Time Series for Beginners

Jake Esprabens, Ari Arango, Joshua Kim

2020-06-12

Contents

Preface

This book is created with an objective to clearly explain the basics of time series analysis. The inspiration came from taking a time series course and constantly getting confused by the theory. Often, time series can be a tricky subject; therefore, this book will try to explain the essentials of time series using R.

Time series is an immense subject with so much to it; therefore, we won't be able to cover all of it in this book. We will solely focus on what we believe is the most important to provide you with the ability to forecast with time series.

Make sure to check out our correspoding API package and Stock Shiny App to explore real world uses of time series. This package will allow you to directly grab the most recent data on stock prices around the world. Moreover, the Shiny App will allow you to visualize and interact with this data as a time series and predict how the stock prices will move in the future!

I recommend installing these 3 packages before we get started:

```
library(tidyverse)
library(ggfortify)
library(forecast)
```

Chapter 1

Introduction to Time Series

1.1 What is a Time Series?

A time series is a series of data points over time. Sounds pretty simple, right?

Let's take a look at an example of time series using real world data. This data set looks at the average global temperature anomalies in celsius per month from January 1950 to December 2016.

Let's see what what this data looks like.

```
##    Mean
## 1 -0.30
## 2 -0.26
## 3 -0.06
## 4 -0.21
## 5 -0.12
```

We must convert the data into a time series object. Now we know that this data is monthly and that it started in January 1950 and ended in December 2016. So we must convert it to a time series object, using the *ts()* function in R. It takes in these arguments.

- start = the starting time of the time series
- end = the ending time of the time series
- frequency = the number of observations per unit of time.
  - For monthly data, frequency will be 12
  - For quarterly data, frequency will be 4
  - For biannual data, frequency will be 2

```
temp.ts <- ts(temp, start = 1950, end = 2016, frequency = 12)
```

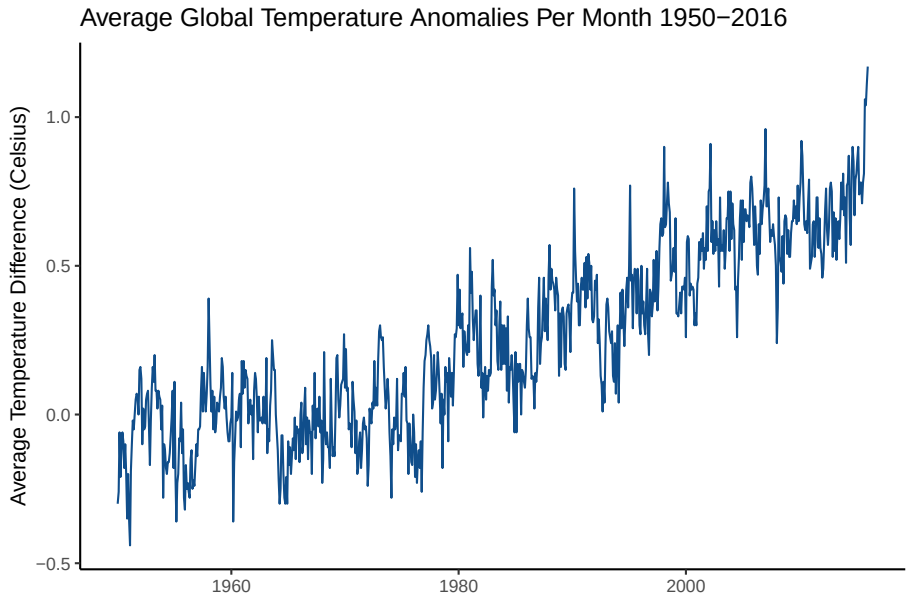
```
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 1950 -0.30 -0.26 -0.06 -0.21 -0.12 -0.06 -0.09 -0.18 -0.10 -0.20 -0.35 -0.20
## 1951 -0.35 -0.44 -0.19 -0.10 -0.02 -0.05  0.00  0.05  0.07  0.06  0.00  0.15
## 1952  0.16  0.12 -0.10  0.02 -0.05 -0.04  0.05  0.07  0.08 -0.04 -0.17 -0.02
```

As we can see, the data is now a time series object with 804 observations where each observation represents a month from 1950 to 2016.

How would this object look plotted?

There are several ways to plot time series in R, including the base R function *plot()*; however, we will be using the *autoplot()* function from the **forecast** package.

```
autoplot(temp.ts, colour = "dodgerblue4") +
  ggtitle("Average Global Temperature Anomalies Per Month 1950-2016") +
  ylab("Average Temperature Difference (Celsius)") +
  theme_classic()
```



What can we take away from this plot?

Well, the first thing you might notice is that the average global temperatue has risen rapidly since about 1970! It also gradually increased from 1950 to 1970. It

also staggers a lot all the way through. One might conclude that temperature has increased in the past 66 years! How can you define the changed happening in this time series though? What does the rapid staggering mean? What do you call the upward movement?

1.2 Components of Time Series

What makes up a time series?

Without going into too much notation, a simple additive decomposed model will look like this:

$$x_t = m_t + s_t + e_t$$

where:

- m_t is the trend
- s_t is the seasonality
- e_t is the error or random white noise

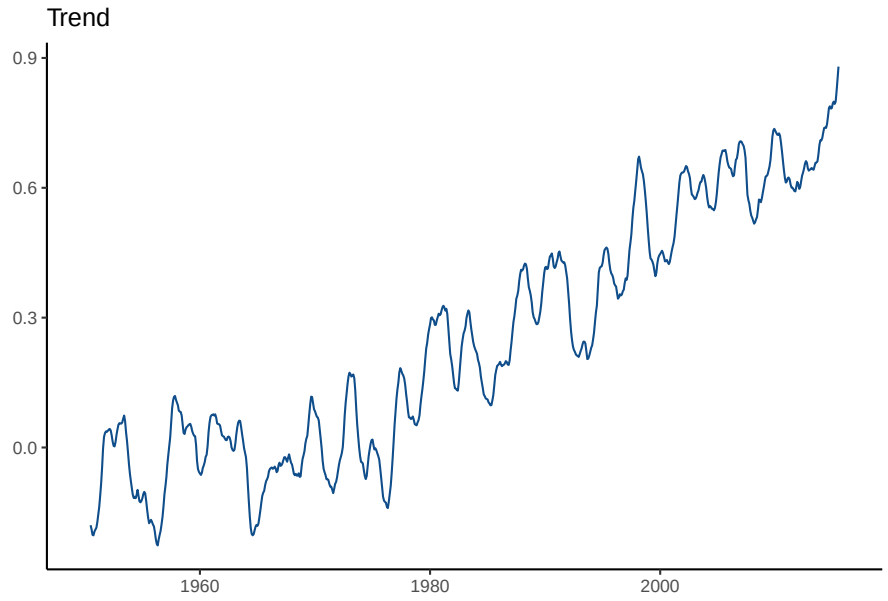
To put it simply, the components of a time series model are defined as:

- trend - the increasing or decreasing values in a series
- seasonality - the repeating short-term cycles in the series
- random white noise - random variation in the series

To visualize these in R, you can easily use the base R function, *decompose()*, to produce plots for all three of these components. However, I chose to create my own plots using **ggplot2**.

Trend

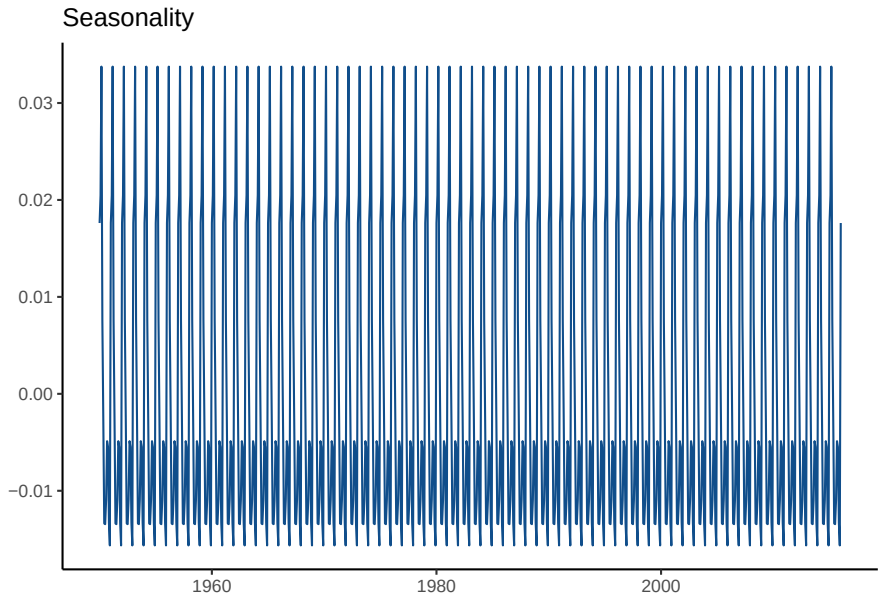
What does the trend plot look like for this global temperature time series?



As we can see, there is strong upward trend. This means that the temperature is gradually going up since 1950! What if the trend line was going down? Then you would conclude that the temperature is decreasing. What if it stayed constant? Then, you can conclude that there is no trend!

Seasonality

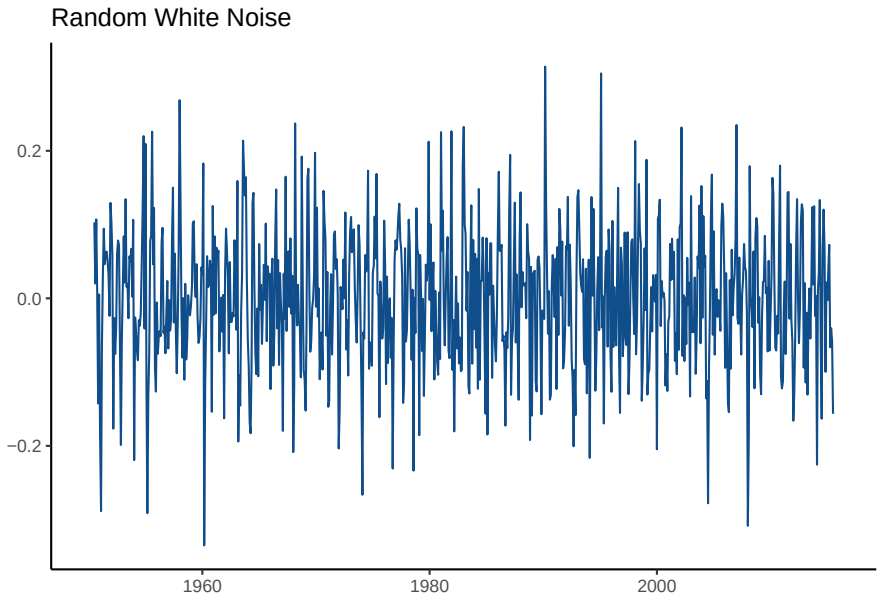
What does the seasonality plot look like for this global temperature time series?



In a seasonality plot, you are looking for a common short-term pattern. Seasonality can be caused by many thing depending in the dataset. It is hard to tell if there is seasonality in this time series because of the rapid staggering. Usually, seasonality will be more obvious. An example of seasonality is airline flyers. Many people are often flying in the summer and in the winter for vacation. Thus, that specific time series will see high spikes every 6 months.

Random White Noise

What does the random white noise plot look like for this global temperature time series?



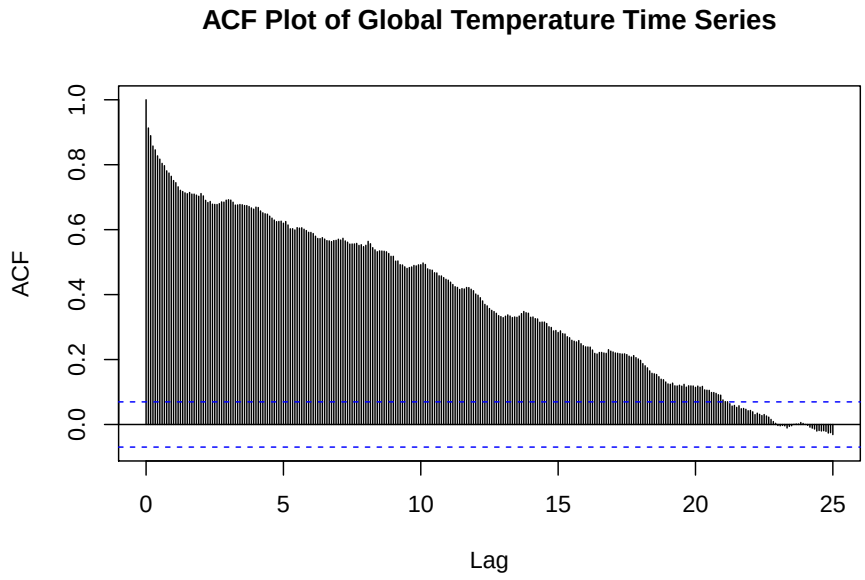
Looks pretty random right? That's because it is! This plot illustrates the variation in the time series. This is the unexplained variation that solely happens by chance. This term is similar to the error term in a regression model.

1.3 Stationarity

Stationarity is one of the most important characteristics of a time series. What does it mean for a time series to be stationarity? It is defined by having a constant mean and variance across the time series. A time series needs to be stationary in order for it to make good prediction. We can check if a time series is stationary by looking at the **autocorrelation function (ACF)**. What exactly is autocorrelation? Autocorrelation basically measures the similarity between observations as a function of the time lag between them.

Let's take a look at the ACF plot of the global temperature time series, using the *acf()* function in R.

```
acf.plot <- acf(temp.ts, lag.max = 300)
```



What does this mean?

If the series was stationary, we would see basically every line within the blue confidence intervals. However, we see that every spike is out of these lines. This slow decay represents means that there is a trend in the time series, but there is no seasonality!

Here are some common ACF plots you may see:

- Trend and seasonality - Takes a long time decay and also looks similar to a sin graph.
- Trend, but no seasonality - Decays very slowly and the spikes are out of the blue confdience intervals.
- Seasonality, but no trend - Decays quickly and looks similar to a sin graph
- Stationarity - All lines within the blue confidence intervals, besides the first spike.

If a time series has trend or seasonality, then it is not stationary. This means that it won't be good for forecasting. What can we do to fix this? We can create new models to represent the time series and make it stationary. We will go over this in the next chapter.

Chapter 2

Modelling Time Series

As mentioned before, a time series must be stationary for it to be used to predict well founded values. We will go over several models that we can create in order to allow forecasting.

Please note that the first 3 models we cover, AR, MA, and ARMA, can be used on already stationary time series in order to allow them to predict better values. The remaining models are used on non-stationary time series.

2.1 AR and MA

Two of the most common models in time series are the Autoregressive (AR) models and the Moving Average (MA) models.

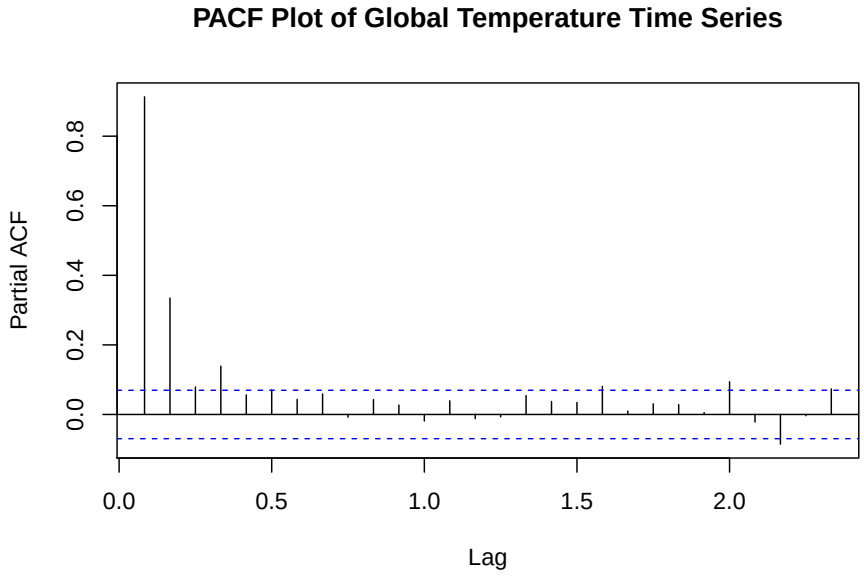
Autoregressive Model: AR(p)

The autoregressive model uses observations from preivous time steps as input to a regression equations to predict the value at the next step. The AR model takes in one argument, p, which determines how many previous time steps will be inputted.

The order, p, of the autoregressive model can be deterimined by looking at the **partial autocorrelation function (PACF)**. The PACF gives the partial correlation of a stationary time series with its own lagged values, regressed of the time series at all shorter lags.

Let's take a look at the PACF plot for the global temperature time series using the *pacf()* function in R.

```
pacf.plot <- pacf(temp.ts)
```



What should we look for in this plot? The primary goal is to look for the number of significant spikes outside of the blue confidence intervals. In this plot, I would determine there to be 2 spikes, one at 0.1 and the other at 0.3. The spike at the 0 does not count and any spikes outside of the blue later in the plot are likely due to random error. Therefore, this looks like an AR(2) model.

Let's look at an AR(2) model for the global temp time series. You can use the *ar()* function in R; however, I recommend using the *Arima()* function from the **forecast** package because we'll be using it later on in this chapter.

```
# the first index in the order argument represents the order of the AR(2) model
ar.model <- Arima(temp.ts, order = c(2,0,0))
ar.model
```

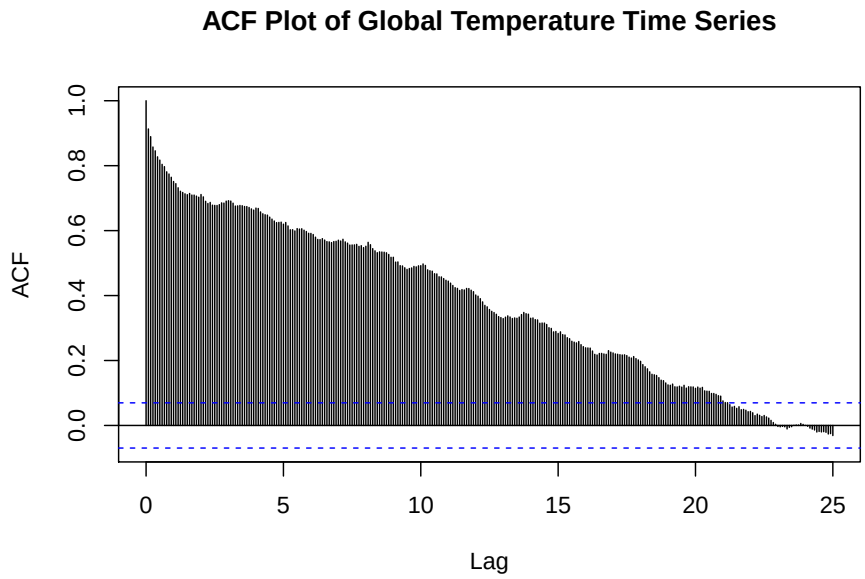
```
## Series: temp.ts
## ARIMA(2,0,0) with non-zero mean
##
## Coefficients:
##         ar1     ar2    mean
##       0.588  0.3700  0.2697
## s.e.  0.033  0.0332  0.0908
##
## sigma^2 estimated as 0.01234:  log likelihood=617.65
## AIC=-1227.3   AICc=-1227.25   BIC=-1208.6
```

So, this is our AR(2) model. How do we know it it's a good model? We will compute other models first and then talk about how to compare them.

Moving Average Model: MA(q)

The moving average model is a time series model that accounts for very short-run autocorrelation. It basically states that the next observation is the mean of every past observation. The order of the moving average model, q, can usually be estimated by looking at the ACF plot of the time series. Let's take a look at the ACF plot again.

```
acf.plot <- acf(temp.ts, lag.max = 300)
```



As we have seen, this ACF plot takes a very long time to converge. What does this mean? This likely means that making a moving average model of this time series would not fix the problem of not having a stationary time series. Thus, the MA model will likely not be a good model to forecast with; however, for the sake of comparing models, we will still view one.

Let's looks at an MA(5) model. Usually, we would pick order, q, for how many signficiant spikes there are in the ACF plot; however, considering that there are hundreds in this example, we will just use 5. **Careful:** if you use too high of an order, it can result in too many predictors in the model which may cause overfitting.

```
# the third index in the order argument represents the order of the MA(5) model.
ma.model <- Arima(temp.ts, order = c(0,0,5))
ma.model
```

```
## Series: temp.ts
## ARIMA(0,0,5) with non-zero mean
##
## Coefficients:
##          ma1     ma2     ma3     ma4     ma5    mean
##       0.8056  0.8663  0.6675  0.5288  0.2949  0.2479
## s.e.  0.0349  0.0420  0.0402  0.0340  0.0299  0.0197
##
## sigma^2 estimated as 0.01799:  log likelihood=470.17
## AIC=-926.35   AICc=-926.2   BIC=-893.61
```

We didn't cover the actual formulas and notation of the models; however, if this interests you, view this webiste below for more details. **Formulas Behind AR and MA Models**

The orders of the AR and the MA models or usually picked by the number of significant spikes in the PACF and ACF plots respectively; however there are other conditions as well that can be seen if you click **here**.

2.2 ARMA, ARIMA, AND SARIMA

The autoregressive moving average model (ARMA), autoregressive integrated moving average model (ARIMA) and the seasonal autoregressive integrated moving average model (SARIMA) are also commonly used models in time series analysis. Evidently, they all come from the same family. Thus, we will explain the small differences between them.

Autoregressive Moving Average Model: ARMA(p,q)

Autoregressive moving average models are simply a combination of an AR model and an MA model. Let's take a look at what our ARMA model would be.

We are going to build an ARMA(2,5) model by simply using the two orders from the previous models.

```
arma.model <- Arima(temp.ts, order = c(2,0,5))
arma.model
```

```
## Series: temp.ts
## ARIMA(2,0,5) with non-zero mean
##
```

```
## Coefficients:
##          ar1      ar2      ma1     ma2      ma3     ma4      ma5    mean
##       1.8305  -0.8306  -1.3337  0.4087  -0.1512  0.1674  -0.0684  0.4220
## s.e.  0.0635   0.0634   0.0720  0.0663   0.0602  0.0591   0.0381  0.4206
##
## sigma^2 estimated as 0.01158:  log likelihood=644.73
## AIC=-1271.45   AICc=-1271.22   BIC=-1229.37
```

Autoregressive Integrated Moving Average Model: ARIMA(p,d,q)

This model is the same as the previous, except now it has this weird d argument. What does this d stand for? d represents the number of nonseasonal differences needed for stationarity. Simply, d just makes nonstationary data stationary by removing trends!

How do you pick your differencing term?

Usually, small terms are picked for the differencing term. If you pick too high, you will likely cause your model to incorrectly represent your data. Some general rules for picking your differencing term are that differencing should not increase your variance and the autocorrelation of the model should be less than -0.5.

Thus, I tried a few differencing terms and concluded that $d = 1$ would be best for the model as it had the lowest variance and the autocorrelation was less than -0.5.

```
arima.model <- Arima(temp.ts, c(2, 1, 5))
arima.model
```

```
## Series: temp.ts
## ARIMA(2,1,5)
##
## Coefficients:
##          ar1     ar2      ma1      ma2     ma3     ma4     ma5
##       0.1022  0.6468  -0.6046  -0.6056  0.1762  0.0520  0.0157
## s.e.  0.2056  0.1859   0.2096   0.2859  0.1040  0.0581  0.0517
##
## sigma^2 estimated as 0.01156:  log likelihood=645.66
## AIC=-1275.33   AICc=-1275.14   BIC=-1237.93
```

Seasonal Autoregressive Integrated Moving Average Model: SARIMA(p,d,q)(P,D,Q)s

The SARIMA model is an extension of the ARIMA model. The only difference now is that this model added on a seasonal component. As we saw, ARIMA is good for making a non-stationary time series stationary by adjusting the trend. However, the SARIMA model can adjust a non-stationary time series by removing trend and seasonality.

As we know:

- p - the order of the autoregressive trend
- d - the order of the trend differencing
- q - the order of the moving average trend

What do (P,D,Q)s mean?

- P - the order of the autoregressive seasonality
- D - the order of the seasonal differncing
- Q - the order of the moving average seasonality
- s - the number of periods in your season

How do you pick these new terms?

There are several ways to pick these orders; however, when trying to use the SARIMA model in practice, it is likely best to let R or other software estimate the parameters for you. This article attached **here** mentions this in more detail.

In our example, we may not have a SARIMA model because our time series did not have seasonality. Therefore, it may follow a SARIMA(2,1,5)(0,0,0)12.

The s term would be 12 because there would be 12 periods (months) in the season if we had seasonality. We will still follow through with an example. We can use the sarima function from the **astsa** package in R.

```
sarima.model <- sarima(temp.ts, 2,1,5,0,0,0,12)
```

```
##
## Call:
## stats::arima(x = xdata, order = c(p, d, q), seasonal = list(order = c(P, D,
##     Q), period = S), xreg = constant, transform.pars = trans, fixed = fixed,
##     optim.control = list(trace = trc, REPORT = 1, reltol = tol))
##
## Coefficients:
##          ar1     ar2      ma1      ma2     ma3     ma4    ma5  constant
##       0.1286  0.6396  -0.6452  -0.5892  0.1750  0.0575  0.014    0.0012
## s.e.  0.2465  0.2227   0.2505   0.3498  0.1224  0.0628  0.055    0.0002
##
## sigma^2 estimated as 0.0113:  log likelihood = 650.7,  aic = -1283.4
```

Now that we have 5 different models, which one do you choose?

Often, this can be done simply by looking at the Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC). In our example, we will be

looking at the AIC. Generally, a smaller AIC means the model fits the time series better.

If you check the output for each model, you will see they all have an AIC value. I will reprint them below as well.

```
##                     Model      AIC
## 1                   AR(2) -1227.30
## 2                   MA(5)  -926.35
## 3               ARMA(2,5) -1271.45
## 4            ARIMA(2,1,5) -1275.33
## 5 SARIMA(2,1,5)(0,0,0)12 -1283.40
```

As we can see, the SARIMA model actually had the lowest AIC, thus we would conclude that the SARIMA made the time series stationary and is most sutiable for forecasting. However, as I mentioned before; when it comes to finding the best model, R or other software is likely the best. Thus, we will use the *auto.arima()* function from the **forecast** package that will automatically select orders for us! In practice, I definitely recommend using this rather than going through each model and testing different orders.

```
best.model <- auto.arima(temp.ts)
best.model
```

```
## Series: temp.ts
## ARIMA(2,1,3)(1,0,0)[12] with drift
##
## Coefficients:
##           ar1     ar2      ma1      ma2     ma3     sar1   drift
##       -0.0069  0.6955  -0.5143  -0.6984  0.2307  -0.0304  0.0012
## s.e.   0.1027  0.0954   0.1128   0.1549  0.0759   0.0371  0.0002
##
## sigma^2 estimated as 0.01142:  log likelihood=650.08
## AIC=-1284.15   AICc=-1283.97   BIC=-1246.76
```

As we can see, this model that was selected for us, SARIMA(2,1,3)(1,0,0)12, has the lowest AIC at -1284.15. We will continue to use this model in the next chapter as we dive into forecasting.

Chapter 3

Forecasting

Now that we covered the fundamentals of time series analysis, we finally get to immerse ourselves in forecasting!

3.1 What is Forecasting?

Forecasting is simply the process of using past data values to make educated predictions on future data values. As stated in the last chapter, the time series should be stationary if you want to make well-informed predicitons. This can be done by fitting an arima model by using the *auto.plot()* function in the **forecast** package. Then, all you have to do is apply the *forecast()* function to get your prediction! The *forecast()* function can take in another argument along with your model. You can also input h, the number of predicted time periods you want. This function is very practical for real world analysis of time series.

Forecasting is done in so many fields around the world. You will often see forecasting in the business and financial field for companies that want to predict their profit or expenses. Forecasting can be used to predict stock prices as well! You will see it in the environemntal field, such as this current example with global warming. The economic field also heavily uses time series and forecasting to predict how societies will behave. This is just a few examples of numerous time series and forecasting uses in the real world.

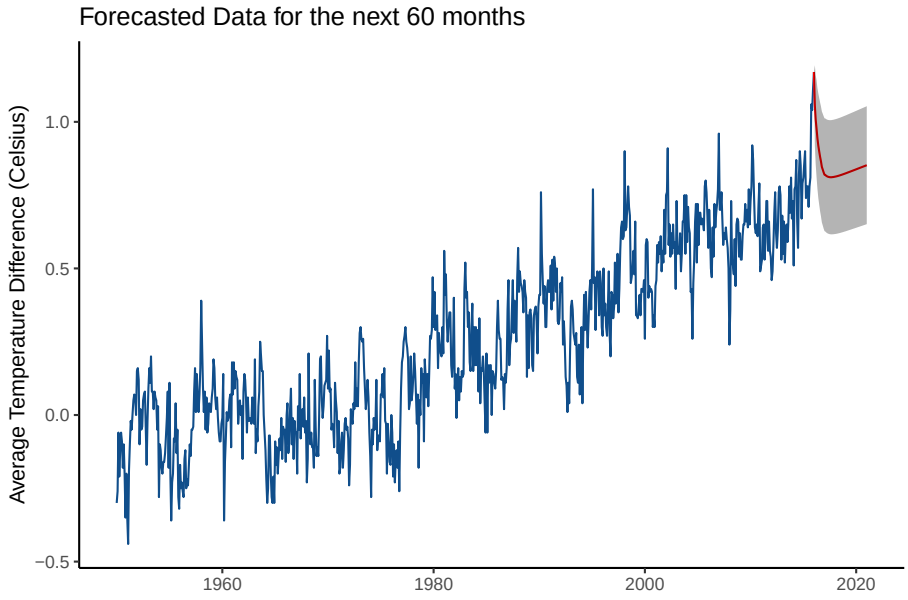
3.2 Example: Global Temperature

Let's forecast with our global temperature data now. As we saw, we fit the data with a SARIMA(2,1,3)(1,0,0)12. Now that we have our model, we can simply use the forecast(ts, h) function from the **forecast** package. As mentioned, h

represents the number of observations we want to predict into the future. Let's say we want to predict 5 years into the future. Since our data ended in December 2016, the next 5 years will include each month from January 2017 to December 2022. Moreover, since our observations were every month, we can set h to $12 * 5 = 60$. Let's see what happens.

```
forecast.data <- forecast(best.model, h = 60)

autoplot(forecast.data, ts.colour = "dodgerblue4", predict.colour = "red") +
  ggtitle("Forecasted Data for the next 60 months") +
  ylab("Average Temperature Difference (Celsius)") +
  theme_classic()
```



The original time series is depicted in dark blue and the predicted data is represented by the red line. The grey shading around the predicted values represents the 95% confidence interval. This simply states that we are 95% percent confident that the data point at time t will fall between two bounds. You can change the confidence bands in the forecast function and even view the confidence interval bounds in the forecasted object.

What can you take away from this forecasted model?

It looks as it the average temperature difference will drop in 2017; however, it seems that it will gradually rise again. This is fairly consistent in what we see in the rest of the data. It seems in the past 66 years, the temperatue decreases

every once in a while, but then gradually rises. This looks like it's a fairly good prediction.

There are many other methods to create a model for your time series data. You can also check which is the best predictor by looking at the mean absolute percentage error (MAPE). This is beyond the scope of this short tutorial, but I encourage you to learn more about it **here**.

Chapter 4

Summary

Throughout this book, we scraped the surface of time series analysis. In review, we covered:

- What is a time series?
  - A time series is a series of data points over time.
- Components of a time series
  - Trend, seasonality, random white noise.
- Stationarity
  - Defined as constant means and varaince throughout the series.
  - Time series should be stationary when forecasting.
  - Must fit a model to the time series to detrend and deseasonalize the series.
- AR and MA models
  - The autoregressive model uses observations from preivous time steps as input to a regression equations to predict the value at the next step.
  - The moving average model is a time series model that accounts for very short-run autocorrelation.
  - You should use the ACF and PACF plots to determine the order of these.
  - These models only work on stationary data.
- ARMA, ARIMA, and SARIMA Models
  - ARMA models are a combination of AR and MA models and only works on stationary data.

  - ARIMA models are the same as an ARMA model, but there is a differencing term to detrend the non-stationary data.
  - SARIMA models are the same as an ARIMA model, but there are seasonal terms to detrend and deseasonalize the non-stationary data.
  - **In practice, use the auto.arima() function from the forecast package.**

- Forecasting
  - Defined as using the previous data points to make a well-informed predictions of the future.
  - The **forecast()** function is best to predict your time series.

These is just a small chunk of what time series has to offer. There are many other components and different techniques to detrend and deseasonalize your time series that we didn't talk about in this book. I hope this short tutorial assisted you in learning the basics of time series and equipped you with the tools to forecast.

Now that you have learned the basics, make sure to check out our correspoding API package and Stock Shiny App to explore real world uses of time series You will be able to look at the series of real time stock prices for companies around the world!

Duke University, *Summary of Rules for Identifying ARIMA Models*

Global Temperature Time Series Data

Holmes, Scheuerell, & Ward, *Applied Time Series Analysis*

Hyndman & Athanasopoulos, *Forecasting Principles and Practice*

Khan, *ARIMA Model for Forescating - Example in R*

Towards Data Science, *The Complete Guide to Time Sereis Analysis and Forecasting*