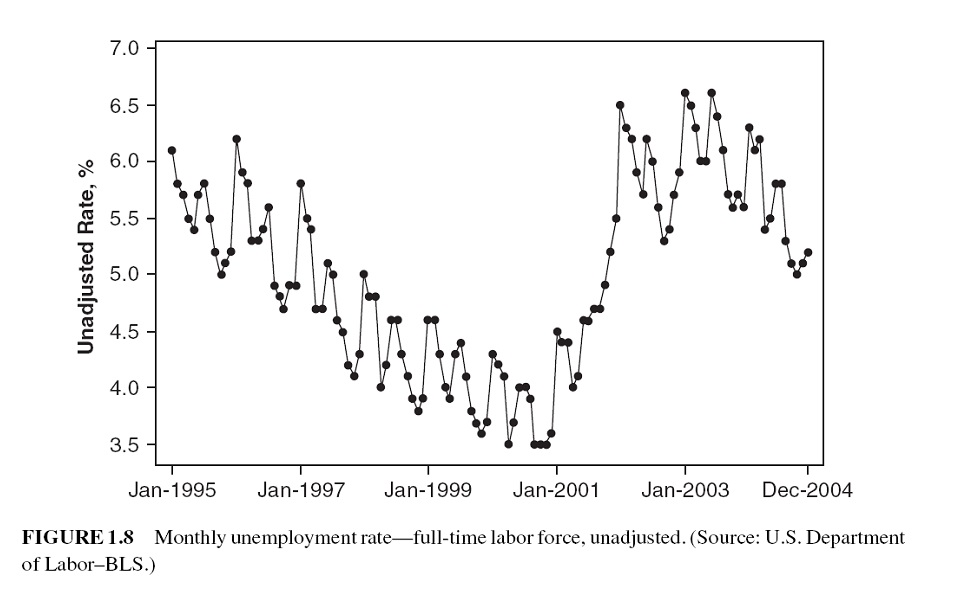


However, a time series with cyclic behaviour (but with no trend or seasonality) is stationary.

## Atypical Event (Outliers)

The analysis of past data can be made very complex when the included values are not typical of the past or future.

These non-typical values are called outliers, which are very large or small observations that are not indicative of repeating past or future patterns. Outliers occur because of unusual events.



# Some Technical Terms and Baseline Forecasting Models

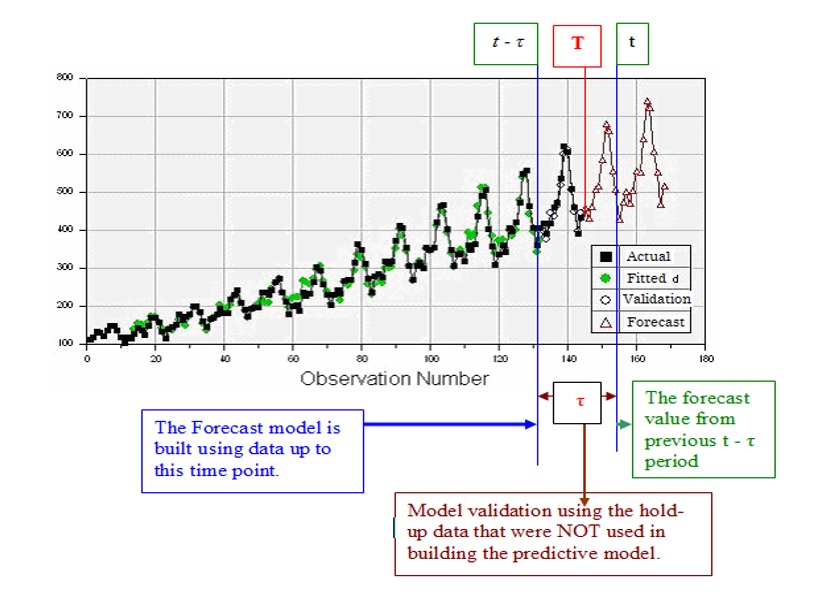
There are many forecasting models for different time series models. Some of them are extremely simple and surprisingly effective. In this module, we introduce some naive methods as baseline methods to understand some the basic concepts and technical terms before introducing formal forecasting models.

## Some Defintions

Notation of a time series data: suppose there are periods of data available with the most recent period. The observed data values will be denoted by for .

**Forecast v.s. fitted values**: in time series analysis, we distinguish between forecast (predicted) values from the fitted value:

* **Forecast value**: the value was produced at some previous period, say, (the predictive model on which the forecasting will be made should be built using the data up to ).
* **Fitted value**: The values produced by the fitted model at historical time points.



## Baseline Methods

**Working Data Set**: The Nile dataset was selected to expand knowledge of time series analysis. The Nile dataset contains 100 annual readings of the Nile River at Aswan for the years 1871 to 1970 (Anonymous, n.d.-a). Time series analysis and modeling are concerned with data elements that are ordered in a temporal fashion. The Nile data set is analyzed using various time series approaches and the results discussed.

1120, 1160, 963, 1210, 1160, 1160, 813, 1230, 1370, 1140, 995, 935, 1110, 994, 1020, 960,  
1180, 799, 958, 1140, 1100, 1210, 1150, 1250, 1260, 1220, 1030, 1100, 774, 840, 874, 694,  
940, 833, 701, 916, 692, 1020, 1050, 969, 831, 726, 456, 824, 702, 1120, 1100, 832, 764,   
821, 768, 845, 864, 862, 698, 845, 744, 796, 1040, 759, 781, 865, 845, 944, 984, 897, 822,  
1010, 771, 676, 649, 846, 812, 742, 801, 1040, 860, 874, 848, 890, 744, 749, 838, 1050,   
918, 986, 797, 923, 975, 815, 1020, 906, 901, 1170, 912, 746, 919, 718, 714, 740

When fitting time series models in R, most of R functions use requires times object which has a special structure and other features. For a given numeric vector, we can use R function **ts()** in library **forecast**.

nile.vec = c(1120, 1160, 963, 1210, 1160, 1160, 813, 1230, 1370, 1140, 995, 935, 1110, 994,  
1020, 960, 1180, 799, 958, 1140, 1100, 1210, 1150, 1250, 1260, 1220, 1030, 1100, 774, 840,  
874, 694, 940, 833, 701, 916, 692, 1020, 1050, 969, 831, 726, 456, 824, 702, 1120, 1100, 832,  
764, 821, 768, 845, 864, 862, 698, 845, 744, 796, 1040, 759, 781, 865, 845, 944, 984, 897,  
822, 1010, 771, 676, 649, 846, 812, 742, 801, 1040, 860, 874, 848, 890, 744, 749, 838, 1050,  
918, 986, 797, 923, 975, 815, 1020, 906, 901, 1170, 912, 746, 919, 718, 714, 740)  
##  
nile.ts = ts(nile.vec,   
 start = 1871,   
 end = 1970,   
 frequency =1 # one observation per year.   
 # if there is a weekly seasonal pattern,  
 # then the frequency = 7  
 )  
 nile.ts

## Time Series:  
## Start = 1871   
## End = 1970   
## Frequency = 1   
## [1] 1120 1160 963 1210 1160 1160 813 1230 1370 1140 995 935 1110 994 1020  
## [16] 960 1180 799 958 1140 1100 1210 1150 1250 1260 1220 1030 1100 774 840  
## [31] 874 694 940 833 701 916 692 1020 1050 969 831 726 456 824 702  
## [46] 1120 1100 832 764 821 768 845 864 862 698 845 744 796 1040 759  
## [61] 781 865 845 944 984 897 822 1010 771 676 649 846 812 742 801  
## [76] 1040 860 874 848 890 744 749 838 1050 918 986 797 923 975 815  
## [91] 1020 906 901 1170 912 746 919 718 714 740

This data set is included in library{forecast} as time series object. We can view the the data values in the this time series by typing `Nile’ to view the time series object.

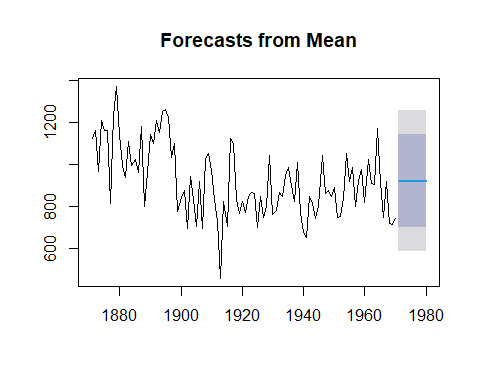
**Moving Average Method**

The moving average method forecasts of all future values to be equal to the average (or “mean”) of the historical data. If we let the historical data be denoted by , then we can write the forecasts as

is the estimated value of based on .

**Example 1**. We only use the basic moving average on the Nile river data. The following plot shows the pattern of the time series.

nile.fcast <- meanf(Nile, h=10)  
plot(nile.fcast)



The forecasted values for 10 future periods (h = 10 is called forecast horizon) is given below

kable(nile.fcast, caption = "Forecated readings of next 10 years with   
 confidence level at 80% and 95%, respectively.")

Forecated readings of next 10 years with confidence level at 80% and 95%, respectively.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Point Forecast | Lo 80 | Hi 80 | Lo 95 | Hi 95 |
| 1971 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1972 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1973 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1974 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1975 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1976 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1977 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1978 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1979 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |
| 1980 | 919.35 | 699.9303 | 1138.77 | 581.8912 | 1256.809 |

**Naïve method**

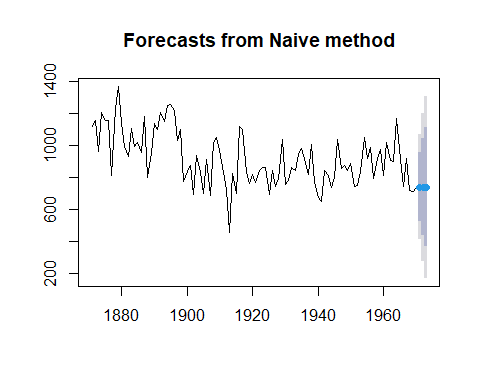
For naïve forecasts, we simply set all forecasts to be the value of the last observation. That is,

The naive forecast method works really well on the time series data generating from a process in which the next value in the sequence is a modification of the previous value in the sequence.

**Example 2**. We will use Naive forecast method on the the Nile River data.

The forecated values for the three future period and the forecasted intervals are plotted in the following figure.

naive.nile = naive(Nile, h=3)  
plot(naive.nile)



The corresponding values and confidence intervals are given in the following table.

kable(naive.nile, caption = "Naive forecasting method on Nile River data")

Naive forecasting method on Nile River data

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Point Forecast | Lo 80 | Hi 80 | Lo 95 | Hi 95 |
| 1971 | 740 | 525.5648 | 954.4352 | 412.0497 | 1067.950 |
| 1972 | 740 | 436.7429 | 1043.2571 | 276.2083 | 1203.792 |
| 1973 | 740 | 368.5874 | 1111.4126 | 171.9735 | 1308.027 |

## Seasonal Naive Method

The values of a seasonal time series data is influenced by seasonal factors (e.g., the quarter of the year, the month, or day of the week). seasonal time series are also called periodic time series.

If a data set has a seasonal pattern such as weekly and monthly patterns, the **moving average** and **naive** methods work poorly. We need a method to capture the seasonal patter. The **seasonal naive method** is modified from the **naive method** and has the following explicit forecasting function

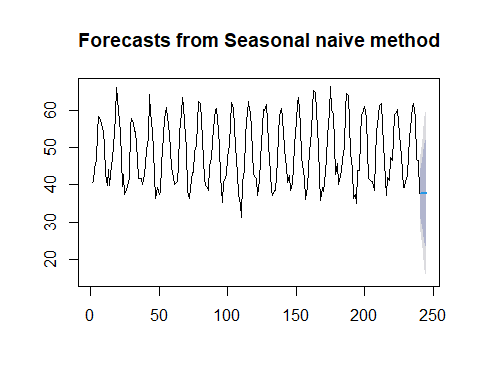
where the seasonal period, and is the integer part of (i.e., the number of complete years in the forecast period prior to time ). The forecasting formula looks complex, but it essentially say that, for example, with monthly data, the seasonal naive method forecasts for all future February values is equal to the last observed February value.

**Example 3** Nottingham monthly average temperature - the following data set is a collection of average monthly temperatures in Nottingham from 1920-1939. Since this is a monthly data, when we define the time series object with **ts()**, the option **frequency** should be set to 12.

nottem =c(40.6,40.8,44.4,46.7,54.1,58.5,57.7,56.4,54.3,50.5,42.9,39.8,44.2,39.8,45.1,47.0,  
 54.1,58.7,66.3,59.9,57.0,54.2,39.7,42.8,37.5,38.7,39.5,42.1,55.7,57.8,56.8,54.3,  
 54.3,47.1,41.8,41.7,41.8,40.1,42.9,45.8,49.2,52.7,64.2,59.6,54.4,49.2,36.3,37.6,  
 39.3,37.5,38.3,45.5,53.2,57.7,60.8,58.2,56.4,49.8,44.4,43.6,40.0,40.5,40.8,45.1,  
 53.8,59.4,63.5,61.0,53.0,50.0,38.1,36.3,39.2,43.4,43.4,48.9,50.6,56.8,62.5,62.0,  
 57.5,46.7,41.6,39.8,39.4,38.5,45.3,47.1,51.7,55.0,60.4,60.5,54.7,50.3,42.3,35.2,  
 40.8,41.1,42.8,47.3,50.9,56.4,62.2,60.5,55.4,50.2,43.0,37.3,34.8,31.3,41.0,43.9,  
 53.1,56.9,62.5,60.3,59.8,49.2,42.9,41.9,41.6,37.1,41.2,46.9,51.2,60.4,60.1,61.6,  
 57.0,50.9,43.0,38.8,37.1,38.4,38.4,46.5,53.5,58.4,60.6,58.2,53.8,46.6,45.5,40.6,  
 42.4,38.4,40.3,44.6,50.9,57.0,62.1,63.5,56.3,47.3,43.6,41.8,36.2,39.3,44.5,48.7,  
 54.2,60.8,65.5,64.9,60.1,50.2,42.1,35.8,39.4,38.2,40.4,46.9,53.4,59.6,66.5,60.4,  
 59.2,51.2,42.8,45.8,40.0,42.6,43.5,47.1,50.0,60.5,64.6,64.0,56.8,48.6,44.2,36.4,  
 37.3,35.0,44.0,43.9,52.7,58.6,60.0,61.1,58.1,49.6,41.6,41.3,40.8,41.0,38.4,47.4,  
 54.1,58.6,61.4,61.8,56.3,50.9,41.4,37.1,42.1,41.2,47.3,46.6,52.4,59.0,59.6,60.4,  
 57.0,50.7,47.8,39.2,39.4,40.9,42.4,47.8,52.4,58.0,60.7,61.8,58.2,46.7,46.6,37.8)  
nottem.ts = ts(nottem, frequency = 12, start = c(1920, 1))

Next, we plot the original series and the forecasted future values as well. Assuming forecasting horizon h = 5.

seasonal.naive = snaive(nottem, h = 5)  
plot(seasonal.naive)



The actual forecasted temperatures in the next 5 months and their predictive intervals are given by

kable(seasonal.naive, caption="Forecasted monthly average temperatures of Nottingham between Jan - May, 1940")

Forecasted monthly average temperatures of Nottingham between Jan - May, 1940

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Point Forecast | Lo 80 | Hi 80 | Lo 95 | Hi 95 |
| 241 | 37.8 | 31.08754 | 44.51246 | 27.53418 | 48.06582 |
| 242 | 37.8 | 28.30715 | 47.29285 | 23.28193 | 52.31807 |
| 243 | 37.8 | 26.17368 | 49.42632 | 20.01907 | 55.58093 |
| 244 | 37.8 | 24.37508 | 51.22492 | 17.26835 | 58.33165 |
| 245 | 37.8 | 22.79048 | 52.80952 | 14.84492 | 60.75508 |

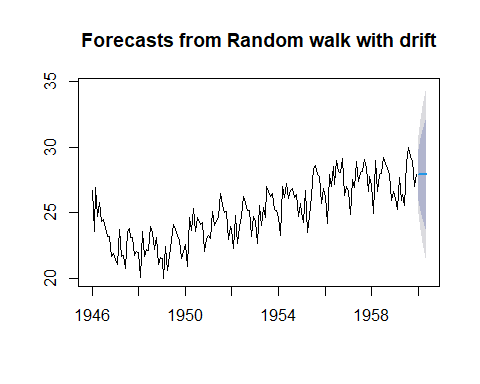
## Drift Method

The seasonal naive method allows non-constant predictive values by taking the same values from the time period in the previous season. The drift method is another variation of the naive method that allows the forecasts to increase or decrease over time. To be more specific, the next one-step forecast value is set to be the average change seen in the historical data. Thus the forecast for time is given by

where the amount of change over time is called the drift. This is the where the name of the method comes from.

**Example 4** New York City Monthly Birth Counts - the dataset contains the number of births per month in New York city, from January 1946 to December 1958. The data set can be downloaded from: <https://stat321.s3.amazonaws.com/w11-nycbirths.txt>. We will use seasonal naive method to forecast future values using function **naive()** in library{forecast}.

nycbirth = read.table("https://stat321.s3.amazonaws.com/w11-nycbirths.txt")  
births <- ts(nycbirth, frequency = 12, start = c(1946, 1))  
drift.pred = rwf(births, h= 5, drift = TRUE)  
plot(drift.pred)



As usual, we can extract the predicted value from the above drift forecast procedure in the following

kable(drift.pred, caption = "Forecast birth counts using drift method")

Forecast birth counts using drift method

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Point Forecast | Lo 80 | Hi 80 | Lo 95 | Hi 95 |
| Jan 1960 | 27.90439 | 25.96961 | 29.83916 | 24.94541 | 30.86337 |
| Feb 1960 | 27.91178 | 25.16741 | 30.65614 | 23.71464 | 32.10892 |
| Mar 1960 | 27.91917 | 24.54803 | 31.29030 | 22.76346 | 33.07487 |
| Apr 1960 | 27.92656 | 24.02241 | 31.83071 | 21.95567 | 33.89744 |
| May 1960 | 27.93395 | 23.55615 | 32.31174 | 21.23869 | 34.62920 |

# Accuracy Measures in Time Series Forecasting

## Training and Testing Data

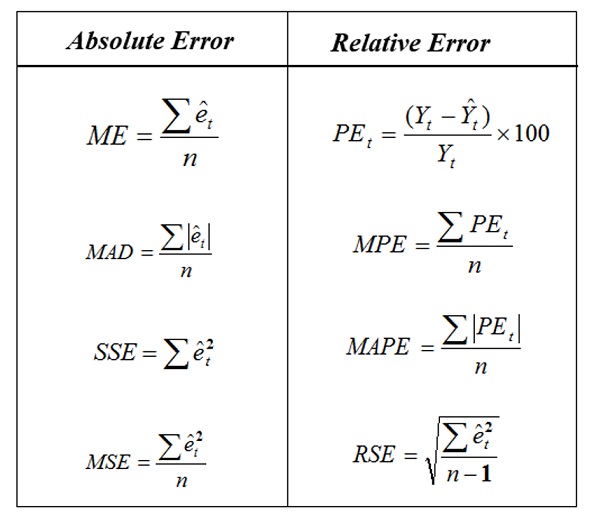
To evaluate forecast accuracy using genuine forecasts, we separate the available data into two portions, training and test data, where the training data is used to estimate any parameters of a forecasting method and the test data is used to evaluate its accuracy. We have used this logic in the logistic predictive modeling. The difference is that we cannot use random split method as we did in the predictive modeling using the logistic regression.

We have introduced several baseline forecasting methods in the previous section.



## Forecasting Errors

In a given time series data with n observations, we hold up k observations for evaluating forecast errors and use the rest of values to build time series models. We set the forecast horizon h = k. We can find the difference between the predicted and actual hold-up values to define various error metrics in the following table.



The error term . We will use the above error measures to define the accuracy metrics in a case study.

# Case Study

**Daily Female Births Dataset**: This dataset describes the number of daily female births in California in 1959. The units are a count and there are 365 observations

**A. Training and testing data**

female.births = read.csv("https://stat321.s3.amazonaws.com/w11-daily-total-female-births.csv")[,-1]  
## training and testing data: hold-up last 6 months of data for calculating forecasting errors  
training = female.births[1:350]  
testing = female.births[351:365]  
##  
female.births.ts = ts(training, frequency = 12, start = c(1959, 1))  
## shampoo.ts

**B. Building 4 Forecasting Models**

We next use the four baseline forecasting methods and the first 30 data values to forecast the next 6 month data values.

pred.mv = meanf(female.births.ts, h=15)$mean  
pred.naive = naive(female.births.ts, h=15)$mean  
pred.snaive = snaive(female.births.ts, h=15)$mean  
pred.rwf = snaive(female.births.ts, h=15, drift = TRUE)$mean  
###  
###  
pred.table = cbind( pred.mv = pred.mv,  
 pred.naive = pred.naive,  
 pred.snaive = pred.snaive,  
 pred.rwf = pred.rwf)  
kable(pred.table, caption = "Forecasting Table")

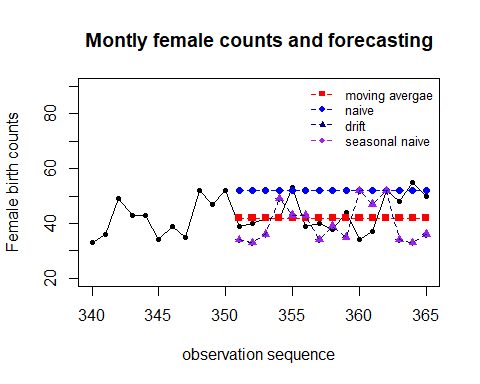
Forecasting Table

|  |  |  |  |
| --- | --- | --- | --- |
| pred.mv | pred.naive | pred.snaive | pred.rwf |
| 41.91429 | 52 | 34 | 34 |
| 41.91429 | 52 | 33 | 33 |
| 41.91429 | 52 | 36 | 36 |
| 41.91429 | 52 | 49 | 49 |
| 41.91429 | 52 | 43 | 43 |
| 41.91429 | 52 | 43 | 43 |
| 41.91429 | 52 | 34 | 34 |
| 41.91429 | 52 | 39 | 39 |
| 41.91429 | 52 | 35 | 35 |
| 41.91429 | 52 | 52 | 52 |
| 41.91429 | 52 | 47 | 47 |
| 41.91429 | 52 | 52 | 52 |
| 41.91429 | 52 | 34 | 34 |
| 41.91429 | 52 | 33 | 33 |
| 41.91429 | 52 | 36 | 36 |

**C. Visualization**

We now make time series plot and the predicted values. Note that, the forecast values were based on the model that use 350 histrical data in the time series. The following only show observations #320 -#365 and the 15 forecasted values.

plot(340:365, female.births[340:365], type="l", xlim=c(340,365), ylim=c(20, 90),  
 xlab = "observation sequence",  
 ylab = "Female birth counts",  
 main = "Montly female counts and forecasting")  
points(340:365, female.births[340:365],pch=20)  
##  
points(351:365, pred.mv, pch=15, col = "red")  
points(351:365, pred.naive, pch=16, col = "blue")  
points(351:365, pred.rwf, pch=18, col = "navy")  
points(351:365, pred.snaive, pch=17, col = "purple")  
##  
lines(351:365, pred.mv, lty=2, col = "red")  
lines(351:365, pred.snaive, lty=2, col = "purple")  
lines(351:365, pred.naive, lty=2, col = "blue")  
lines(351:365, pred.rwf, lty=2, col = "navy")  
##   
legend("topright", c("moving avergae", "naive", "drift", "seasonal naive"),  
 col=c("red", "blue", "navy", "purple"), pch=15:18, lty=rep(2,4),  
 bty="n", cex = 0.8)



We can see that **moving average** method worked pretty well. The seasonal and drift methods in this seasonal time series are equivalent. But **naive**, **seasnonal naive**, and **drift** methods worked poorly comaprative to **moving average** method. Intuitively, **moving average** should produce non-constant forecast values. What

**D. Accuracy Metrics**

We will use the mean absolute predict error (MAPE) to compare the performance of the four forecasting methods.

true.value = female.births[351:365]  
PE.mv = 100\*(true.value - pred.mv)/true.value  
PE.naive = 100\*(true.value - pred.naive)/true.value  
PE.snaive = 100\*(true.value - pred.snaive)/true.value  
PE.rwf = 100\*(true.value - pred.rwf)/true.value  
##  
MAPE.mv = mean(abs(PE.mv))  
MAPE.naive = mean(abs(PE.naive))  
MAPE.snaive = mean(abs(PE.snaive))  
MAPE.rwf = mean(abs(PE.rwf))  
##  
MAPE = c(MAPE.mv, MAPE.naive, MAPE.snaive, MAPE.rwf)  
## residual-based Error  
e.mv = true.value - pred.mv  
e.naive = true.value - pred.naive  
e.snaive = true.value - pred.snaive  
e.rwf = true.value - pred.rwf  
## MAD  
MAD.mv = sum(abs(e.mv))  
MAD.naive = sum(abs(e.naive))  
MAD.snaive = sum(abs(e.snaive))  
MAD.rwf = sum(abs(e.rwf))  
MAD = c(MAD.mv, MAD.naive, MAD.snaive, MAD.rwf)  
## MSE  
MSE.mv = mean((e.mv)^2)  
MSE.naive = mean((e.naive)^2)  
MSE.snaive = mean((e.snaive)^2)  
MSE.rwf = mean((e.rwf)^2)  
MSE = c(MSE.mv, MSE.naive, MSE.snaive, MSE.rwf)  
##  
accuracy.table = cbind(MAPE = MAPE, MAD = MAD, MSE = MSE)  
row.names(accuracy.table) = c("Moving Average", "Naive", "Seasonal Naive", "Drift")  
kable(accuracy.table, caption ="Overall performance of the four forecasting methods")

Overall performance of the four forecasting methods

|  |  |  |  |
| --- | --- | --- | --- |
|  | MAPE | MAD | MSE |
| Moving Average | 11.29861 | 77.08571 | 41.93687 |
| Naive | 22.83107 | 135.00000 | 111.00000 |
| Seasonal Naive | 20.37455 | 133.00000 | 112.86667 |
| Drift | 20.37455 | 133.00000 | 112.86667 |

In summary, the **moving average method** has the best performance. Note that the methods introduced in this module are the baseline forecasting. They are all descriptive since we did not use any statistical assumptions. In the next module, we will introduce a few formal non-parametric forecasting methods - exponential forecasting methods. We will use the same accuracy measures to compare different forecasting methods.