Chapter 8
ARIMA Models

1 Introduction

Like exponential smoothers, ARIMA models are flexible and can fit a wide variety of time series data. However, in contrast to exponential smoothers, which are built to match the features of the data that can be observed in time plots (such as trend or changing level), ARIMA models are built to match the autocorrelation in the data. In other words, while the starting point of an exponential smoother is the time plot, to see if you can notice any trend or seasonality, the starting point of an ARIMA model is the autocorrelation function plot (ACF plot), to see what autocorrelation patterns exist. ARIMA models are even more flexible than exponential smoothers — in fact, all of the "additive" exponential smoothers are special cases of ARIMA models.

Before jumping all the way into ARIMA models, we will break them down into smaller pieces, and then put the pieces together. ARIMA stands for Auto-Regressive, Integrated, Moving Average, and we can treat each one of these three pieces separately to see what they mean and how they work. We will start with autoregressive models, progress to moving average models, combine them to form an ARMA model, and finally investigate ARIMA models. Note that data used in an AR, MA, or ARMA models should be stationary. Data used in an ARIMA model may have a unit root or trend, while data used in a "seasonal ARIMA" model may also have seasonality.

2 Autoregressive (AR) Models

AR models are appropriate when the current value of the data series depends on its past values. One example of this is in sports, where a team's average margin of victory one year tends to be similar to its average margin of victory the following year. From one year to the next, most teams have the same coach and mostly the same players. Of course, far enough in the future

the entire roster and coach will have changed, and the performance of the team far in the future will be totally unrelated to how they perform this year. AR models can exploit this type of autocorrelation to make forecasts that are similar to the recent data in the short-run, but that approach the historical average in the long-run.

Mathematically, we will refer to AR(p) models, where p is the number of time lags of the data that are included in the AR equation. For example, an AR(1) model means that the data in the current time period directly influences the value of the data in the next time period. An AR(5) model means that the data in the five most recent time periods directly influence the data in the next time period. In an equation, we express AR(p) models as:

$$y_t = b_0 + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$$
$$\varepsilon_t \sim \mathcal{N}(0, \sigma)$$

where b_0 is the intercept and ϕ_k is the regression coefficient on the k^{th} time lag.

2.1 AR(1) Model

For a more specific example, an AR(1) model would be written as:

$$y_t = b_0 + \phi_1 y_{t-1} + \varepsilon_t$$
$$\varepsilon_t \sim \mathcal{N}(0, \sigma)$$

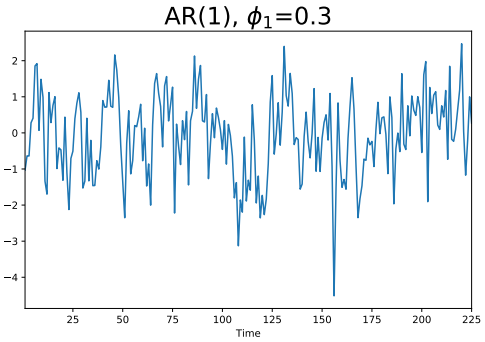
For simplicity, let's assume that the mean of the data series is equal to zero (i.e. $b_0 = 0$). Note that if we subtract the mean from any data series, the transformed data would have mean zero. Then we could write:

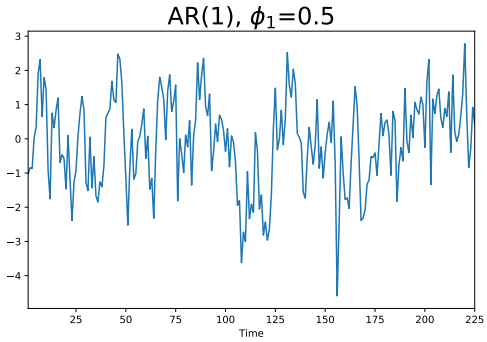
$$y_t = \phi_1 y_{t-1} + \varepsilon_t$$

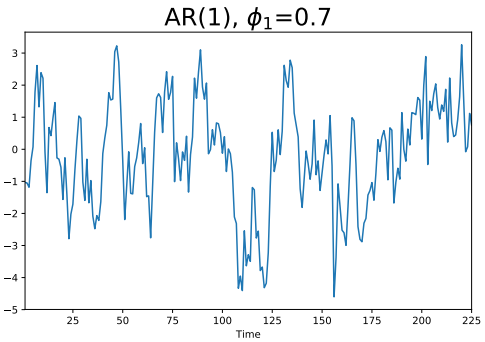
which is simpler to analyze. In the AR(1) model (with or without an intercept), ϕ_1 expresses the degree of autocorrelation at the first lag. If $\phi_1 > 0$ then there is positive autocorrelation (values above average tend to be followed by values above average). If $\phi_1 < 0$ then there is negative autocorrelation (values above average tend to be followed by values *below* average). If $\phi_1 = 0$, then there is no autocorrelation - the model collapses to the "simple" intercept model we encountered long ago. Finally, a common restriction placed on AR(1) models is that

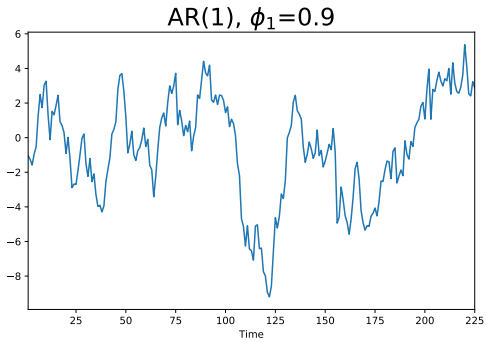
$-1 < \phi_1 < 1$. This restriction is appropriate if the data is stationary.

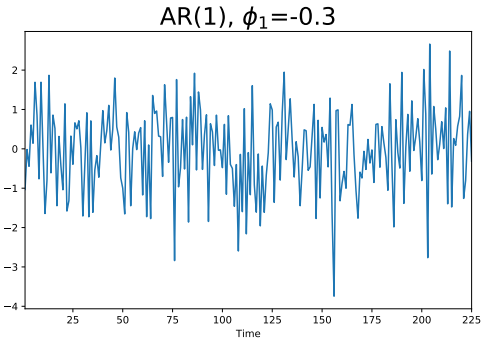
In an AR(1) model, by changing the value of ϕ_1 we can match a wide variety of time-series data. Below are four examples of data created with positive values of ϕ_1 and four examples of data created with negative values of ϕ_1. Note that the larger $|\phi_1|$, the more autocorrelation there is.

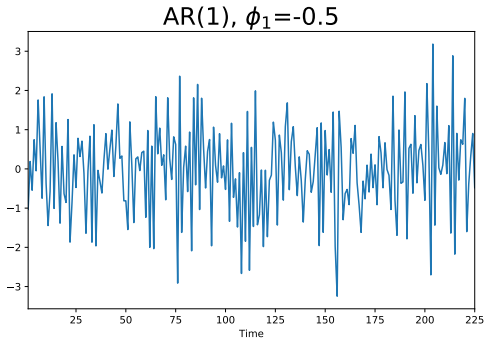


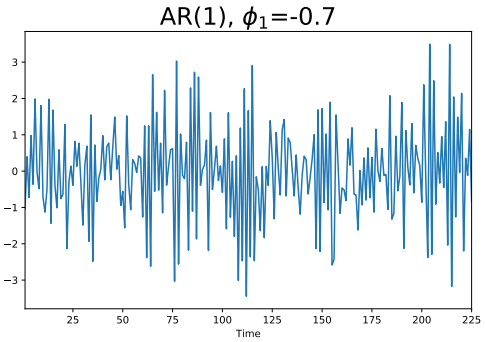


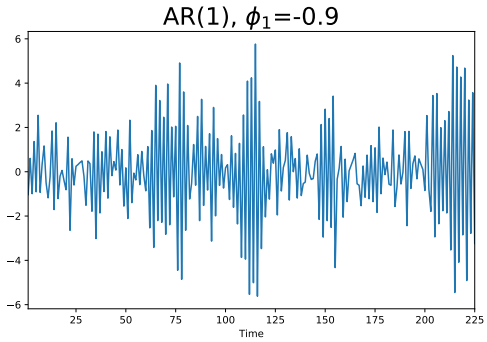












2.1.1 Forecasting with AR(1)

Let's consider optimal forecasts from an AR(1) model. Suppose we wanted to forecast one-period ahead, for time period $T+1$. Then, we would write the model as:

$$y_{T+1} = \phi_1 y_T + \varepsilon_{T+1}$$

Under squared error loss, we can use the expected value operator to find the optimal forecast:

$$\begin{aligned}
E(y_{T+1}|Y) &= E(\phi_1 y_T + \varepsilon_{T+1}|Y) \\
E(y_{T+1}|Y) &= E(\phi_1 y_T|Y) + E(\varepsilon_{T+1}|Y) \\
E(y_{T+1}|Y) &= \phi_1 E(y_T|Y) + 0 \\
E(y_{T+1}|Y) &= \phi_1 y_T \\
\hat{y}_{T+1|Y} &= \phi_1 y_T
\end{aligned}$$

Let's continue by assuming we want to form the two-period ahead forecast. From the model equation, the data at time $T+2$ can be written as:

$$y_{T+2} = \phi_1 y_{T+1} + \varepsilon_{T+2}$$

Under squared error loss, we can use the expected value operator to find the optimal forecast:

$$\begin{aligned}
E(y_{T+2}|Y) &= E(\phi_1 y_{T+1} + \varepsilon_{T+2}|Y) \\
E(y_{T+2}|Y) &= E(\phi_1 y_{T+1}|Y) + E(\varepsilon_{T+2}|Y) \\
E(y_{T+2}|Y) &= \phi_1 E(y_{T+1}|Y) + 0 \\
E(y_{T+2}|Y) &= \phi_1(\phi_1 y_T) \\
\hat{y}_{T+2|Y} &= \phi_1^2 y_T
\end{aligned}$$

In general, the h-period ahead forecast from an AR(1) model can be written:

$$\hat{y}_{T+h|Y} = \phi_1^h y_T$$

Essentially the AR(1) model with $|\phi_1| < 1$ always forecasts mean-reversion. If we were 10 units above average at the end of the sample, we will be $\phi_1(10)$ units above average during the next period. As our forecast horizon increases, our forecasts move closer and closer to the mean of the observed data. In other words, the long-run forecasts from the AR(1) model will approach the in-sample average of the data.

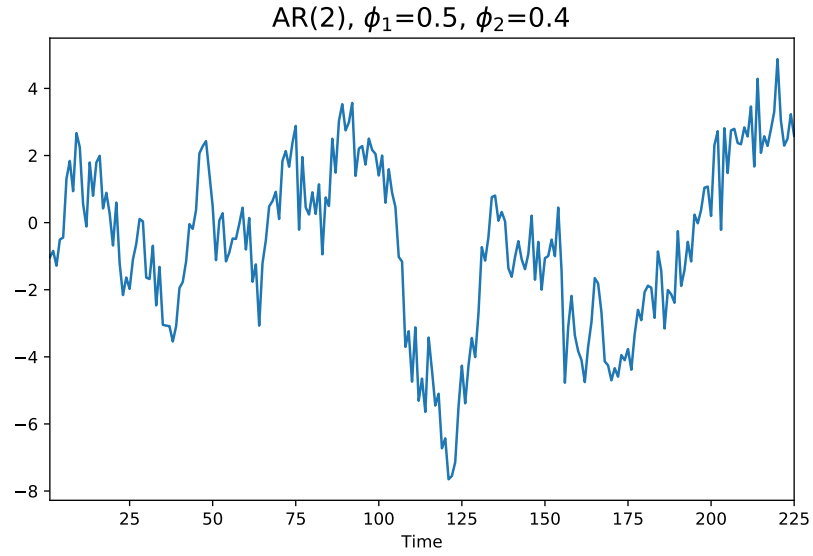
2.2 AR(2) Models

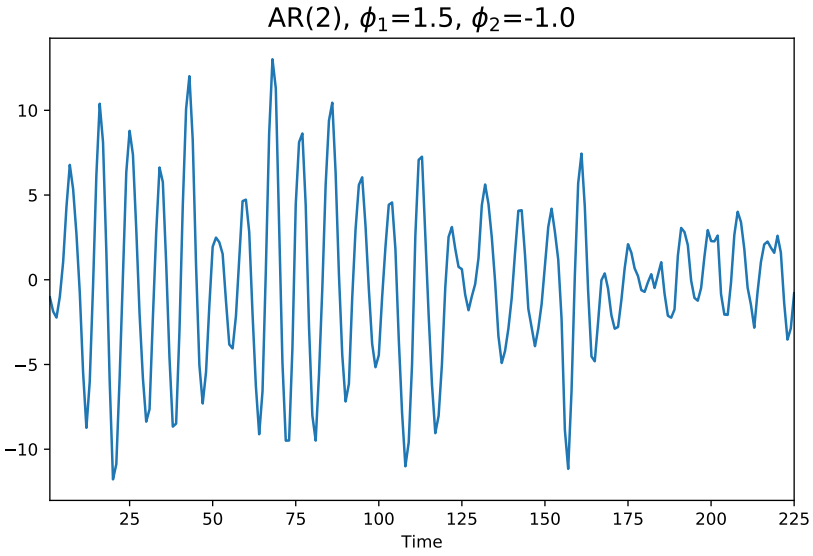
Now let's consider an AR(2) model. An AR(2) model says that the data at time t depends on the previous two lags of the data:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \varepsilon_t$$
$$\varepsilon_t \sim N(0, \sigma)$$

This AR(2) model can fit an even wider variety of time series than the AR(1) model. For example, it can forecast series that have short-term momentum — series that if they were above average at time period $t-1$ might be even *more* above average at time period t, before declining back towards the mean of the series. Like an AR(1) model, in the long-run, we expect that the data will return to its in-sample mean.

Also note that we can fit a wider variety of data with an AR(2) model. I have included examples of data simulated with two different AR(2) parameters to give you a sense of what the time series can look like when an AR(2) model is appropriate.





2.2.1 Forecasing with AR(2)

Suppose we forecasted with the AR(2) model. The $h = 1$ period ahead forecast would be fairly simple to find. Let's plug $t = T + 1$ into the AR(2) equation. We have:

$$y_{T+1} = \phi_1 y_T + \phi_2 y_{T-1} + \varepsilon_{T+1}$$

Now, since we will compute the optimal forecast under squared error loss, let's take the expected value (conditional on observing the data, Y) of both sides:

$$E(y_{T+1}|Y) = E(\phi_1 y_T + \phi_2 y_{T-1} + \varepsilon_{T+1}|Y)$$
$$E(y_{T+1}|Y) = E(\phi_1 y_T|Y) + E(\phi_2 y_{T-1}|Y) + E(\varepsilon_{T+1}|Y)$$

Treating ϕ_1 and ϕ_2 as constants, we have:

$$E(y_{T+1}|Y) = \phi_1 E(y_T|Y) + \phi_2 E(y_{T-1}|Y) + 0$$

Finally note that both y_T and y_{T-1} are observed data points. Therefore we have:

$$E(y_{T+1}|Y) = \phi_1 y_T + \phi_2 y_{T-1}$$
$$\hat{y}_{T+1|T} = \phi_1 y_T + \phi_2 y_{T-1}$$

The optimal point forecasts for $h = 2, \cdots$ are more mathematically challenging without using matrices, so I will skip over the details. However, the process is generally the same as for the AR(1) forecasts, with the long-run forecast converging to the in-sample mean of the data series.

In general, the same will be true of forecasts from any AR(p) forecast. While there may be some very short-term momentum, with the $h = 1$ or $h = 2$ forecasts being farther from the mean than y_T, the long-run forecasts from these models will converge to the in-sample mean of Y.

2.3 Examples of Forecasting with AR Models

Consider the following data set of quarterly US Real GDP growth (annualized) since 1985. We will end the data during the Great Recession to make it more clear how the forecasts from AR and MA models differ.

```
# Read in Data
data <- fredr(
series_id = "GDPC1",
observation_start = as.Date("1985-01-01"),
observation_end = as.Date("2008-10-01")
```

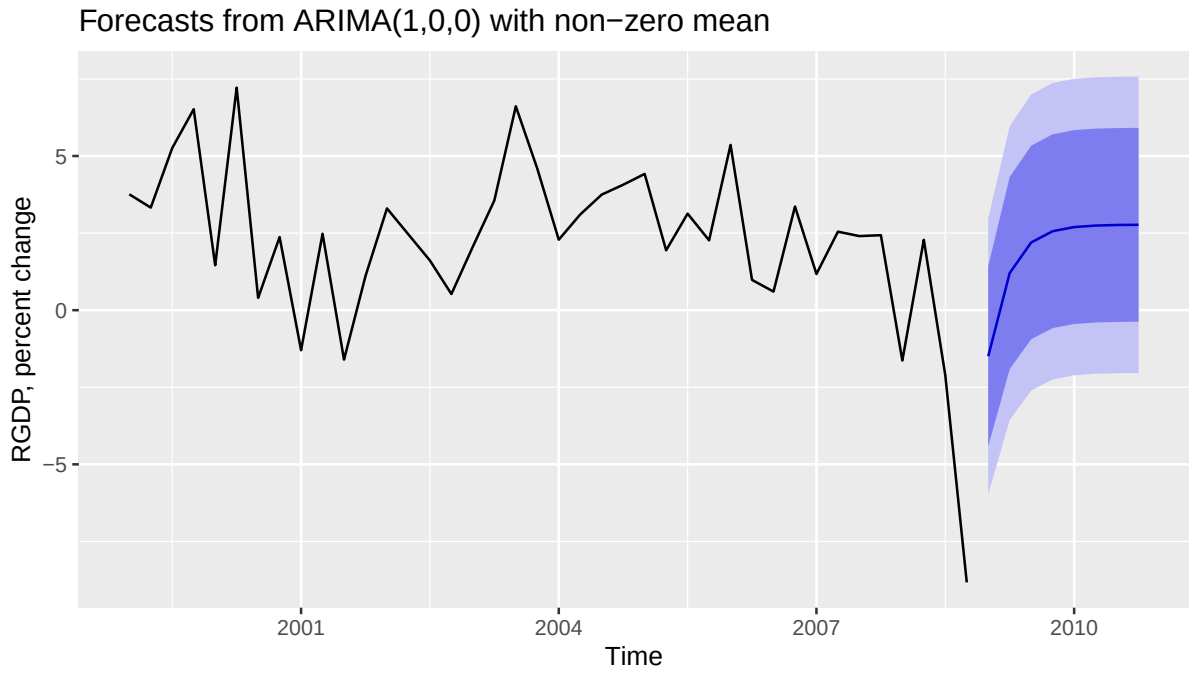
```
)
  # Declare as Time Series
  Y <- ts(data$value,start=c(1985,1),frequency = 4)
  Y <- diff(400*log(Y))
```

Let's compute the mean of RGDP. Again, in the long-run, we should expect forecasts from any AR model to approach the in-sample mean of our data.

```
  # Calculate mean of data
  # This is roughly what the long-run AR forecasts will converge to
  mean(Y)

## [1] 2.839279
```

Let's use an AR(1) model to forecast future RGDP growth, and plot the forecasts.

```
  # Fit AR(1) Model
  fit_AR <- Arima(Y,order=c(1,0,0))
  # Forecast with AR(1) Model
  fcst_AR <- forecast(fit_AR,h=8)
  # Time plot of data & forecasts
  autoplot(fcst_AR,include=40) + ylab("RGDP, percent change")
```



Note that we can inspect the results of the model (including the estimate for ϕ_1, the AICc of the AR model, and the point forecasts)

```
  # Summarize results of AR model
  summary(fcst_AR)
```

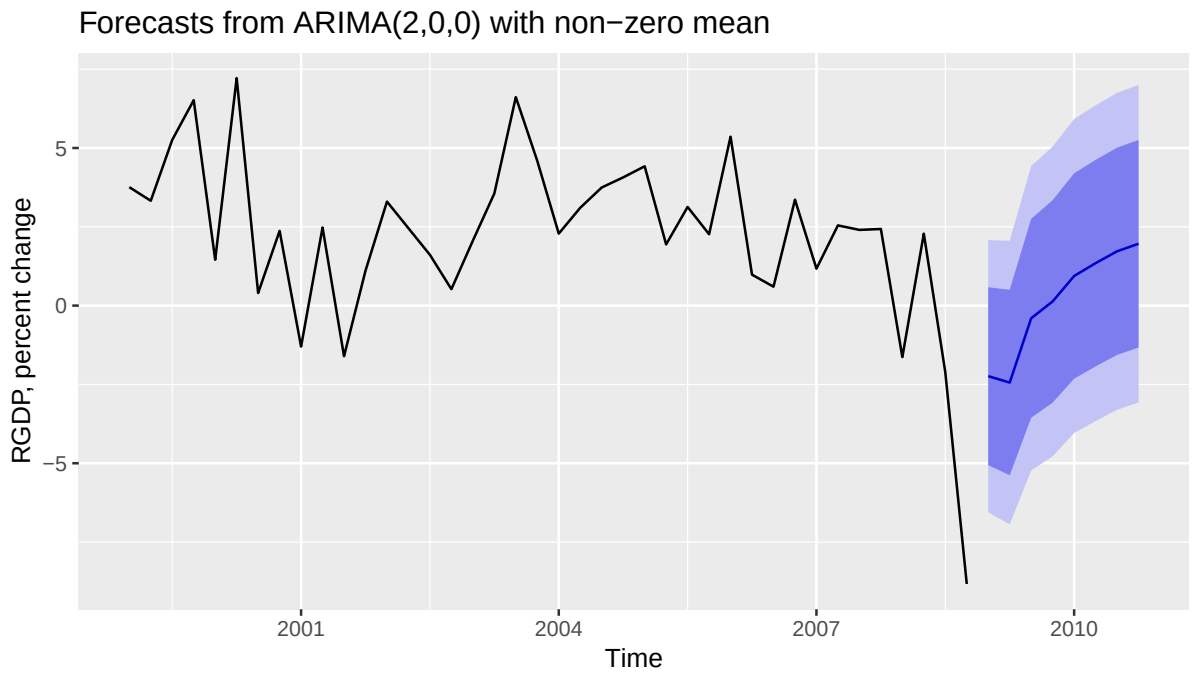
```
##
## Forecast method: ARIMA(1,0,0) with non-zero mean
##
## Model Information:
## Series: Y
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##          ar1    mean
##       0.3678  2.7727
## s.e.  0.1115  0.3654
##
## sigma^2 estimated as 5.203:  log likelihood=-212.2
## AIC=430.39   AICc=430.66   BIC=438.06
##
## Error measures:
##                       ME     RMSE      MAE      MPE     MAPE      MASE
## Training set -0.003381592 2.256815 1.674165 -18.5393 84.18066 0.7721016
##                     ACF1
## Training set -0.08304566
##
## Forecasts:
##         Point Forecast      Lo 80    Hi 80     Lo 95    Hi 95
## 2009 Q1      -1.493973 -4.4171315 1.429186 -5.964558 2.976612
## 2009 Q2       1.203606 -1.9109614 4.318174 -3.559714 5.966926
## 2009 Q3       2.195670 -0.9438893 5.335230 -2.605872 6.997212
```

```
## 2009 Q4       2.560513 -0.5824118 5.703437 -2.246175 7.367201
## 2010 Q1       2.694687 -0.4486919 5.838066 -2.112696 7.502071
## 2010 Q2       2.744032 -0.3994092 5.887472 -2.063446 7.551509
## 2010 Q3       2.762178 -0.3812706 5.905627 -2.045312 7.569669
## 2010 Q4       2.768852 -0.3745980 5.912302 -2.038640 7.576344
```

Notice that since RGDP growth was below average in the 4th quarter of 2008 (-8.83% in that quarter vs. the 2.84% average), and $\phi_1 = 0.3678 > 0$, we forecast RGDP growth to be below average in the 1st quarter of 2009. In the long-run, the point-forecast gets closer and closer to the average of the series.

Let's do the same with a AR(2) model:

```
# Fit AR(1) Model
fit_AR <- Arima(Y,order=c(2,0,0))
# Forecast with AR(1) Model
fcst_AR <- forecast(fit_AR,h=8)
# Time plot of data & forecasts
autoplot(fcst_AR,include=40) + ylab("RGDP, percent change")
```



```r
# Summarize results of AR model
summary(fcst_AR)
```

```
##
## Forecast method: ARIMA(2,0,0) with non-zero mean
##
## Model Information:
## Series: Y
## ARIMA(2,0,0) with non-zero mean
##
## Coefficients:
##          ar1     ar2    mean
##       0.2937  0.3184  2.6511
## s.e.  0.1092  0.1137  0.5686
##
## sigma^2 estimated as 4.848:  log likelihood=-208.47
## AIC=424.94   AICc=425.39   BIC=435.16
##
## Error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.02114087 2.166862 1.606533 -23.01841 82.63953 0.7409105
##                      ACF1
## Training set 0.003659303
##
## Forecasts:
##         Point Forecast     Lo 80     Hi 80     Lo 95    Hi 95
## 2009 Q1     -2.2377732 -5.059632 0.5840852 -6.553433 2.077887
## 2009 Q2     -2.4401712 -5.381231 0.5008888 -6.938134 2.057792
## 2009 Q3     -0.4009224 -3.555890 2.7540456 -5.226030 4.424185
## 2009 Q4      0.1335928 -3.077791 3.3449762 -4.777794 5.044980
## 2010 Q1      0.9398889 -2.316515 4.1962929 -4.040351 5.920129
## 2010 Q2      1.3469013 -1.928184 4.6219869 -3.661910 6.355712
```

```
## 2010 Q3      1.7231734 -1.563388 5.0097350 -3.303189 6.749536
## 2010 Q4      1.9632839 -1.328871 5.2554386 -3.071632 6.998200
```

3 Moving Average (MA) Models

Moving Average (MA) models can be written in a very similar fashion as AR models. However, instead of including past values of the data on the right hand side, we instead include past values of the residuals.[1] In terms of matching features of the data, the major difference between an AR(p) model and an MA(q) model is that under an AR model autocorrelation at different lags declines gradually, while under an MA(q) model autocorrelation drops off suddenly after the q^{th} lag.

In general, an MA(q) model can be written as:

$$y_t = b_0 + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}$$

where b_0 is the intercept (which we will ignore for the time being — again, if we siply subtract the mean from our data series, we can ignore the mean in our calculations).

3.1 MA(1) Model

For example, ignoring the intercept term, an MA(1) model could be written as:

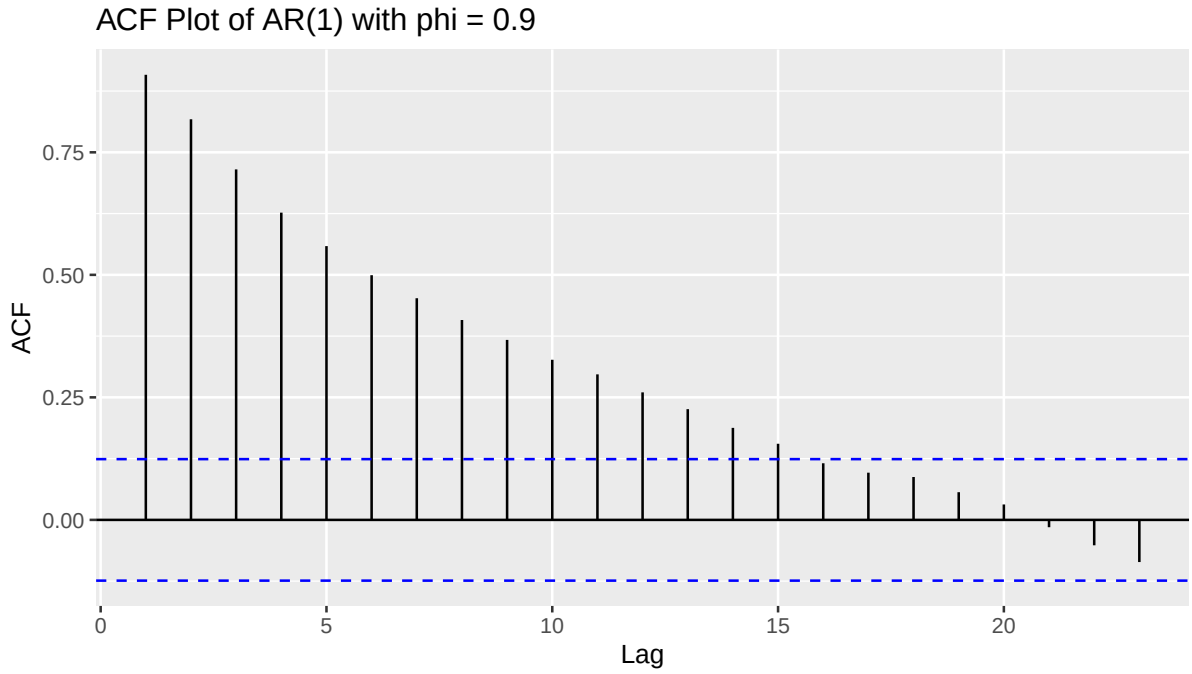
$$y_t = \varepsilon_t + \theta_1 \varepsilon_{t-1}$$

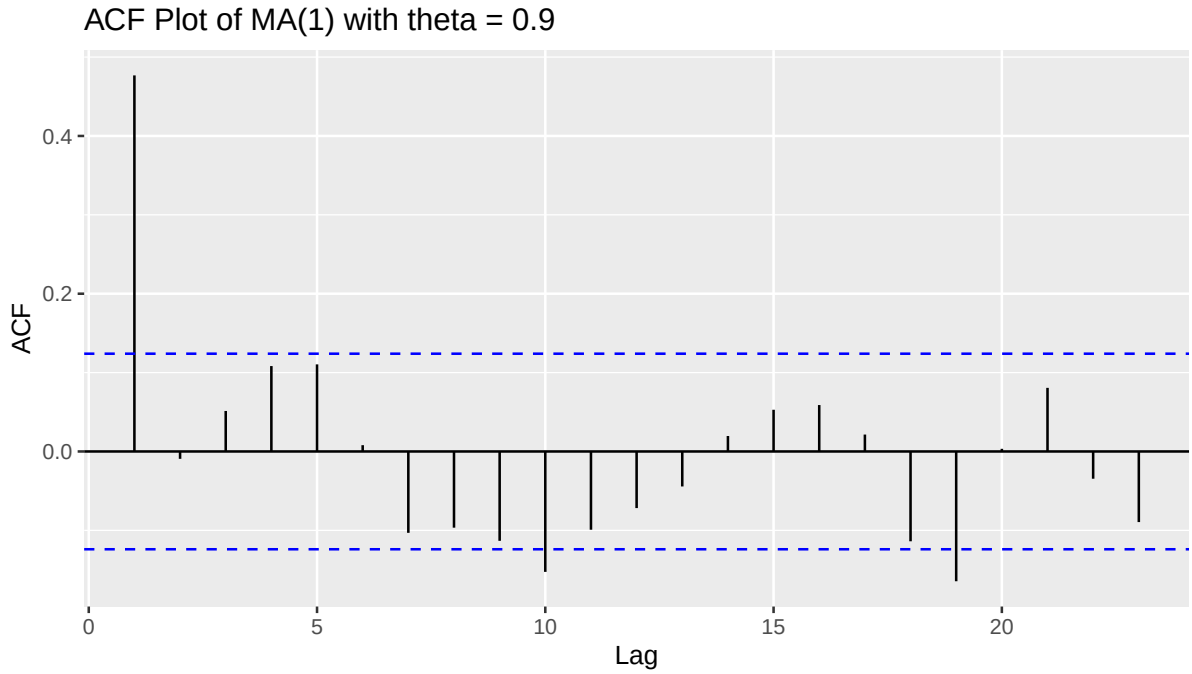
In words, the value of the data in period "t" depends on the random error (i.e. "shock") today, ε_t, plus a constant times the shock in the previous period. The constant, θ_1 (pronounced "theta one") expresses the amount of correlation at the first lag. Similarly to the AR(1) model, in this MA(1) model, $\theta_1 > 0$ means there is positive autocorrelation at the first lag, $\theta_1 < 0$ means there is negative autocorrelation at the first lag, and $\theta_1 = 0$ means there is no autocorrelation at the first lag. The major difference between the AR(1) and MA(1) models is that the MA(1) model will only be appropriate if the data has non-zero correlation *only* at the first lag. If there

[1] Note that this definition of a moving average is the one commonly used in time-series econometrics, but in other fields "moving average" may refer to taking the average of the past k data points. We will not use this type of "moving average" in this course.

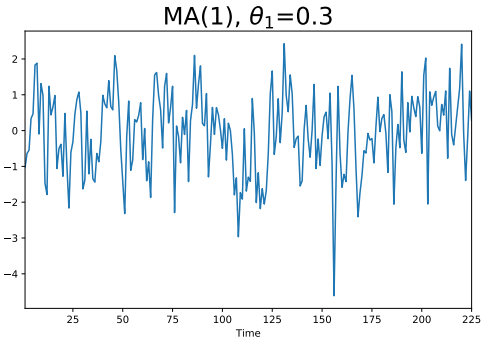
is any correlation at lags greater than one, then you should either use a higher order MA model or an AR model.

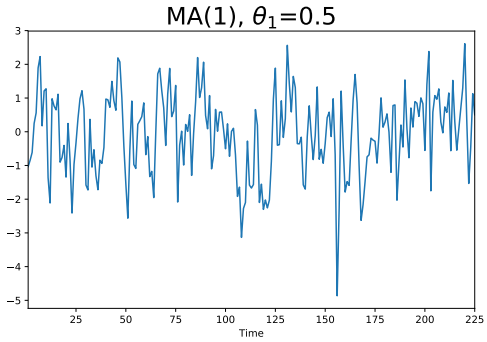
Below, I plotted an ACF from simulated data from an AR(1) with $\phi_1 = 0.9$ and an MA(1) with $\theta_1 = 0.9$. Note that under the MA(1) model, the autocorrelation drops off sharply after the first lag, while under the AR(1) model the autocorrelation declines slowly and geometrically.

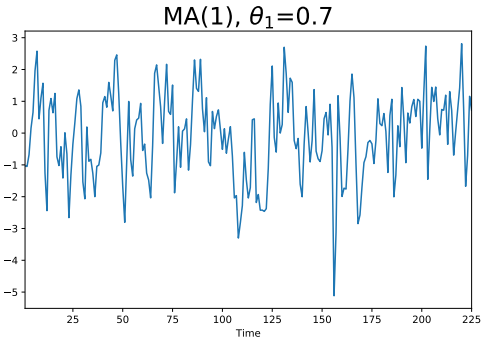


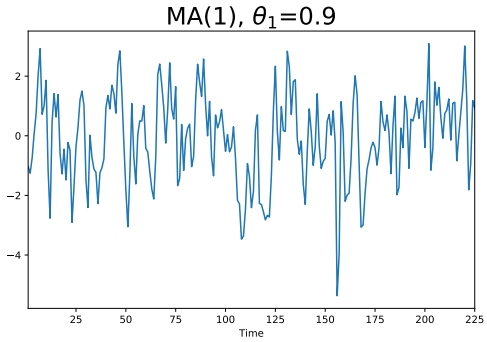


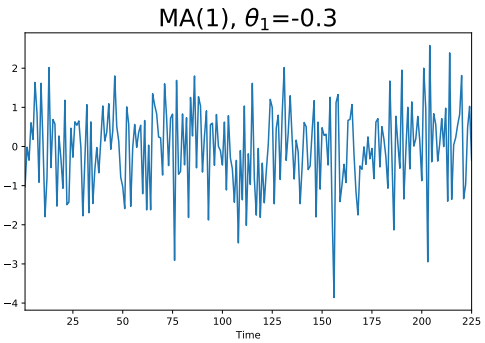
Like AR models, MA models can fit a wide variety of time series. I simulated data from different MA(1) and MA(2) models so you can get a sense of the types of time series for which MA models are appropriate.

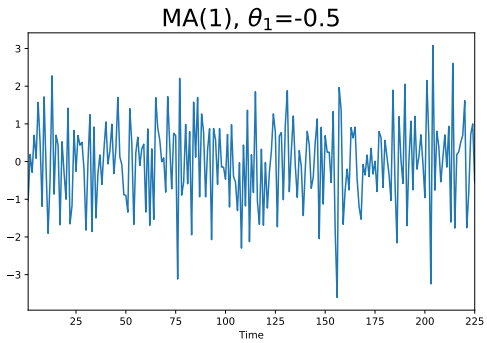


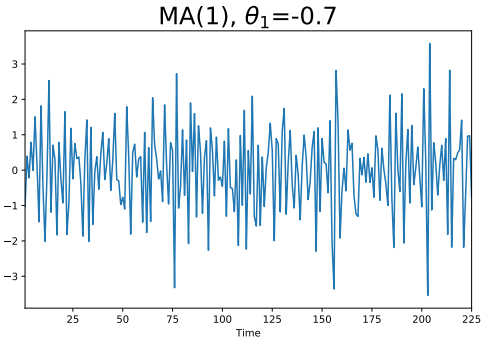


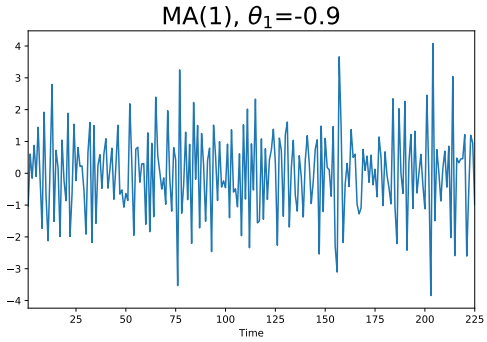


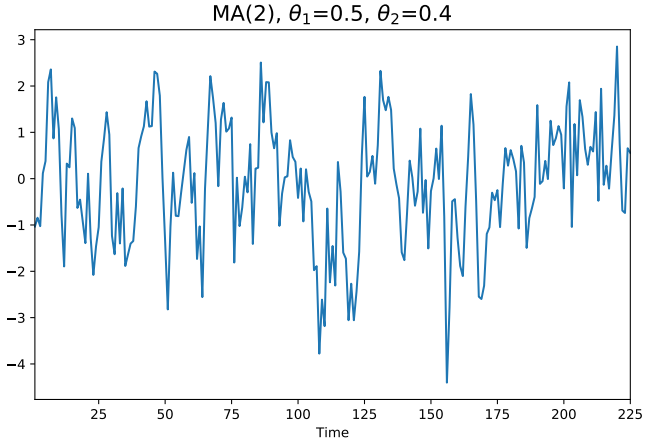


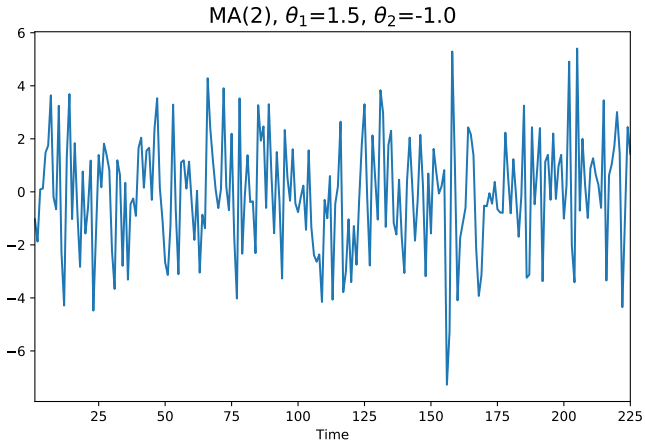












3.1.1 Forecasting with MA(1)

Let's consider forecasts from an MA(1) model for $h = 1$ period ahead. First substitute $t = T+1$ into the MA(1) equation:

$$y_{T+1} = \theta_1 \varepsilon_T + \varepsilon_{T+1}$$

To find the optimal forecast under squared error loss, take the expected value of both sides:

$$\begin{aligned}
E(y_{T+1}|Y) &= E(\theta_1 \varepsilon_T + \varepsilon_{T+1}|Y) \\
E(y_{T+1}|Y) &= E(\theta_1 \varepsilon_T|Y) + E(\varepsilon_{T+1}|Y) \\
E(y_{T+1}|Y) &= \theta_1 E(\varepsilon_T|Y) + 0 \\
E(y_{T+1}|Y) &= \theta_1 \varepsilon_T \\
\hat{y}_{T+1|T} &= \theta_1 \varepsilon_T
\end{aligned}$$

For the $h = 2$ period ahead forecast, plug in $t = T + 2$ into the MA(1) equation:

$$y_{T+2} = \theta_1 \varepsilon_{T+1} + \varepsilon_{T+2}$$

To find the optimal forecast under squared error loss, take the expected value of both sides:

$$\begin{aligned}
E(y_{T+2}|Y) &= E(\theta_1 \varepsilon_{T+1} + \varepsilon_{T+2}|Y) \\
E(y_{T+2}|Y) &= E(\theta_1 \varepsilon_{T+1}|Y) + E(\varepsilon_{T+2}|Y) \\
E(y_{T+2}|Y) &= \theta_1 E(\varepsilon_{T+1}|Y) + 0 \\
E(y_{T+2}|Y) &= 0 + 0 \\
\hat{y}_{T+2|T} &= 0
\end{aligned}$$

Above, we went from lines three to four by noting that $E(\varepsilon_{T+1}|Y) = 0$.

3.1.2 Forecasting with MA(4)

In general, when forecasting h-periods ahead from an MA(q) model, the forecast for $h \leq q$ will need to be calculated using at least some of the $\theta\varepsilon$ terms. However, the forecast for $h > q$ will be exactly the mean of the series.

To help see this, consider the $h = 4$ and $h = 5$ period ahead forecasts from an MA(4) model. First the $h = 4$ forecast — plug in $t = T + 4$

$$\begin{aligned}
y_t &= \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \theta_3 \varepsilon_{t-3} + \theta_4 \varepsilon_{t-4} + \varepsilon_t \\
y_{T+4} &= \theta_1 \varepsilon_{T+3} + \theta_2 \varepsilon_{T+2} + \theta_3 \varepsilon_{T+1} + \theta_4 \varepsilon_T + \varepsilon_{T+4} \\
E(y_{T+4}|Y) &= E(\theta_1 \varepsilon_{T+3} + \theta_2 \varepsilon_{T+2} + \theta_3 \varepsilon_{T+1} + \theta_4 \varepsilon_T + \varepsilon_{T+4}|Y) \\
E(y_{T+4}|Y) &= E(\theta_1 \varepsilon_{T+3}|Y) + E(\theta_2 \varepsilon_{T+2}|Y) + E(\theta_3 \varepsilon_{T+1}|Y) + E(\theta_4 \varepsilon_T|Y) + E(\varepsilon_{T+4}|Y) \\
E(y_{T+4}|Y) &= \theta_1 E(\varepsilon_{T+3}|Y) + \theta_2 E(\varepsilon_{T+2}|Y) + \theta_3 E(\varepsilon_{T+1}|Y) + \theta_4 E(\varepsilon_T|Y) + E(\varepsilon_{T+4}|Y) \\
E(y_{T+4}|Y) &= \theta_1(0) + \theta_2(0) + \theta_3(0) + \theta_4 \varepsilon_T + 0 \\
E(y_{T+4}|Y) &= \theta_4 \varepsilon_T
\end{aligned}$$

For the $h = 5$ period ahead forecast:

$$y_t = \theta_1\varepsilon_{t-1} + \theta_2\varepsilon_{t-2} + \theta_3\varepsilon_{t-3} + \theta_4\varepsilon_{t-4} + \varepsilon_t$$

$$y_{T+5} = \theta_1\varepsilon_{T+4} + \theta_2\varepsilon_{T+3} + \theta_3\varepsilon_{T+2} + \theta_4\varepsilon_{T+1} + \varepsilon_{T+5}$$

$$E(y_{T+5}|Y) = E(\theta_1\varepsilon_{T+4} + \theta_2\varepsilon_{T+3} + \theta_3\varepsilon_{T+2} + \theta_4\varepsilon_{T+1} + \varepsilon_{T+5}|Y)$$

$$E(y_{T+5}|Y) = E(\theta_1\varepsilon_{T+4}|Y) + E(\theta_2\varepsilon_{T+3}|Y) + E(\theta_3\varepsilon_{T+2}|Y) + E(\theta_4\varepsilon_{T+1}|Y) + E(\varepsilon_{T+5}|Y)$$

$$E(y_{T+5}|Y) = \theta_1 E(\varepsilon_{T+4}|Y) + \theta_2 E(\varepsilon_{T+3}|Y) + \theta_3 E(\varepsilon_{T+2}|Y) + \theta_4 E(\varepsilon_{T+1}|Y) + E(\varepsilon_{T+5}|Y)$$

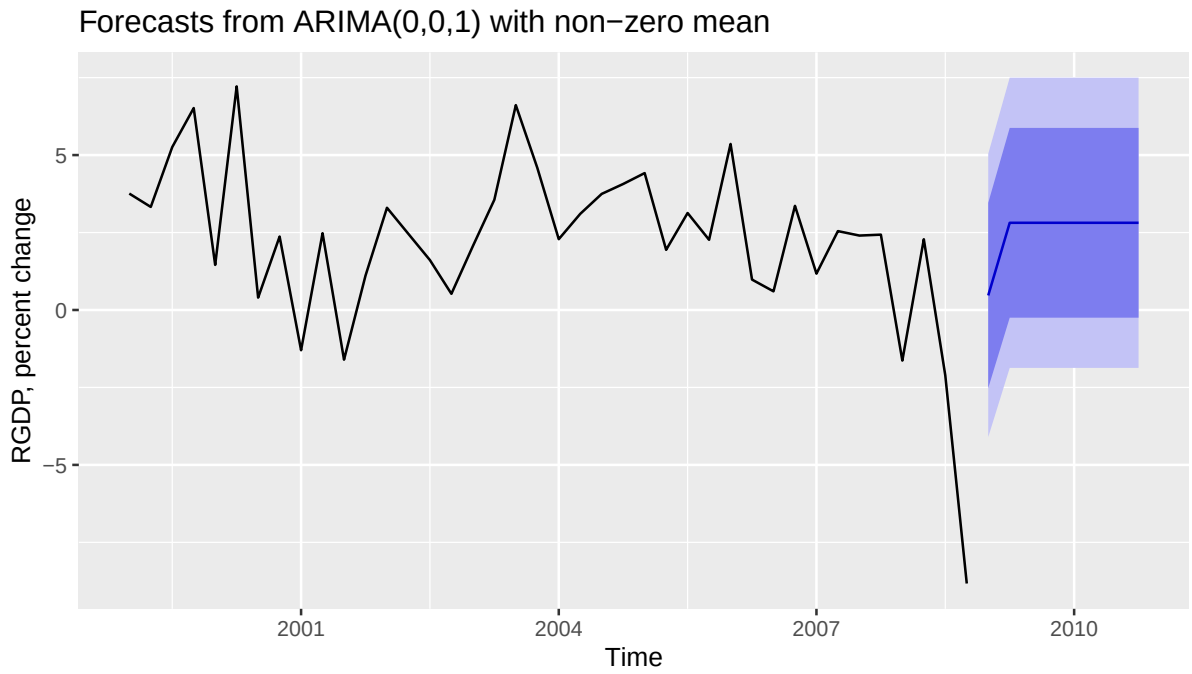
$$E(y_{T+5}|Y) = \theta_1(0) + \theta_2(0) + \theta_3(0) + \theta_4(0) + 0$$

$$E(y_{T+5}|Y) = 0$$

4 Examples of Forecasting with MA Models

Let's use an MA(1) model to forecast future RGDP growth, and plot the forecasts.

```
# Fit AR(1) Model
fit_MA <- Arima(Y,order=c(0,0,1))
# Forecast with AR(1) Model
fcst_MA <- forecast(fit_MA,h=8)
# Time plot of data & forecasts
autoplot(fcst_MA,include=40) + ylab("RGDP, percent change")
```




```
  # Summarize Forecasts from MA(1)
  summary(fcst_MA)
```

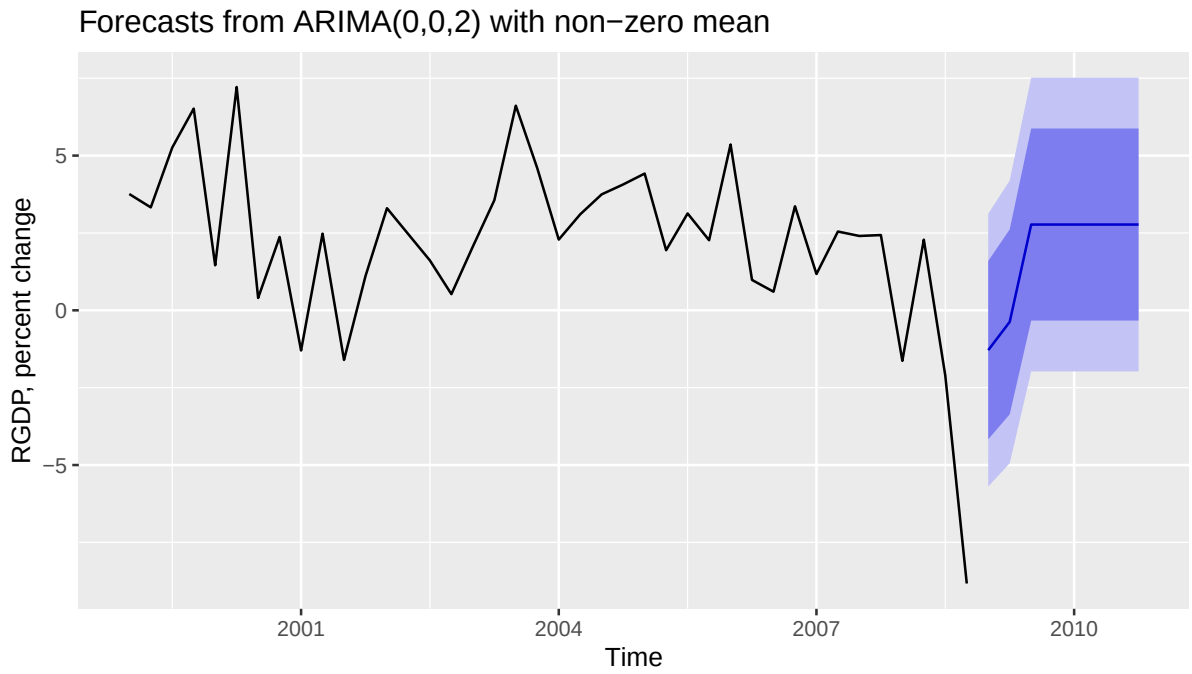
```
##
## Forecast method: ARIMA(0,0,1) with non-zero mean
##
## Model Information:
## Series: Y
## ARIMA(0,0,1) with non-zero mean
##
## Coefficients:
##          ma1    mean
##       0.2228  2.8145
## s.e.  0.0857  0.2891
##
## sigma^2 estimated as 5.44:  log likelihood=-214.27
## AIC=434.54   AICc=434.81   BIC=442.21
##
## Error measures:
```

```
##                         ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -6.057974e-05 2.307788 1.673984 -15.98341 83.35527 0.772018
##                     ACF1
## Training set 0.06327223
##
## Forecasts:
##         Point Forecast      Lo 80    Hi 80     Lo 95    Hi 95
## 2009 Q1      0.4694696 -2.5197122 3.458652 -4.102090 5.041029
## 2009 Q2      2.8145026 -0.2479973 5.877003 -1.869187 7.498192
## 2009 Q3      2.8145026 -0.2479973 5.877003 -1.869187 7.498192
## 2009 Q4      2.8145026 -0.2479973 5.877003 -1.869187 7.498192
## 2010 Q1      2.8145026 -0.2479973 5.877003 -1.869187 7.498192
## 2010 Q2      2.8145026 -0.2479973 5.877003 -1.869187 7.498192
## 2010 Q3      2.8145026 -0.2479973 5.877003 -1.869187 7.498192
## 2010 Q4      2.8145026 -0.2479973 5.877003 -1.869187 7.498192
```

Note that as expected, the point-forecast for periods $h = 2$ and beyond are all exactly equal to the mean of the series.

Now let's use an MA(2) model to forecast:

```
# Fit AR(1) Model
fit_MA <- Arima(Y,order=c(0,0,2))
# Forecast with AR(1) Model
fcst_MA <- forecast(fit_MA,h=8)
# Time plot of data & forecasts
autoplot(fcst_MA,include=40) + ylab("RGDP, percent change")
```




```
  # Summarize Forecasts from MA(1)
  summary(fcst_MA)
```

```
##
## Forecast method: ARIMA(0,0,2) with non-zero mean
##
## Model Information:
## Series: Y
## ARIMA(0,0,2) with non-zero mean
##
## Coefficients:
##          ma1     ma2    mean
##       0.2730  0.2916  2.7716
## s.e.  0.1184  0.1024  0.3537
##
## sigma^2 estimated as 5.057:  log likelihood=-210.37
## AIC=428.74   AICc=429.19   BIC=438.96
##
## Error measures:
```

```
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.005735359 2.212931 1.614716 -20.14895 81.47128 0.7446845
##                    ACF1
## Training set 0.0337427
##
## Forecasts:
##         Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## 2009 Q1      -1.287928 -4.169781 1.593925 -5.695341 3.119486
## 2009 Q2      -0.373123 -3.360459 2.614213 -4.941860 4.195614
## 2009 Q3       2.771627 -0.331623 5.874877 -1.974384 7.517638
## 2009 Q4       2.771627 -0.331623 5.874877 -1.974384 7.517638
## 2010 Q1       2.771627 -0.331623 5.874877 -1.974384 7.517638
## 2010 Q2       2.771627 -0.331623 5.874877 -1.974384 7.517638
## 2010 Q3       2.771627 -0.331623 5.874877 -1.974384 7.517638
## 2010 Q4       2.771627 -0.331623 5.874877 -1.974384 7.517638
```

5 Autoregressive Moving Average (ARMA) Models

ARMA models combine both AR and MA components in the same equation. As both AR and MA models are fairly flexible by themselves, ARMA models are extremely flexible and can fit data with almost any autocorrelation structure with only a small number of parameters. The only requirement is that when using these models, the data, Y, should be stationary.

We will write ARMA models as ARMA(p,q) — these models will have p lags of the data and q lags of the residual:

$$y_t = b_0 + \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

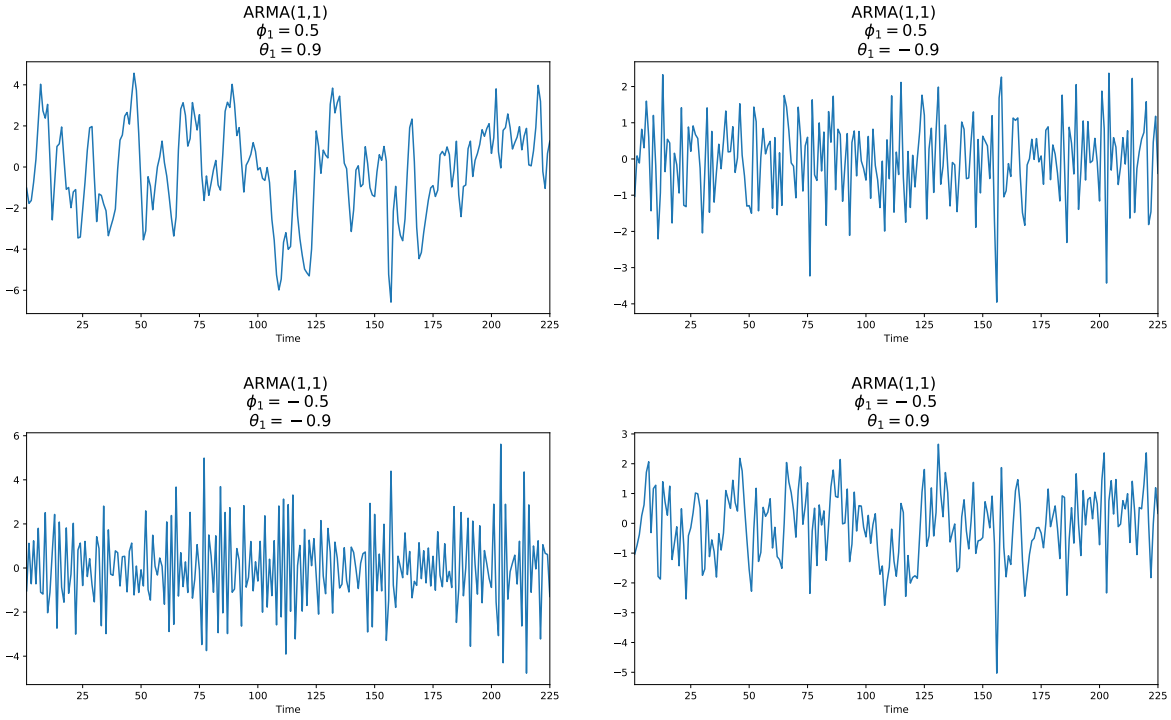
Like the AR and MA models, we will typically assume we are working with data that has been de-meaned (so that the mean is zero). Then we could ignore b_0 and write:

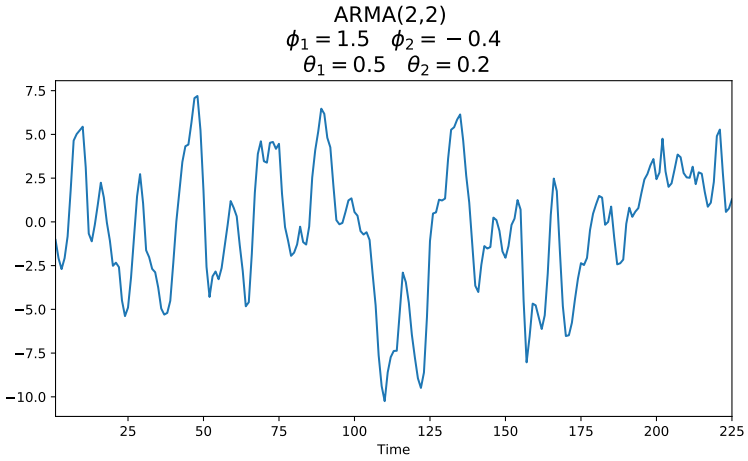
$$y_t = \phi_1 y_{t-1} + \cdots + \phi_p y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$$

For example, the ARMA(1,2) model would be written:

$$y_t = \phi_1 y_{t-1} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \varepsilon_t$$

As you can imagine, ARMA(p,q) models can fit an even wider variety of time series than either an AR(p) or MA(q) alone can. I simulated data from a couple of ARMA(p,q) models to show you the types of data that ARMA models can fit.



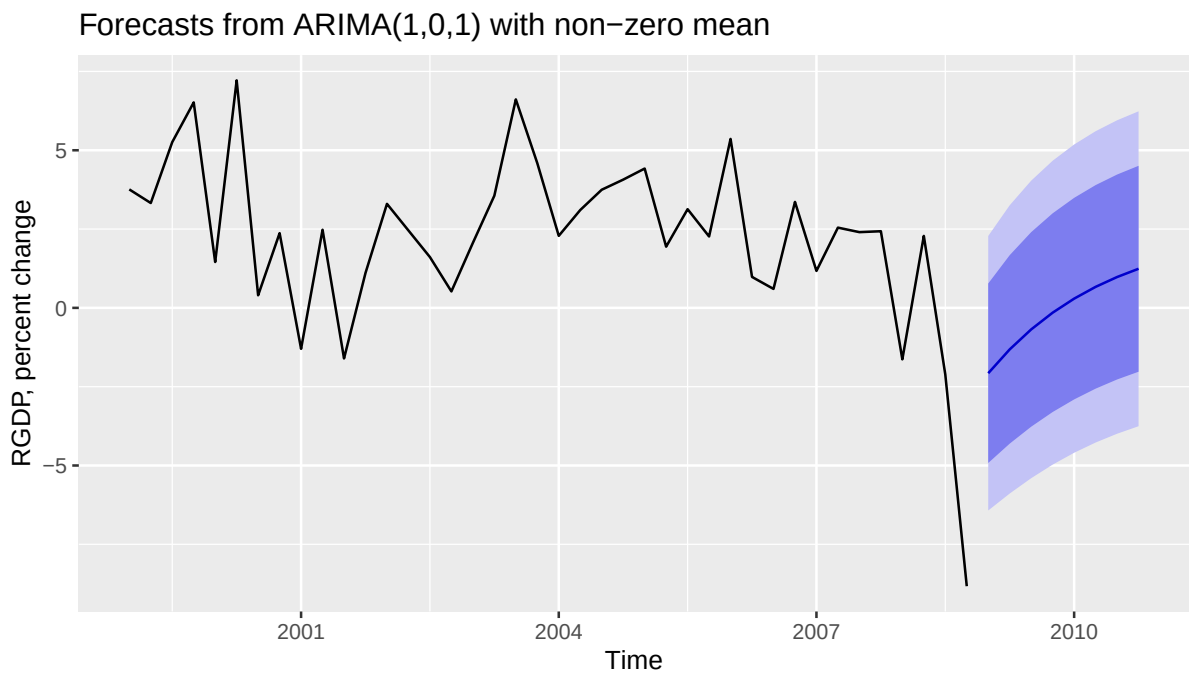


Forecasts from these models will take on the properties of forecasts of both the AR and MA models. If there is at least one AR component, the forecasts will slowly approach the mean of the series, with the contribution of the MA components completely dropping out once $h > q$

5.1 Examples of Forecasting with ARMA Models

Let's forecast with an ARMA(1,1) model:

```
# Fit AR(1) Model
fit_ARMA <- Arima(Y,order=c(1,0,1))
# Forecast with AR(1) Model
fcst_ARMA <- forecast(fit_ARMA,h=8)
# Time plot of data & forecasts
autoplot(fcst_ARMA,include=40) + ylab("RGDP, percent change")
```



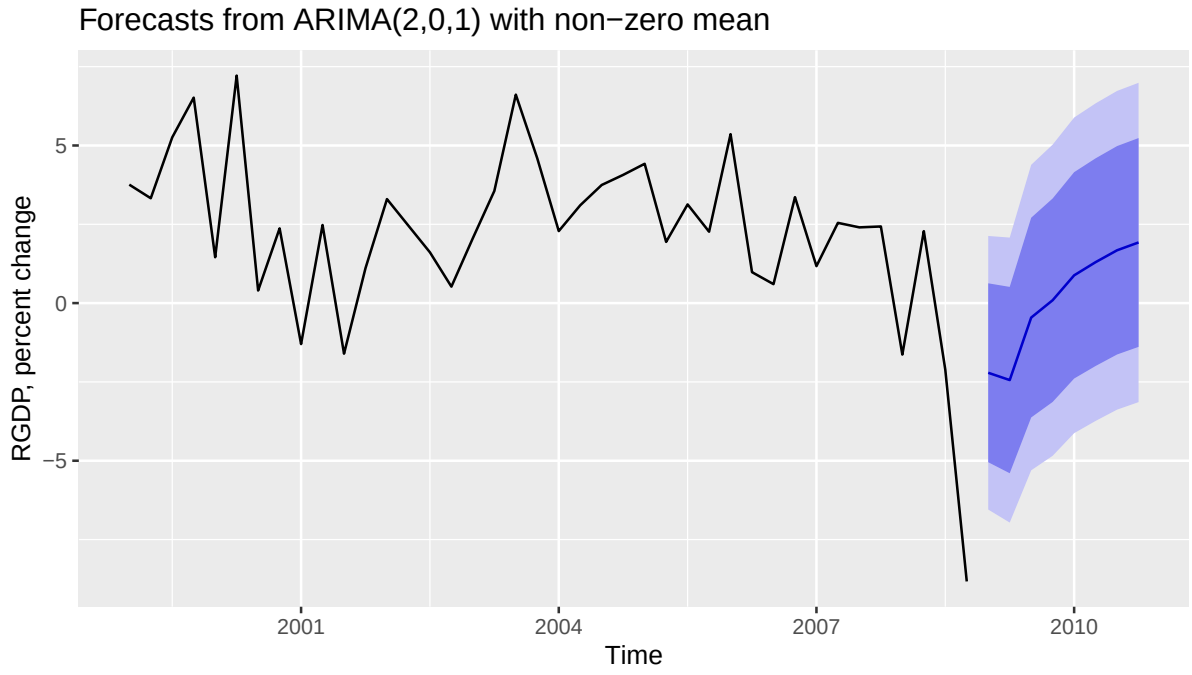

```
# Summarize Forecasts from MA(1)
summary(fcst_ARMA)
```

```
##
## Forecast method: ARIMA(1,0,1) with non-zero mean
##
```

```
## Model Information:
## Series: Y
## ARIMA(1,0,1) with non-zero mean
##
## Coefficients:
##          ar1      ma1    mean
##       0.8381  -0.5176  2.5972
## s.e.  0.1201   0.1569  0.6666
##
## sigma^2 estimated as 4.934:  log likelihood=-209.28
## AIC=426.57   AICc=427.01   BIC=436.78
##
## Error measures:
##                       ME     RMSE      MAE       MPE     MAPE     MASE
## Training set -0.0275876 2.185848 1.628772 -21.10033 83.85103 0.751167
##                      ACF1
## Training set -0.04093359
##
## Forecasts:
##         Point Forecast     Lo 80     Hi 80     Lo 95    Hi 95
## 2009 Q1     -2.0733031 -4.919887 0.7732807 -6.426778 2.280171
## 2009 Q2     -1.3171148 -4.306361 1.6721312 -5.888772 3.254542
## 2009 Q3     -0.6833583 -3.768869 2.4021524 -5.402240 4.035523
## 2009 Q4     -0.1522112 -3.303581 2.9991582 -4.971815 4.667392
## 2010 Q1      0.2929396 -2.903878 3.4897572 -4.596171 5.182050
## 2010 Q2      0.6660176 -2.562340 3.8943755 -4.271330 5.603365
## 2010 Q3      0.9786918 -2.271637 4.2290206 -3.992257 5.949641
## 2010 Q4      1.2407420 -2.024931 4.5064147 -3.753674 6.235157
```

Now from an ARMA(2,1) model:

```r
# Fit AR(1) Model
fit_ARMA <- Arima(Y,order=c(2,0,1))
# Forecast with AR(1) Model
fcst_ARMA <- forecast(fit_ARMA,h=8)
# Time plot of data & forecasts
autoplot(fcst_ARMA,include=40) + ylab("RGDP, percent change")
```




```r
# Summarize Forecasts from MA(1)
summary(fcst_ARMA)
```

```
##
## Forecast method: ARIMA(2,0,1) with non-zero mean
##
## Model Information:
## Series: Y
## ARIMA(2,0,1) with non-zero mean
##
## Coefficients:
##          ar1     ar2      ma1    mean
```

```
##       0.3142  0.3105  -0.0227  2.6482
## s.e.  0.5245  0.2325   0.5710  0.5791
##
## sigma^2 estimated as 4.902:  log likelihood=-208.47
## AIC=426.94   AICc=427.61   BIC=439.71
##
## Error measures:
##                       ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -0.0216657 2.166827 1.607897 -22.95995 82.70934 0.7415394
##                      ACF1
## Training set 0.004765582
##
## Forecasts:
##         Point Forecast     Lo 80     Hi 80     Lo 95    Hi 95
## 2009 Q1    -2.21003445 -5.047309 0.6272402 -6.549272 2.129203
## 2009 Q2    -2.44150495 -5.396829 0.5138196 -6.961284 2.078274
## 2009 Q3    -0.45914787 -3.626975 2.7086793 -5.303921 4.385626
## 2009 Q4     0.09175656 -3.135236 3.3187486 -4.843502 5.027015
## 2010 Q1     0.88030380 -2.392787 4.1533945 -4.125456 5.886064
## 2010 Q2     1.29907454 -1.994078 4.5922266 -3.737367 6.335516
## 2010 Q3     1.67546022 -1.629913 4.9808335 -3.379672 6.730592
## 2010 Q4     1.92372312 -1.387811 5.2352569 -3.140831 6.988277
```

6 Autoregressive Intergrated Moving Average (ARIMA) Models

An ARIMA model first transforms the data before using it in an ARMA model. This transformation means that you can use an ARIMA model with data that has a trend or a unit root.

The extra term in ARIMA, "I", stands for "integrated". This "I" parameter will be a non-negative integer (typically either 0 or 1) that represents how many times the data has been differenced before being used in an ARMA model. For example, if your data has a unit root, you must take the first difference of the data before it can be used in the ARMA model. Therefore,

I=1, since you are taking one difference of the data. If your data does not have a unit root, you do not need to take any differences and I=0, since you need to take zero differences of the data before using it in the ARMA model.

Again, if your data has a unit root or a trend, we will "first difference" the data (and therefore set I=1). We could do this manually and then use the adjusted data (which is now stationary) in an ARMA model, or we could use the ARIMA notation in RStudio and have the computer do the adjustment for us.

We write ARIMA models as ARIMA(p,d,q), where d is the number of differences we will take. For example, consider the ARIMA(2,1,2) model. Since $d = 1$ we need to transform the data, Y, by taking the first-difference:

$$\Delta y_t = y_t - y_{t-1}$$

for each time period, $t = 2, 3, 4, \cdots, T$. Then we could write the ARIMA(2,1,2) equation as:

$$\Delta y_t = \phi_1 \Delta y_{t-1} + \phi_2 \Delta y_{t-2} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \varepsilon_t$$

Below are examples of data generated by ARIMA(1,1,1) models. Note that if the mean of ΔY is NOT equal to zero, then the series will have a trend over time. We call these models "ARIMA with drift," and I've included two examples below.

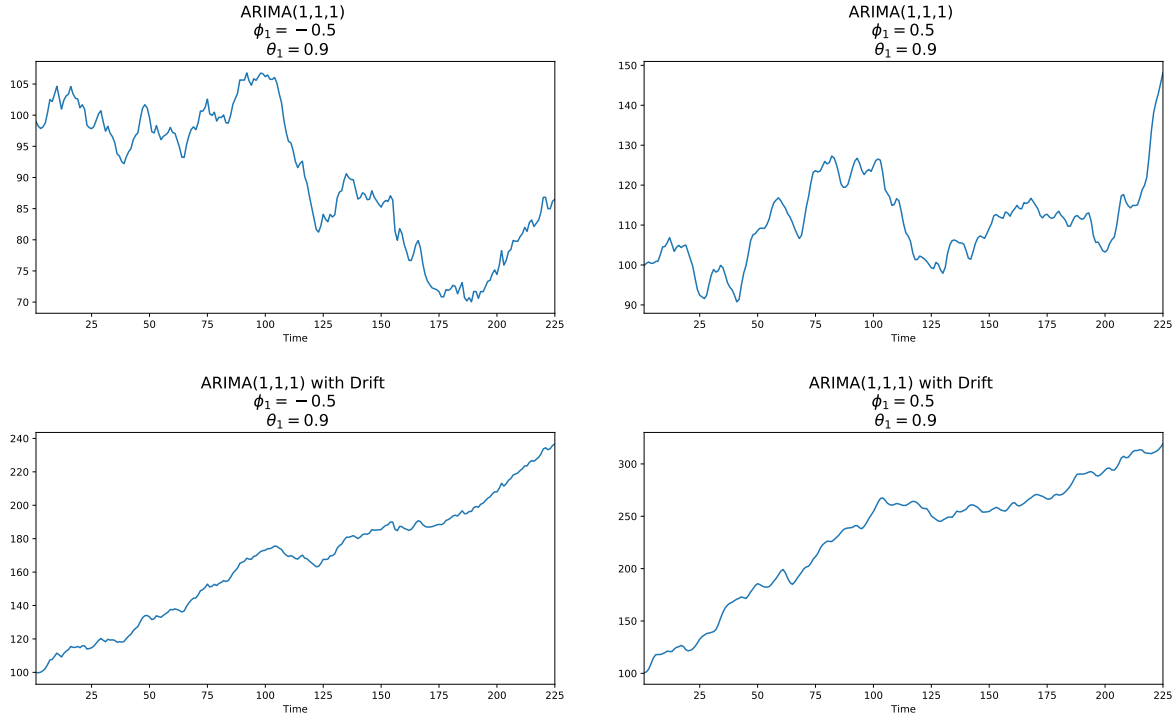
6.1 Testing for a Unit Root

It may not always be obvious when a time series has a unit root. Therefore it is often useful to use a statistical test to see if the series has a unit root. Recall that you have already seen model that has a unit root — the random walk model:

$$y_t = y_{t-1} + \varepsilon_t$$

Note that the random walk model is a special case of the AR(1) model, with $\phi_1 = 1$

$$y_t = \phi_1 y_{t-1} + \varepsilon_t$$



If $\phi_1 = 1$ Then:

$$y_t = (1)y_{t-1} + \varepsilon_t$$

$$y_t = y_{t-1} + \varepsilon_t$$

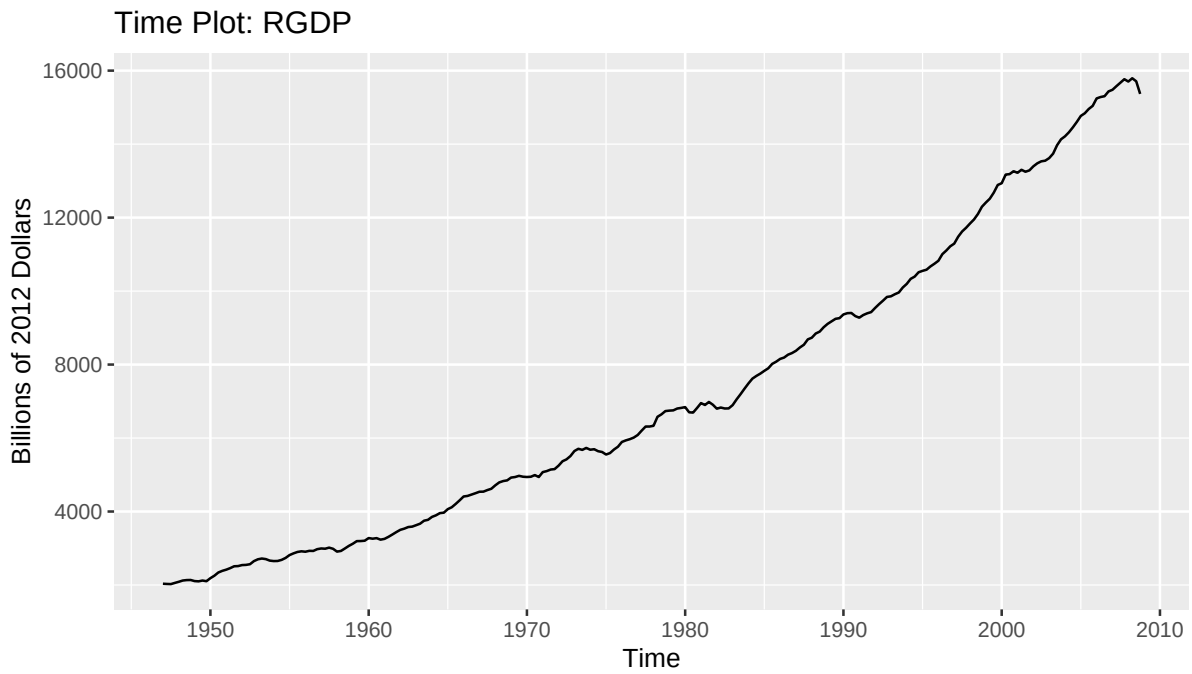
Therefore, one idea to test for a unit root is to estimate an AR(1) model and test the value of ϕ_1. If $\phi_1 = 1$ the series has a unit root, but if $\phi_1 < 1$ then the series does not have a unit root.

This is the logic underlying the **Augmented Dickey-Fuller test**, a statistical test for a unit root. Under the Null Hypothesis (H_0) of this test, the series has a unit root ($\phi_1 = 1$). Under the Alternative Hypothesis (H_1), the series does NOT have a unit root $\phi_1 < 1$. There are some more advanced details that I will skip over here, but that is the essence of the test. **If the p-value < 0.05, we reject the null hypothesis and conclude that there is NOT a unit root.** Equivalently, if the test statistic is **less** than the critical value, we will **reject** the null hypothesis. If the p-value is ≥ 0.05, or if the test statistic is $\geq$ the critical value, then we cannot reject the null and are forced to act as if there is a unit root, and take the first difference of the data (i.e. set "I=1" in the ARIMA model).

6.2 Example of Forecasting with ARIMA

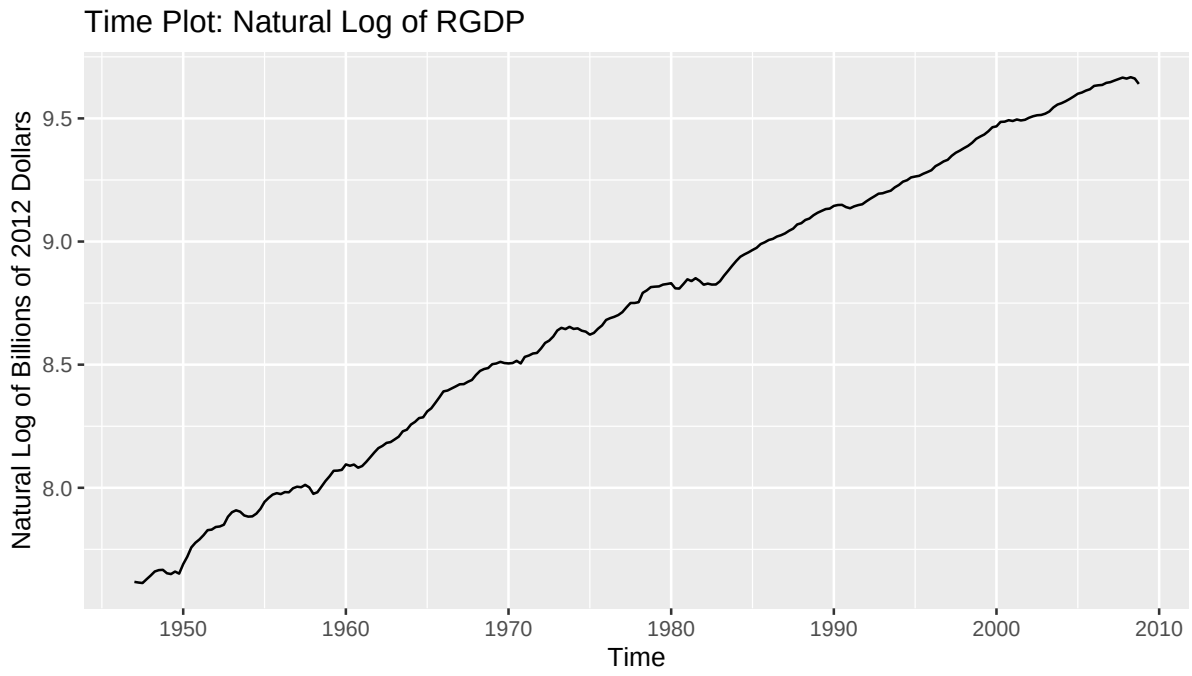
Suppose we have downloaded the raw RGDP data since 1947, which is in dollars. Like above, we will stop this data in 2008 Q4 to focus on forecasts during the Great Recession. Looking at a time plot, we would have:

```
  # Read in Data
  data <- fredr(
  series_id = "GDPC1",
  observation_start = as.Date("1947-01-01"),
  observation_end = as.Date("2008-10-01")
)
  # Declare as Time Series
  Y <- ts(data$value,start=c(1947,1),frequency = 4)

  autoplot(Y) + ggtitle("Time Plot: RGDP") +
    ylab("Billions of 2012 Dollars")
```



Note that the series appears to have exponential growth, since it is bowed upwards. Let's transform our data by taking the natural log so that it has a trend that is closer to linear.

```r
# Transform Data to have Linear Growth
y <- log(Y)
autoplot(y) + ggtitle("Time Plot: Natural Log of RGDP") +
  ylab("Natural Log of Billions of 2012 Dollars")
```



Next, let's test this transformed data for a unit root.[2]

```r
# Load test package
library(urca)
# Perform ADF test on logged data
summary(ur.df(y,type="trend",selectlags="AIC"))
```

```
##
## ###############################################
## # Augmented Dickey-Fuller Test Unit Root Test #
## ###############################################
##
## Test regression trend
```

[2]Note that because we have a trend, we would probably want to set "I=1" regardless of whether or not we have a unit root. A series with a unit root does not necessarily need to have a trend, and a series with a trend does not necessarily need to have a unit root.

```
##
##
## Call:
## lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.029965 -0.004617  0.000763  0.005313  0.032637
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.2410926  0.1081786   2.229   0.0268 *
## z.lag.1     -0.0305417  0.0141138  -2.164   0.0314 *
## tt           0.0002431  0.0001187   2.048   0.0416 *
## z.diff.lag   0.3596380  0.0612827   5.869 1.44e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.009139 on 242 degrees of freedom
## Multiple R-squared:  0.1454,Adjusted R-squared:  0.1348
## F-statistic: 13.72 on 3 and 242 DF,  p-value: 2.701e-08
##
##
## Value of test-statistic is: -2.164 18.241 3.6054
##
## Critical values for test statistics:
##       1pct  5pct 10pct
## tau3 -3.99 -3.43 -3.13
## phi2  6.22  4.75  4.07
## phi3  8.43  6.49  5.47
```

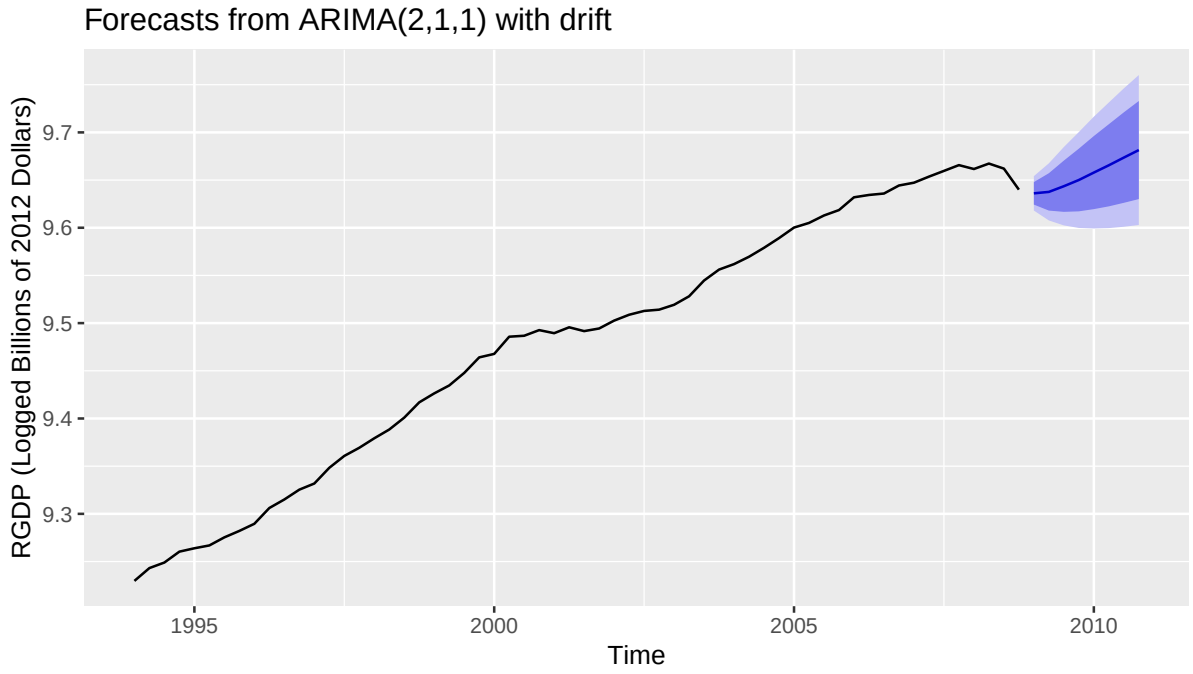
The output is a bit messy to read. The key is to compare the first test statistic (-2.164)

to the 5% critical value (−3.43). Since the test statistic is closer to zero than the critical value, we fail to reject the null hypothesis of a unit root. In other words, we should proceed as if the data has a unit root. Therefore, we need to either take the first difference of the data before using it in an ARMA model or we need to set "I=1" in the ARIMA model (which will have R take the first difference for us, behind the scenes). Let's follow the second approach, setting I=1, and fit an ARIMA(2,1,1) model to the data.

```
# Fit ARIMA(2,1,1) to logged data
fit_ARIMA <- Arima(y,order=c(2,1,1),include.constant=TRUE)
# Forecast with ARIMA model
fcst_ARIMA <- forecast(fit_ARIMA,h=8)
```

Let's plot the forecasts. I will only include the past 60 quarters of data in the plot so it is easier to see the forecasts.

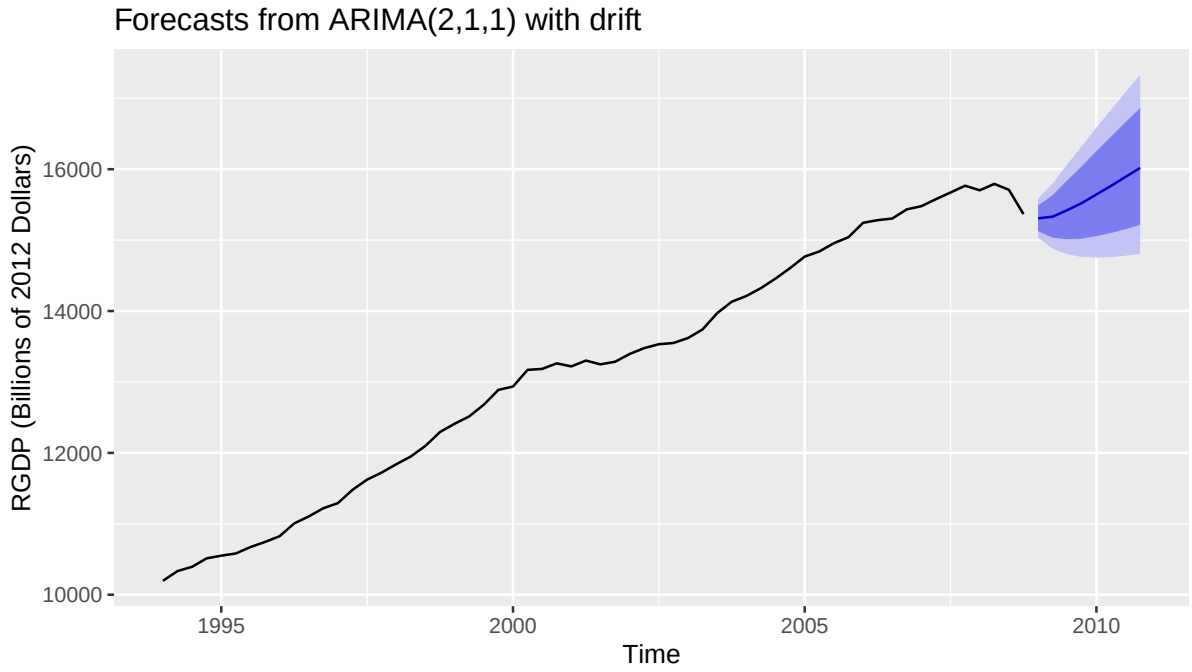
```
# Plot Forecasts
autoplot(fcst_ARIMA,include=60) + ylab("RGDP (Logged Billions of 2012 Dollars)")
```



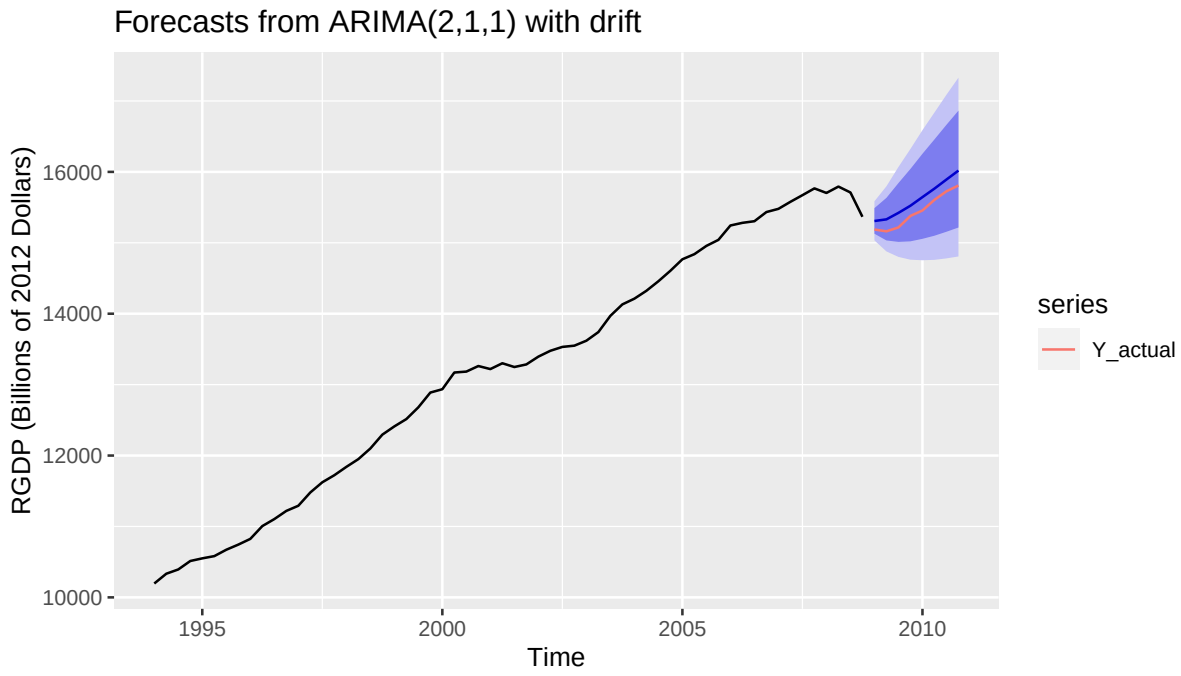
These forecasts aren't super helpful since they are in logged dollars — we probably want our forecasts in the original units, dollars. There is more than one way to do this. I am going to tell R to take the log in the ARIMA function. Then, it will return the forecasts in the original

units. Again, this is only one possible way to do this.

```
# Fit ARIMA(2,1,1) to logged data (setting lambda=0 takes log)
fit_ARIMA <- Arima(Y,order=c(2,1,1),lambda=0,include.constant=TRUE)
# Forecast with ARIMA model
fcst_ARIMA <- forecast(fit_ARIMA,h=8)
# Plot forecasts
autoplot(fcst_ARIMA,include=60) + ylab("RGDP (Billions of 2012 Dollars)")
```



Since this "forecast" is for GDP in 2009-2010, we can overlay what actually ended up happening. We see that the growth rate in GDP looks like it was forecast accurately (since the slope of the forecast and actual are similar), but the predicted level was a bit too high.



Finally we can look at a summary for the model and forecasts, as well as the measured accuracy (since we have the values that actually occured over this period).

```
  # Print summary of forecasts
  summary(fcst_ARIMA)

##
## Forecast method: ARIMA(2,1,1) with drift
##
## Model Information:
## Series: Y
## ARIMA(2,1,1) with drift
## Box Cox transformation: lambda= 0
##
## Coefficients:
##           ar1     ar2     ma1   drift
##       -0.2634  0.3213  0.5904  0.0081
## s.e.   0.2744  0.0926  0.2834  0.0010
##
## sigma^2 estimated as 8.459e-05:  log likelihood=809.94
```

```
## AIC=-1609.88   AICc=-1609.63   BIC=-1592.34
##
## Error measures:
##                     ME     RMSE      MAE         MPE      MAPE      MASE
## Training set -3.325951 59.53717 41.32429 0.001333071 0.6753398 0.1725145
##                    ACF1
## Training set 0.00916404
##
## Forecasts:
##         Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 2009 Q1       15307.35 15127.98 15488.84 15033.89 15585.78
## 2009 Q2       15330.60 15033.28 15633.81 14878.23 15796.73
## 2009 Q3       15422.11 15013.07 15842.30 14800.95 16069.35
## 2009 Q4       15522.88 15020.90 16041.64 14761.78 16323.23
## 2010 Q1       15644.27 15056.24 16255.26 14753.96 16588.30
## 2010 Q2       15764.15 15099.85 16457.67 14759.59 16837.07
## 2010 Q3       15892.11 15155.98 16664.00 14780.20 17087.67
## 2010 Q4       16018.41 15215.85 16863.30 14807.40 17328.45
```

```
  # Accuracy of forecasts
  accuracy(fcst_ARIMA,Y_actual)
```

```
##                       ME      RMSE       MAE          MPE      MAPE      MASE
## Training set   -3.325951  59.53717  41.32429  0.001333071 0.6753398 0.1725145
## Test set     -170.108570 172.50783 170.10857 -1.100833423 1.1008334 0.7101441
##                     ACF1 Theil's U
## Training set  0.00916404        NA
## Test set     -0.26215781   1.67696
```

7 Seasonal ARIMA Models

If your data has seasonality, then the ARIMA methods described above are not appropriate. Instead, we need to modify them to be able to account for seasonal data — these are called

seasonal ARIMA models.

A seasonal ARIMA model can contain the "regular" AR and MA terms described above, and sometimes a first-difference may be appropriate. However, in order to capture the seasonality of the data set, we may also wish to include seasonal AR terms, seasonal MA terms, or seasonal differences. For example, if we were to include one seasonal AR term, then we would include a lag of the data from exactly a year ago. For a seasonal MA term, we would include an MA term from exactly a year ago. Finally, the seasonal difference is simply the year-over-year change in the data.

Seasonal ARIMA models are written as ARIMA(p,d,q)(P,D,Q) where the capital "P", "D", and "Q" represent the number of seasonal AR, seasonal differences, and seasonal MA terms. Suppose we have monthly data. Then consider an ARIMA(0,0,0)(1,0,0) with intercept. This model has no regular AR or MA terms, no first difference, one seasonal AR term, no seasonal differences, and no seasonal MA terms. We would write this model as, ARIMA(0,0,0)(1,0,0)

$$y_t = b_0 + \phi_{12} y_{t-12} + \varepsilon_t$$

where we have included y_{t-12} since that data was from the same month exactly one year ago (since we are using monthly data).

Consider a few more examples. ARIMA(0,0,0)(2,0,1)

$$y_t = b_0 + \phi_{12} y_{t-12} + \phi_{24} y_{t-24} + \theta_{12} \varepsilon_{t-12} + \varepsilon_t$$

ARIMA(1,1,1)(1,0,0)

$$\Delta y_t = b_0 + \phi_1 \Delta y_{t-1} + \theta_1 \varepsilon_{t-1} + \phi_{12} \Delta y_{t-12} + \varepsilon_t$$

ARIMA(1,0,1)(1,1,0)

$$\Delta_{12} y_t = b_0 + \phi_1 \Delta_{12} y_{t-1} + \theta_1 \varepsilon_{t-1} + \phi_{12} \Delta_{12} y_{t-12} + \varepsilon_t$$

where $\Delta_{12} y_t = y_t - y_{t-12}$.

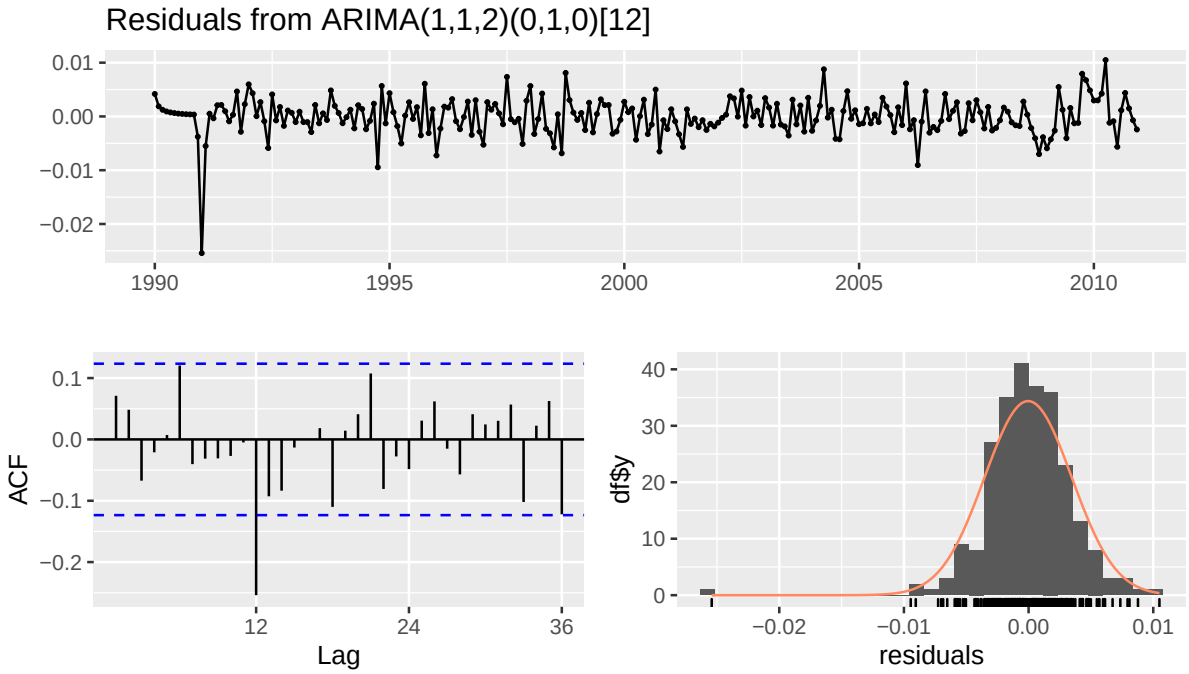
ARIMA(2,0,1)(0,0,3)

$$y_t = b_0 + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \theta_1 \varepsilon_{t-1} + \theta_{12} \varepsilon_{t-12} + \theta_{24} \varepsilon_{t-24} + \theta_{36} \varepsilon_{t-36} + \varepsilon_t$$

We will fit a seasonal ARIMA model to Minneapolis payroll employment data. We will restrict our sample to 1990-2010, and assume we wanted to forecast for 2011. Like with RGDP data, we could use the entire available history of payroll employment data, but then we would not be able to compare our "forecasts" to what actually happened.

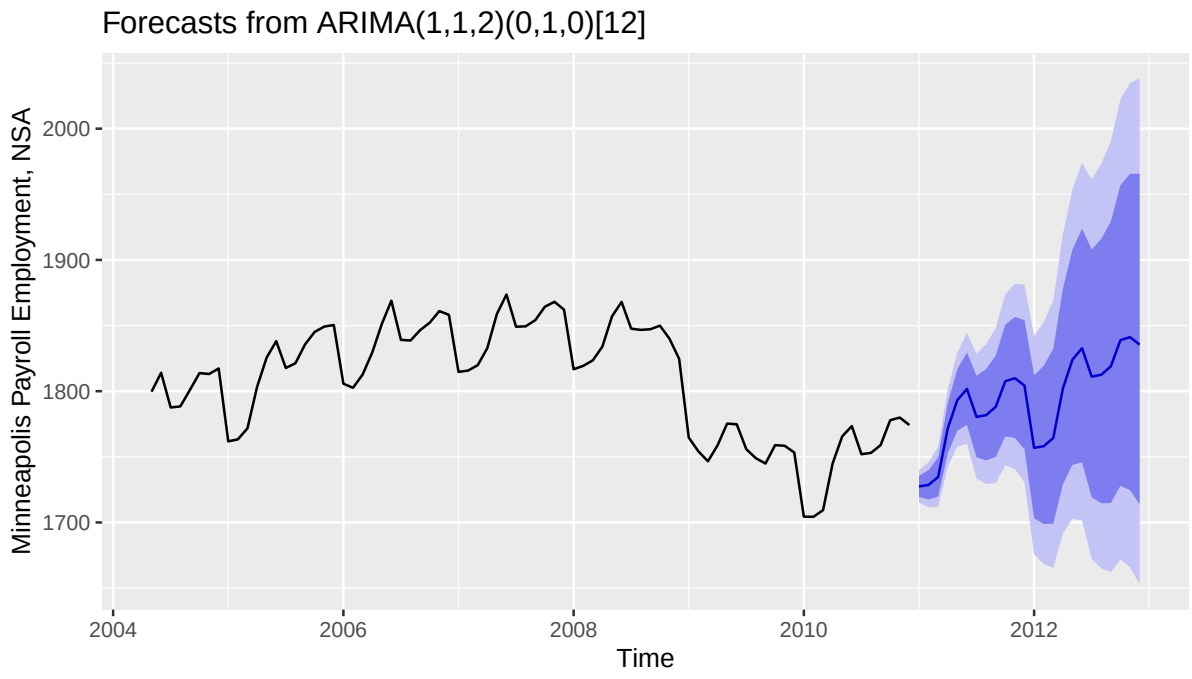
```
# Read in Data
data <- fredr(
series_id = "MINN427NAN",
observation_start = as.Date("1990-01-01"),
observation_end = as.Date("2010-12-01")
)
# Declare as Time Series
Y <- ts(data$value,start=c(1990,1),frequency = 12)


# Time Plot
autoplot(Y) +
  ggtitle("Minneapolis Payroll Employment, Not Seasonally Adjusted") +
  ylab("Percent")
```


```
# Fit seasonal ARIMA model to data
fit_ARIMA <- Arima(Y,order=c(1,1,2),seasonal=c(0,1,0),lambda=0,include.constant = FALSE)
# Check residuals - any autocorrelation left?
checkresiduals(fit_ARIMA)
```



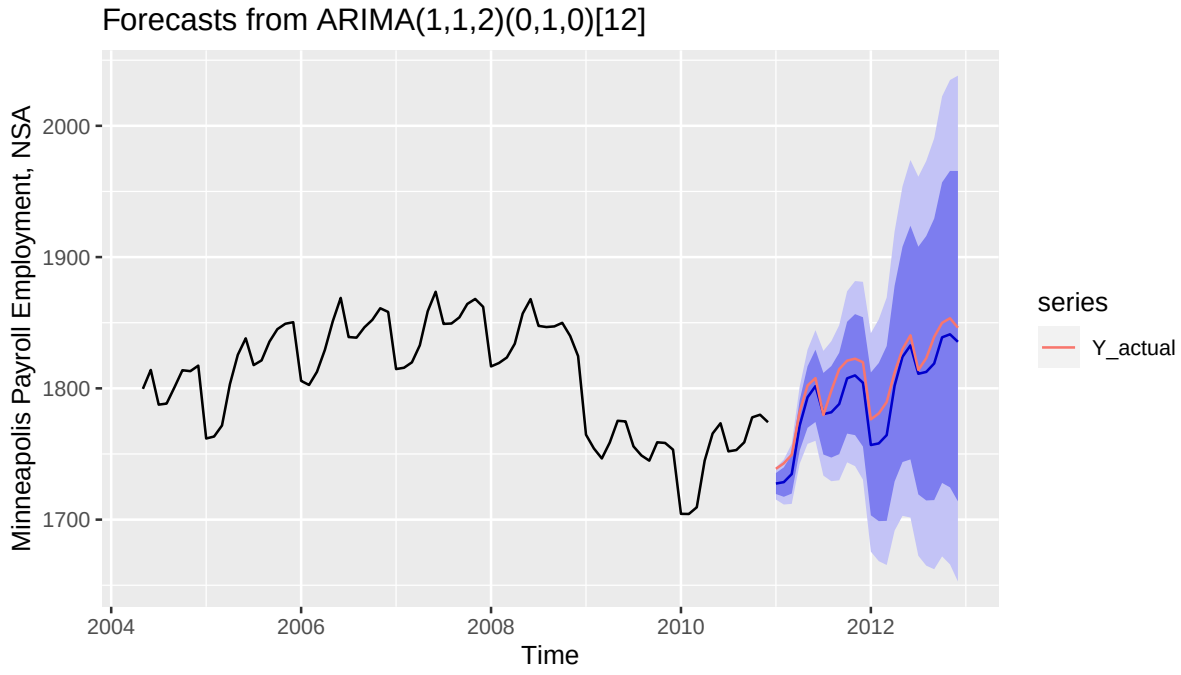
```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,2)(0,1,0)[12]
## Q* = 39.333, df = 21, p-value = 0.008953
##
## Model df: 3.   Total lags used: 24
```

```r
  # Forecast with ARIMA model
  fcst_ARIMA <- forecast(fit_ARIMA,h=24)
  # Plot forecasts
  autoplot(fcst_ARIMA,include=80) + ylab("Minneapolis Payroll Employment, NSA")
```




```r
  # Read in Data
  data <- fredr(
  series_id = "MINN427NAN",
  observation_start = as.Date("2011-01-01"),
  observation_end = as.Date("2012-12-01")
  )
```

```r
# Declare as Time Series
Y_actual <- ts(data$value,start=c(2011,1),frequency = 12)


# Plot forecasts
autoplot(fcst_ARIMA,include=80) +
  autolayer(Y_actual) +
  ylab("Minneapolis Payroll Employment, NSA")
```




```r
# Print summary of forecasts
summary(fcst_ARIMA)
```

```
##
## Forecast method: ARIMA(1,1,2)(0,1,0)[12]
##
## Model Information:
## Series: Y
## ARIMA(1,1,2)(0,1,0)[12]
## Box Cox transformation: lambda= 0
##
## Coefficients:
```

```
##          ar1      ma1     ma2
##       0.8170  -0.8348  0.2556
## s.e.  0.0684   0.0929  0.0573
##
## sigma^2 estimated as 1.304e-05:  log likelihood=1036.39
## AIC=-2064.78   AICc=-2064.6   BIC=-2050.87
##
## Error measures:
##                       ME     RMSE     MAE          MPE     MAPE      MASE
## Training set -0.07658986 5.758582 4.25303 -0.004156033 0.252282 0.1378299
##                     ACF1
## Training set 0.05991457
##
## Forecasts:
##          Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## Jan 2011       1727.476 1719.500 1735.489 1715.293 1739.746
## Feb 2011       1728.558 1717.382 1739.807 1711.496 1745.792
## Mar 2011       1734.702 1719.831 1749.700 1712.011 1757.693
## Apr 2011       1771.638 1752.554 1790.931 1742.534 1801.228
## May 2011       1793.222 1769.937 1816.814 1757.733 1829.428
## Jun 2011       1801.694 1774.365 1829.444 1760.066 1844.307
## Jul 2011       1780.397 1749.596 1811.739 1733.507 1828.554
## Aug 2011       1781.776 1747.268 1816.964 1729.273 1835.873
## Sep 2011       1788.070 1749.870 1827.104 1729.980 1848.111
## Oct 2011       1807.631 1765.533 1850.732 1743.647 1873.963
## Nov 2011       1809.865 1764.361 1856.544 1740.737 1881.739
## Dec 2011       1804.233 1755.650 1854.160 1730.464 1881.147
## Jan 2012       1756.848 1703.291 1812.090 1675.604 1842.032
## Feb 2012       1758.056 1698.864 1819.309 1668.341 1852.594
## Mar 2012       1764.391 1699.039 1832.256 1665.430 1869.232
## Apr 2012       1802.033 1729.067 1878.078 1691.644 1919.624
## May 2012       1824.047 1743.874 1907.907 1702.869 1953.849
```

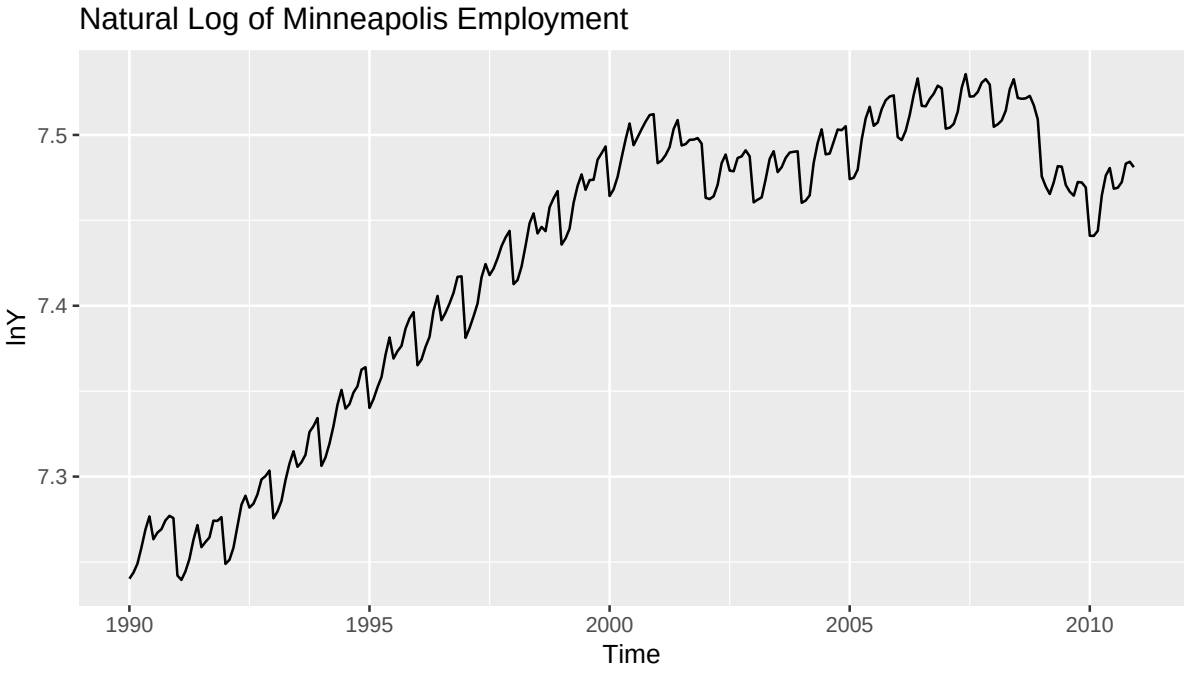
```
## Jun 2012       1832.714 1745.878 1923.869 1701.588 1973.945
## Jul 2012       1811.090 1719.187 1907.905 1672.440 1961.234
## Aug 2012       1812.525 1714.601 1916.042 1664.923 1973.213
## Sep 2012       1818.955 1714.878 1929.348 1662.216 1990.473
## Oct 2012       1838.876 1727.965 1956.905 1671.987 2022.422
## Nov 2012       1841.167 1724.594 1965.620 1665.902 2034.871
## Dec 2012       1835.452 1713.907 1965.617 1652.857 2038.218
```

The equation of the estimated model is:

$$\Delta\Delta_{12}y_t = 0.817\Delta\Delta_{12}y_{t-1} - 0.8348\varepsilon_{t-1} + 0.2556\varepsilon_{t-2} + \varepsilon_t$$

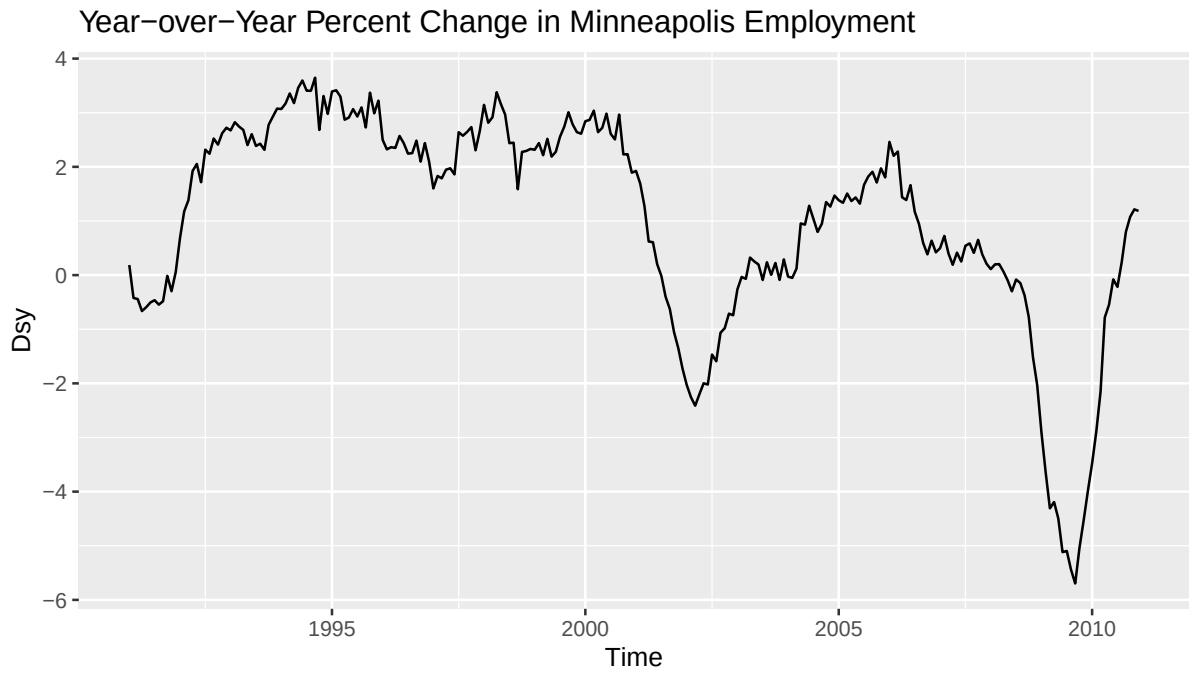
The model is fit to the change in the year-over-year change in the natural log of payroll employment in Minneapolis. This is what that transformed data looks like as it gets transformed. First, we take the natural log of the data, to remove the exponential trend:

```
lnY <- log(Y)
autoplot(lnY) + ggtitle("Natural Log of Minneapolis Employment")
```

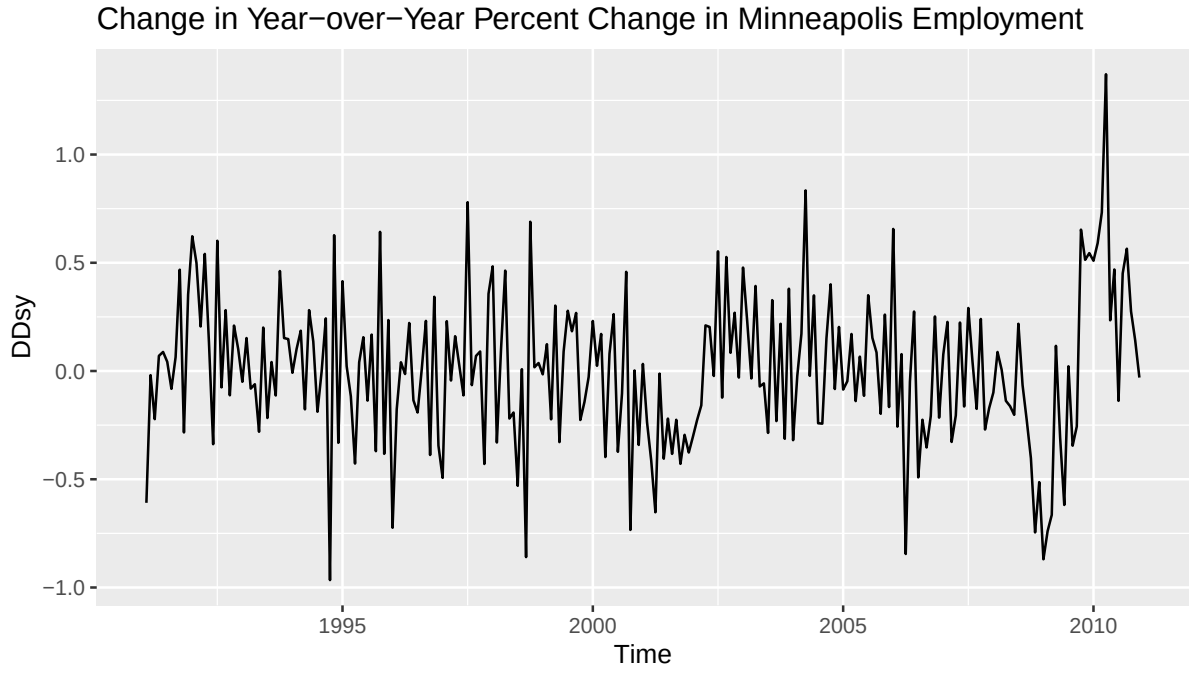


Next, we take the seasonal difference. That is, at each period, we compute the year-over-year change in the natural log of the data. This is the year-over-year percent change:

```
Dsy <- 100*diff(lnY,12)

autoplot(Dsy) + ggtitle("Year-over-Year Percent Change in Minneapolis Employment")
```

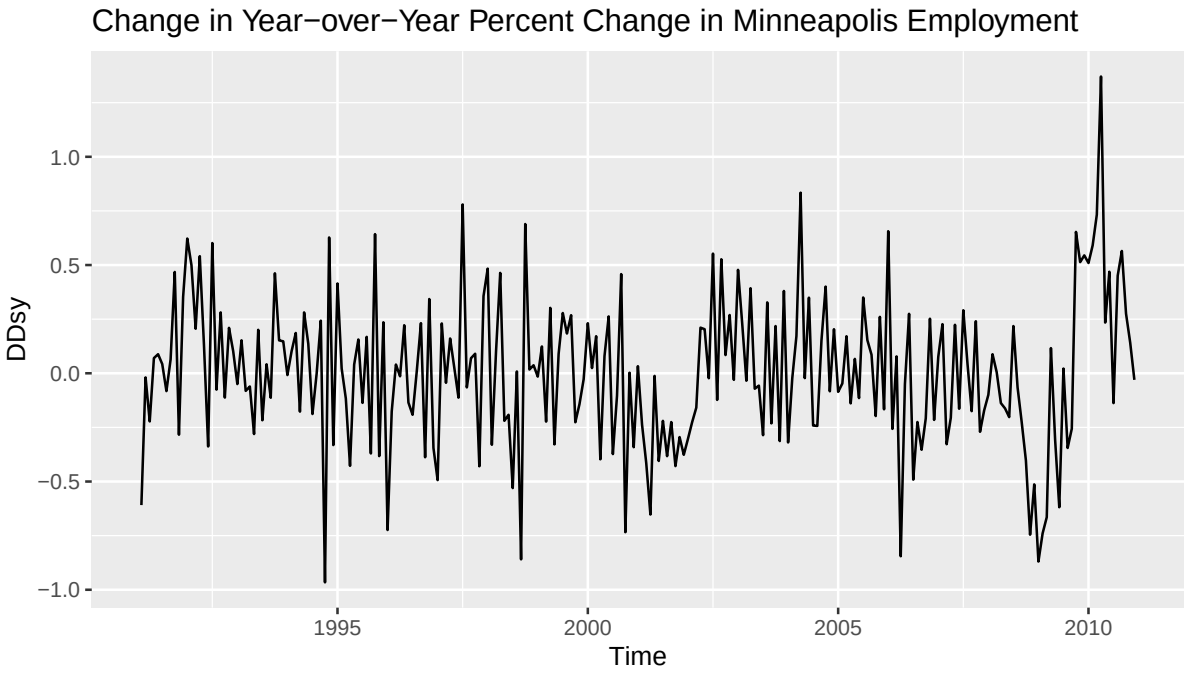



```
DDsy <- diff(Dsy)

autoplot(DDsy) +

ggtitle("Change in Year-over-Year Percent Change in Minneapolis Employment")
```



This data looks like it may have a unit root (you could test this more formally). Therefore, we remove the unit root in the year-over-year percent change by taking the first difference. This creates the change in the year-over-year percent change:

```
DDsy <- diff(Dsy)
autoplot(DDsy) +
ggtitle("Change in Year-over-Year Percent Change in Minneapolis Employment")
```



This is the fully transformed data. Once the data has been transformed like this, we fit an ARMA(1,2) to it, which produces the estimates and forecasts shown above.