Question 1 (25 points)

The following data contains the monthly number of airlines tickets sold by a travel agency for four years.

Month	Year	Tickets
January	1	605
February	1	647
March	1	636
April	1	612
May	1	714
June	1	765
July	1	698
August	1	615
September	1	588
October	1	685
November	1	711
December	1	664
January	2	630
February	2	696
March	2	670
April	2	671
May	2	724
June	2	787
July	2	724

August	2	651
September	2	589
October	2	697
November	2	750
December	2	705
January	3	664
February	3	704
March	3	691
April	3	672
May	3	753
June	3	787
July	3	751
August	3	695
September	3	643
October	3	724
November	3	803
December	3	705
January	4	720
February	4	757
March	4	707
April	4	692
May	4	828
June	4	827
July	4	763
August	4	710

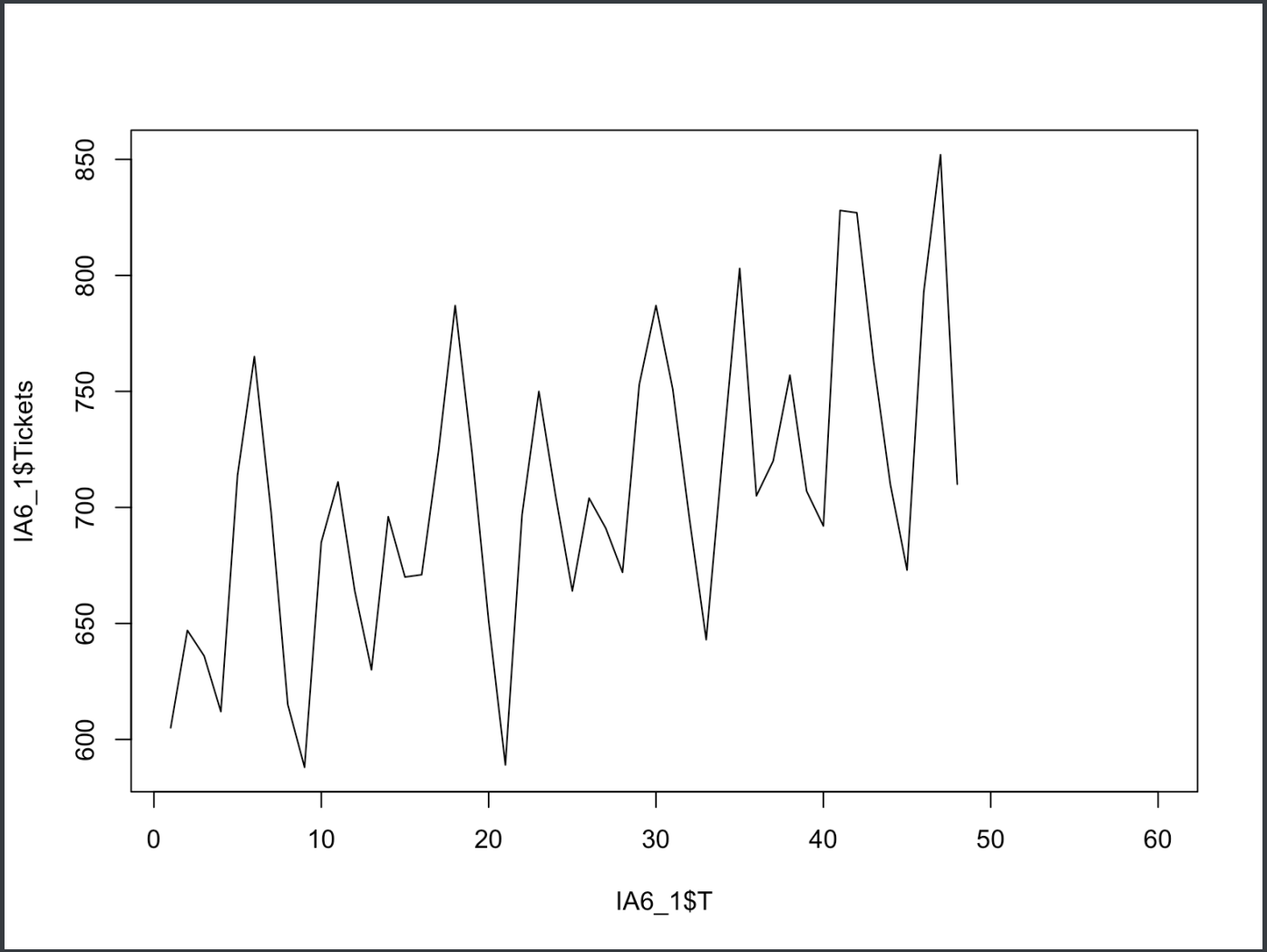
September	4	673
October	4	793
November	4	852
December	4	710

Our goal is to build a regression model to predict the demand for the following 12 months.

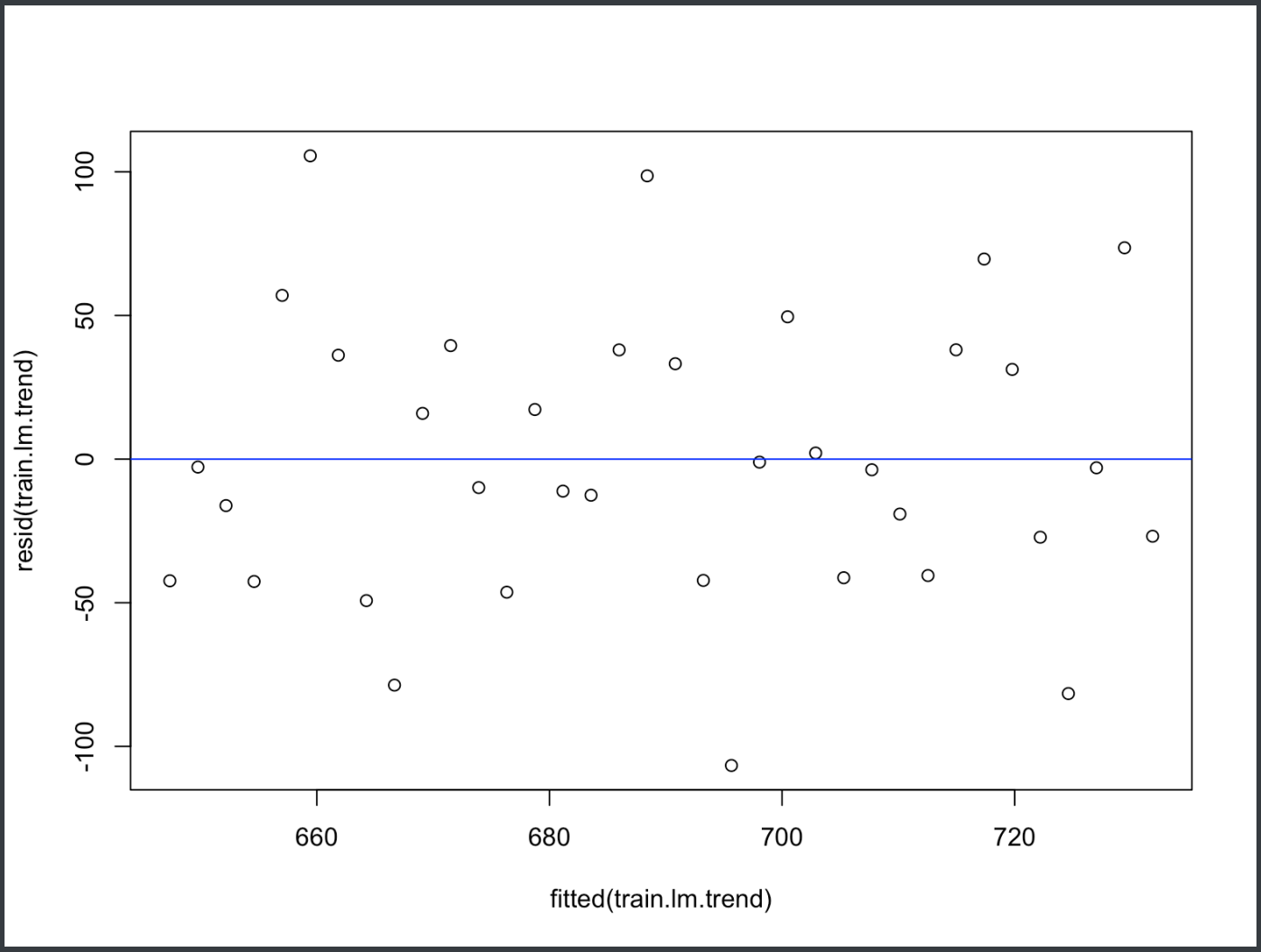
a) (2 pts) Does a linear trend appear to fit these data well? Explain why or why not. Reference any tables/figures that you need to make your point.

Solution:

Linear trend appear to fit these data well because the trend goes up approximately linear, as the time series plot shows:



Also from residual plot, we cannot see any non-linear pattern:



Thus, we say linear trend appear to fit these data well and use it for modeling.

b) (5+2+2 = 9 pts) Build a linear trend model or nonlinear trend regression model (depending on your answer in part a). Do not add a seasonality factor to this model. To validate your model, use the last 12 months as a validation data set.

1. Copy and paste your R code and display the regression output.

   Solution:

   FIrst, import the data from excel and split data for training and testing.

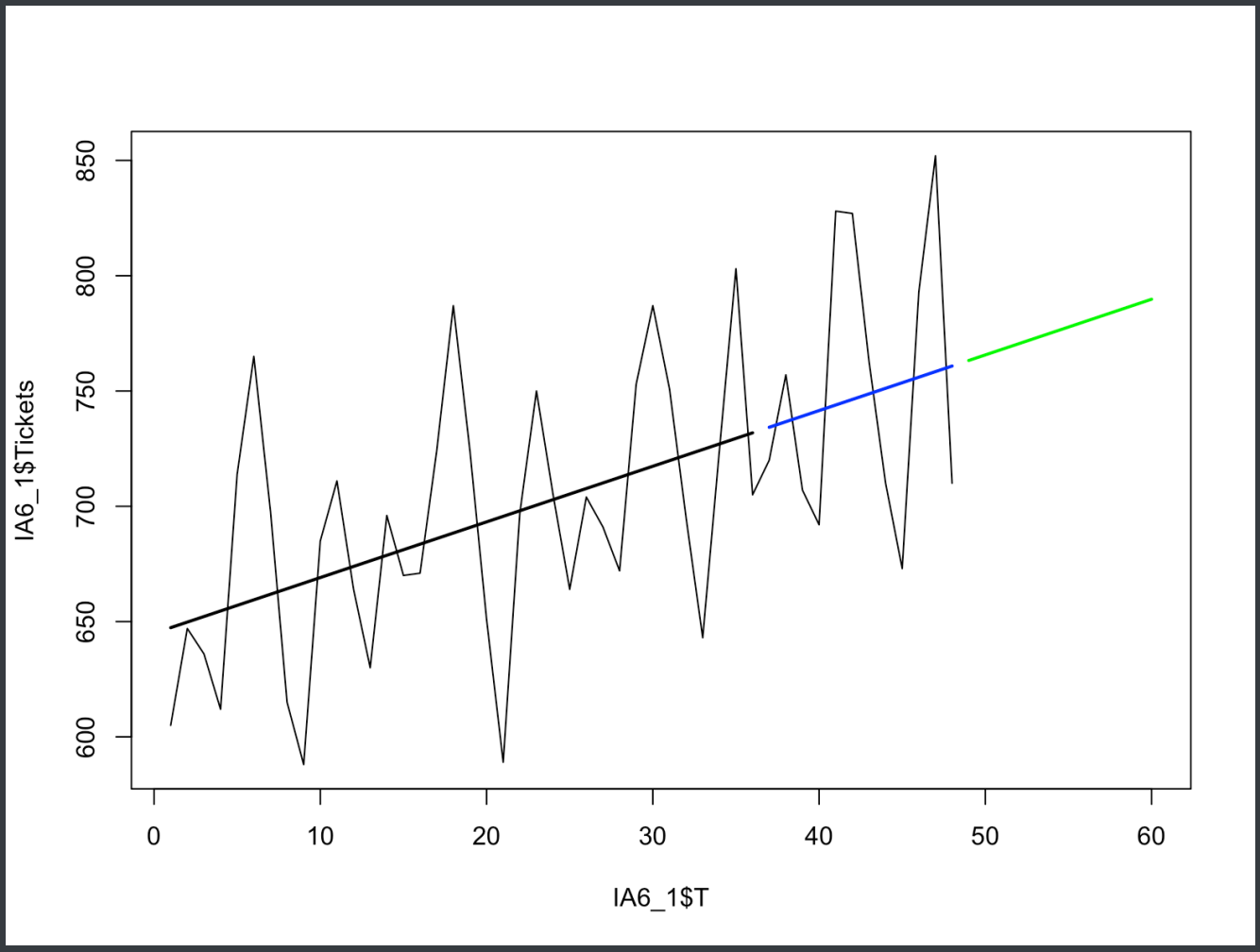
```
# attach data for question 1
attach(IA6_1)

# time series plot
plot(IA6_1$T, IA6_1$Tickets, type="l")

```

```
# split data
ndata = length(IA6_1$T)
nTrain <- ndata - 24
train <- IA6_1[1:nTrain, ]
test <-  IA6_1[nTrain+1:12, ]
# print(test)
fore <-  IA6_1[nTrain+13:24, ]
# print(fore)
```

Then, build the linear trend model as:

```
# trend model
train.lm.trend <- lm(Tickets~T, data = train)
summary(train.lm.trend)
observed <- test$Tickets
predicted <- predict(train.lm.trend, test)
# forecast for the 5 year
forecasted <- predict(train.lm.trend, fore)
print(forecasted)

# plot data and forecasts
plot(IA6_1$T, IA6_1$Tickets, type = "l")
# plot fitted value in the training period
lines(train.lm.trend$fitted, lwd=2)
lines(c(nTrain+1:12), predicted, lwd=2, col="blue")
lines(c(nTrain+13:24), forecasted, lwd=2, col="green")
```

The model goes like this:



2. What are the RMSE and MAPE of the trend model based on the validation data? Discuss the overall performance of you model.

   Solution:

   Compute the RMSE and MAPE as:

```
# compute rmse and mape
rmse.lm.trend <- rmse(observed, predicted)
mape.lm.trend <- mape(observed, predicted)*100
print(c(rmse.lm.trend,mape.lm.trend))
[1] 56.84390  6.55971
```

   RMSE=56.84390 and MAPE=6.55971%.

   Also, Adjusted R-squared: 0.1866. Considering these factors, my model's overall performance is not bad, because MAPE is within the tolerance and the model catch the trend. But it's not good enough, since it R-squared is too small, and it cannot fit in the original curve, without a seasonality factor.

3. Fill in the table with your predictions for the following 12 months.

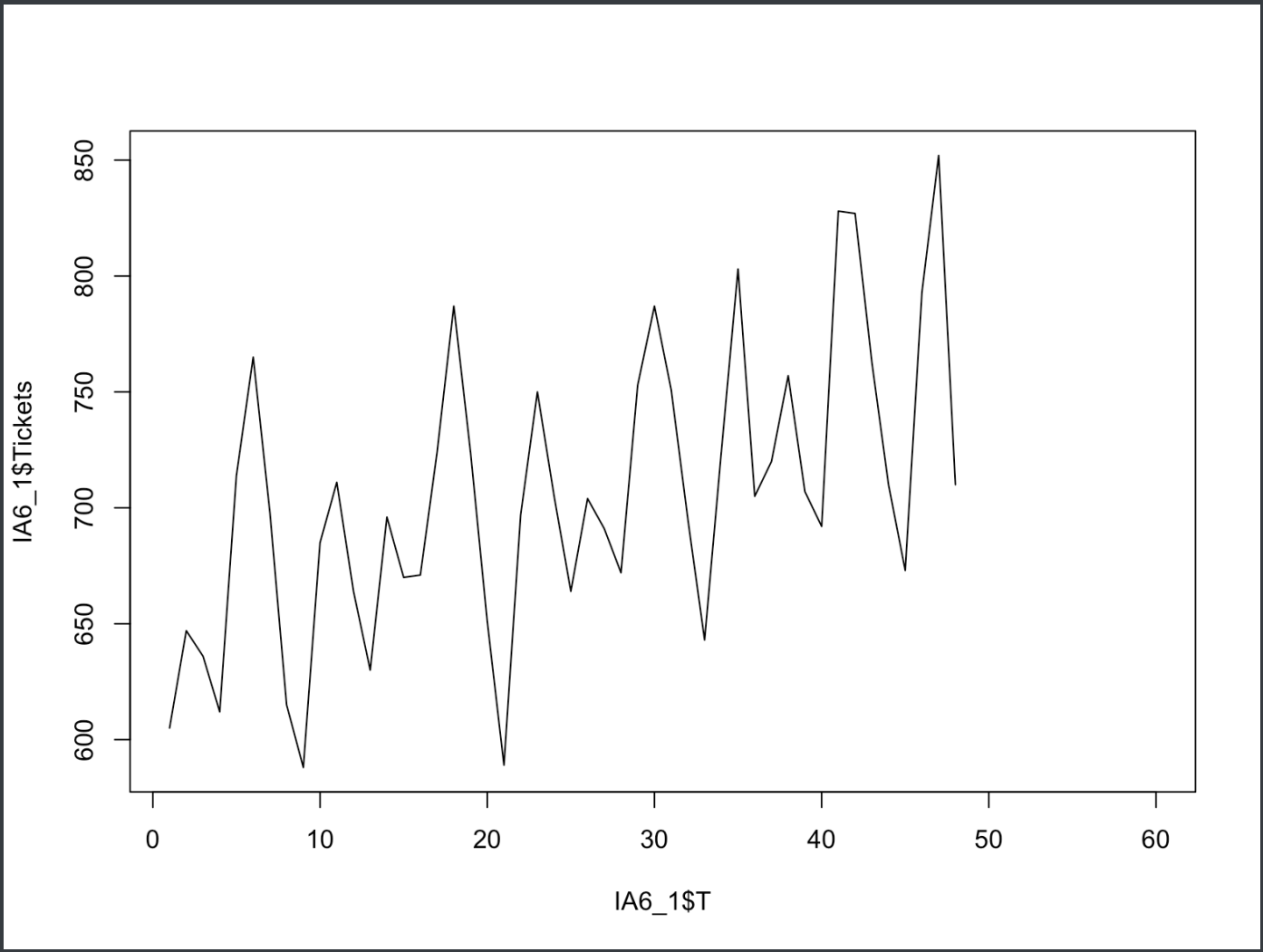
```
# forecast for the 5 year
forecasted <- predict(train.lm.trend, fore)
print(forecasted)
```

Month	Year	Tickets (Prediction)
January	5	763.2429
February	5	765.6571
March	5	768.0712
April	5	770.4854
May	5	772.8995
June	5	775.3137
July	5	777.7278
August	5	780.1420
September	5	782.5562
October	5	784.9703
November	5	787.3845
December	5	789.7986

c) (2 pts) Is there evidence of some seasonal pattern in the sales data? If so, characterize the seasonal pattern (monthly, quarterly, or yearly).

Solution:

There is strong evidence of **monthly** seasonal patterns in the sales data, as the time series plot shows:



d) (5+2+2= 9 pts) Build a regression model with trend and seasonality. To validate your model, use the last 12 months as a validation data set.

1. Copy and paste your R code and display the regression output.

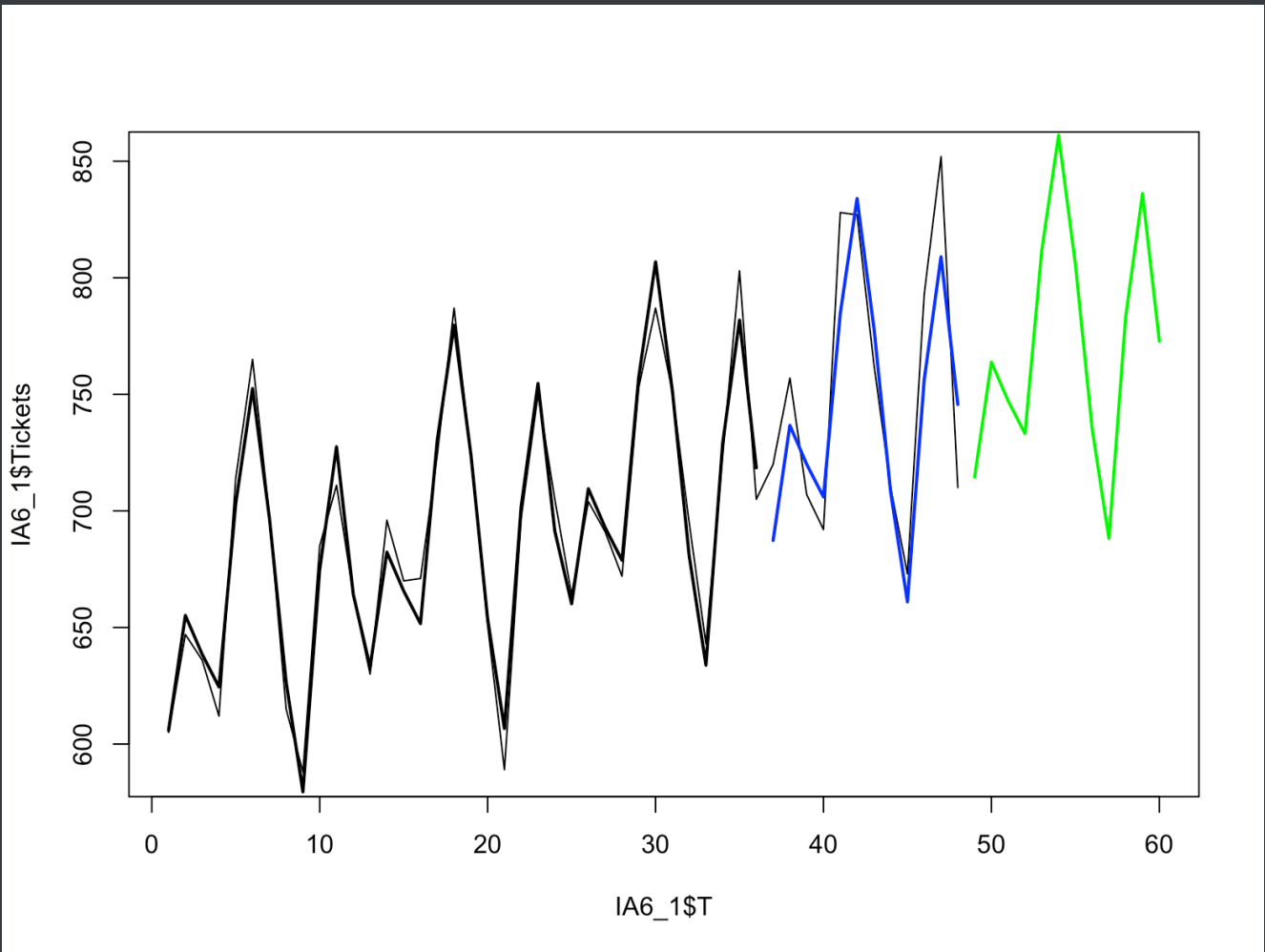
   Solution:

   Build the regression model with trend and seasonality as:

```
# trend model
train_monthly.lm.trend <- lm(Tickets~T+factor(Month), data = train)
summary(train_monthly.lm.trend)
observed <- test$Tickets
predicted <- predict(train_monthly.lm.trend, test)
# forecast for the 5 year
```

```r
forecasted_monthly <- predict(train_monthly.lm.trend, fore)
print(forecasted_monthly)

# plot data and forecasts
plot(IA6_1$T, IA6_1$Tickets, type = "l")
# plot fitted value in the training period
lines(train_monthly.lm.trend$fitted, lwd=2)
lines(c(nTrain+1:12), predicted, lwd=2, col="blue")
lines(c(nTrain+13:24), forecasted_monthly, lwd=2, col="green")
```

The model goes like:



2. What are the RMSE and MAPE of the trend model based on the validation data? Discuss the overall performance of you model.

   Solution:

   Compute the RMSE and MAPE as:

```
# compute rmse and mape
rmse_monthly.lm.trend <- rmse(observed, predicted)
mape_monthly.lm.trend <- mape(observed, predicted)*100
print(c(rmse_monthly.lm.trend, mape_monthly.lm.trend))
[1] 26.819355  2.998095
```

RMSE=26.819355 and MAPE=2.998095%.

Also, Adjusted R-squared: 0.9468. Considering these factors, my model's overall performance is very great, because MAPE is ideally small and R-squared is close to 1. The fitted values go so close to the original that even shows some over-fitted patterns, and the predict is close to the test data.

3. Fill in the table with your predictions for the following 12 months.

```
# forecast for the 5 year
forecasted_monthly <- predict(train_monthly.lm.trend, fore)
print(forecasted_monthly)
```

Month	Year	Tickets (Prediction)
January	5	714.5000
February	5	763.8333
March	5	747.1667
April	5	733.1667
May	5	811.8333
June	5	861.1667
July	5	805.8333
August	5	735.1667
September	5	688.1667
October	5	783.5000
November	5	836.1667
December	5	772.8333

e) (3 pts) Between the two models (part b and part d), which model will you use? Explain your answer.

Solution:

I choose the regression model with linear trend and seasonality in **d)**, as the graph shows clearly monthly patterns, and also the **d)** model gives a better performance, comparing the other, because of it's small MAPE, RMSE, higher R-squared, and closer prediction.

Question 2 (25 points)

The following data contains the annual revenue of a convenient store in thousand dollars.

Year	Revenue
1990	143.16
1991	156.36
1992	151.36
1993	158.56
1994	149.20
1995	171.92
1996	159.24
1997	180.60
1998	159.72
1999	194.20
2000	169.20
2001	230.80
2002	258.28
2003	228.52
2004	274.48
2005	284.16
2006	262.32
2007	313.48
2008	270.84
2009	338.04
2010	399.00
2011	395.00
2012	358.20

Our goal is to predict the revenue for the following 4 years.

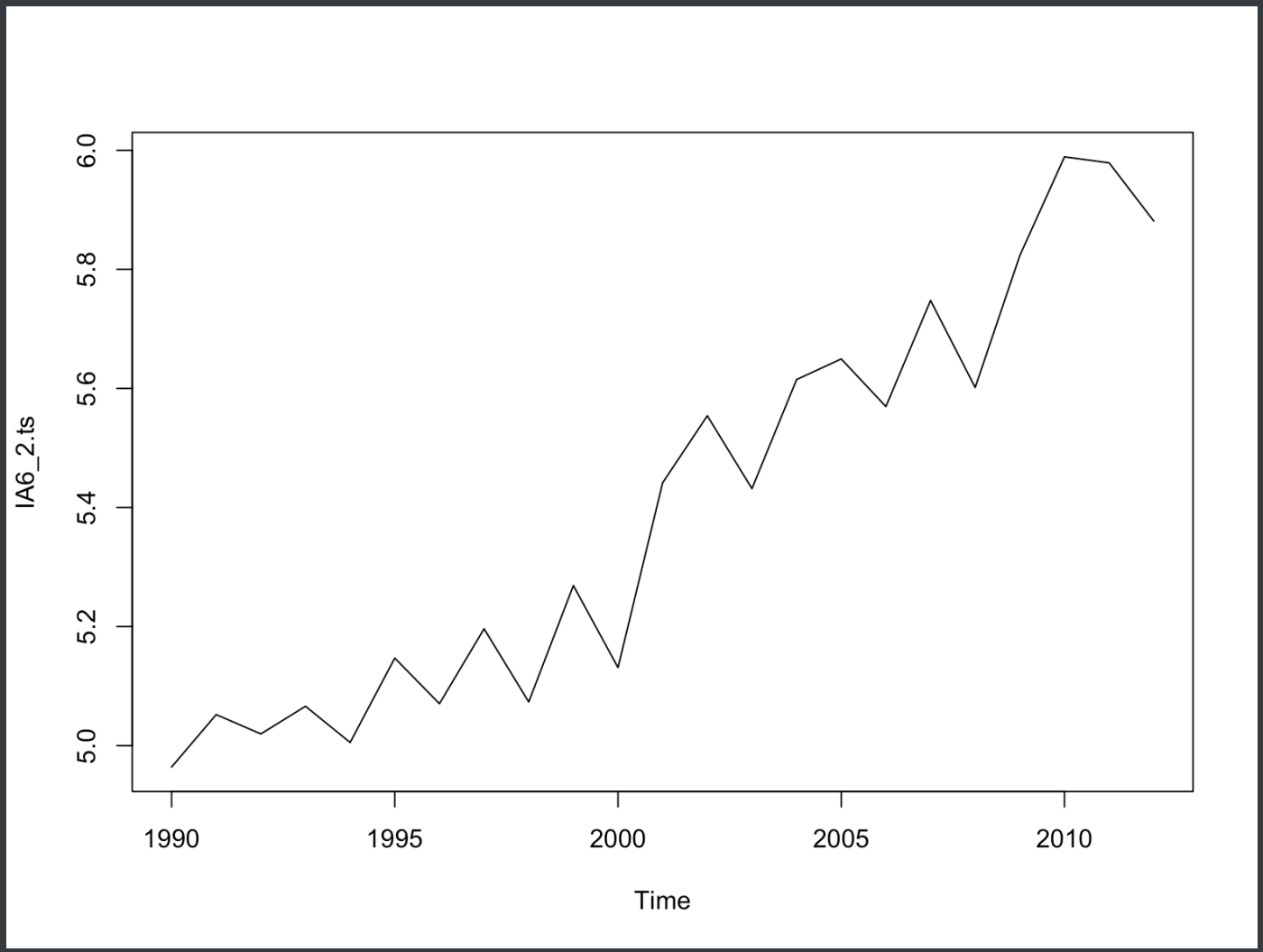
Model A:

a) (2 pts) Which exponential smoothing method would be the best (select one)?

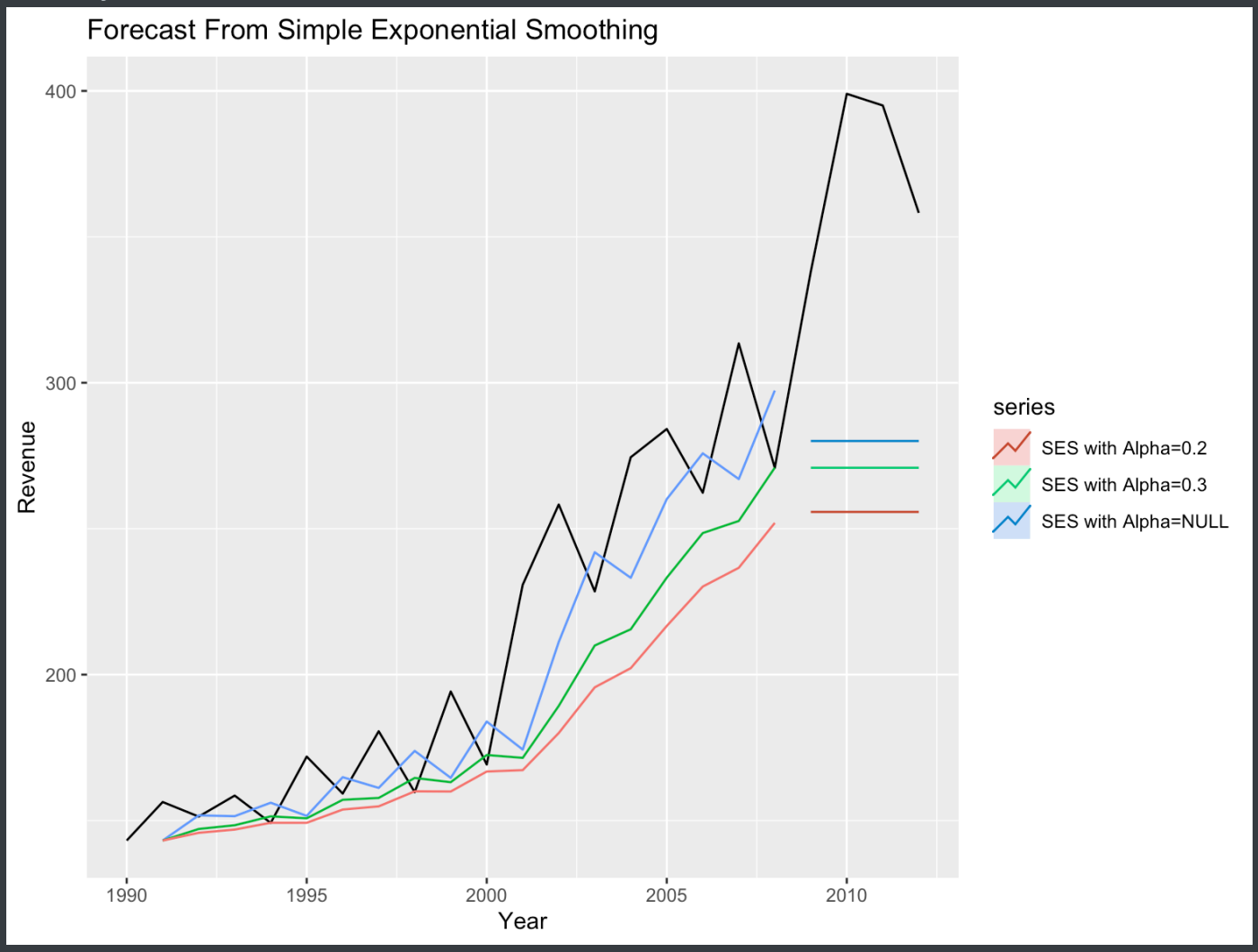
1. Simple Exponential smoothing
2. Double (Holt's) exponential smoothing
3. Triple (Holt-Winter's) exponential smoothing

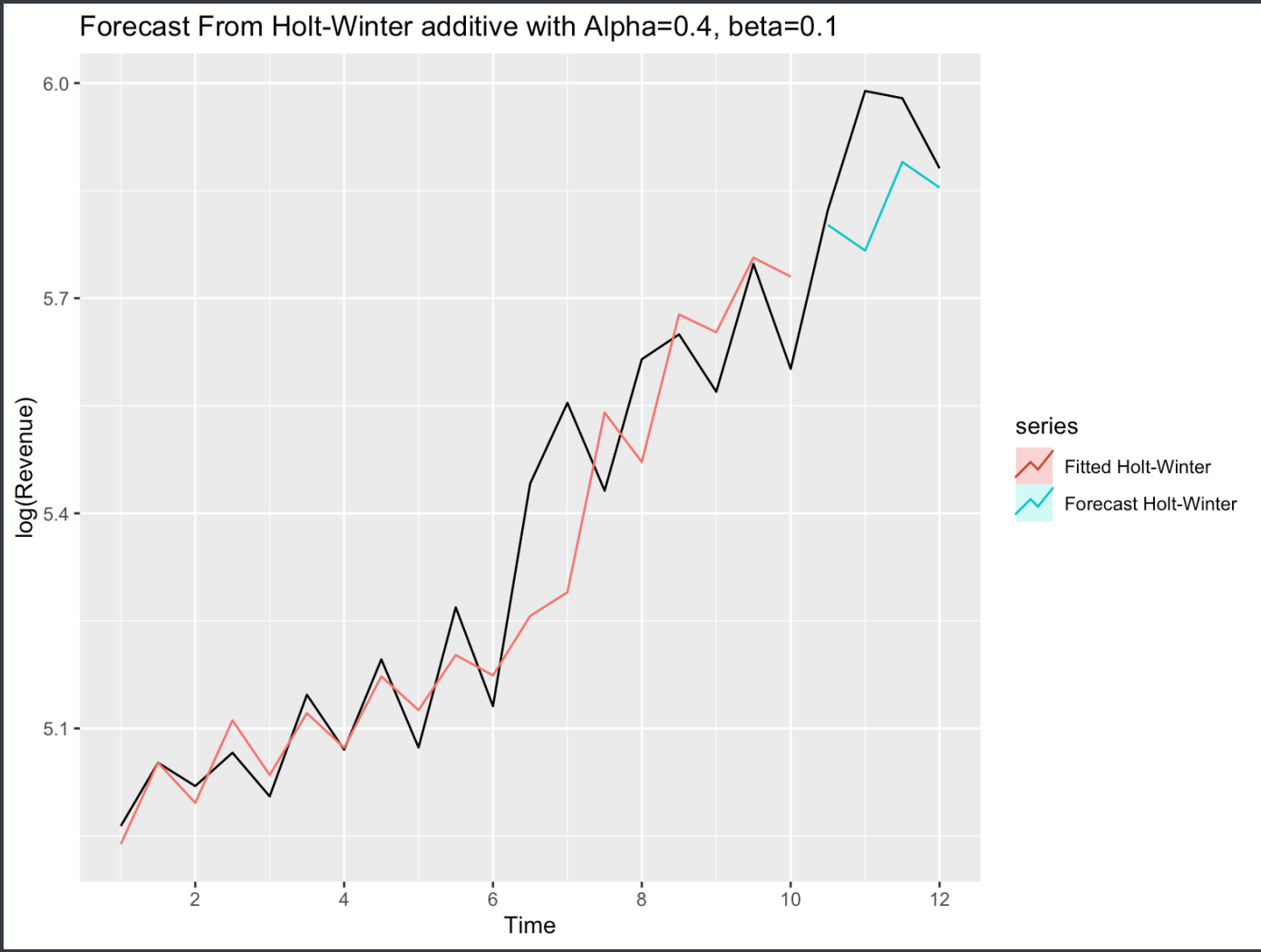
Solution:

I choose Double (Holt's) exponential smoothing, because the plot shows a trend, but the seasonality is mixed with 2 years and 3 years:



Actually, for method 1 and 3, I've also tried:





See R file for details.

b) (5+2+2=9 pts) Build an appropriate exponential smoothing model (depending on your answer in part a). To validate your model, use the last 4 years as a validation data set.

1. 1. Copy and paste your R code and display the regression output.

   Solution:

   Import data and split:

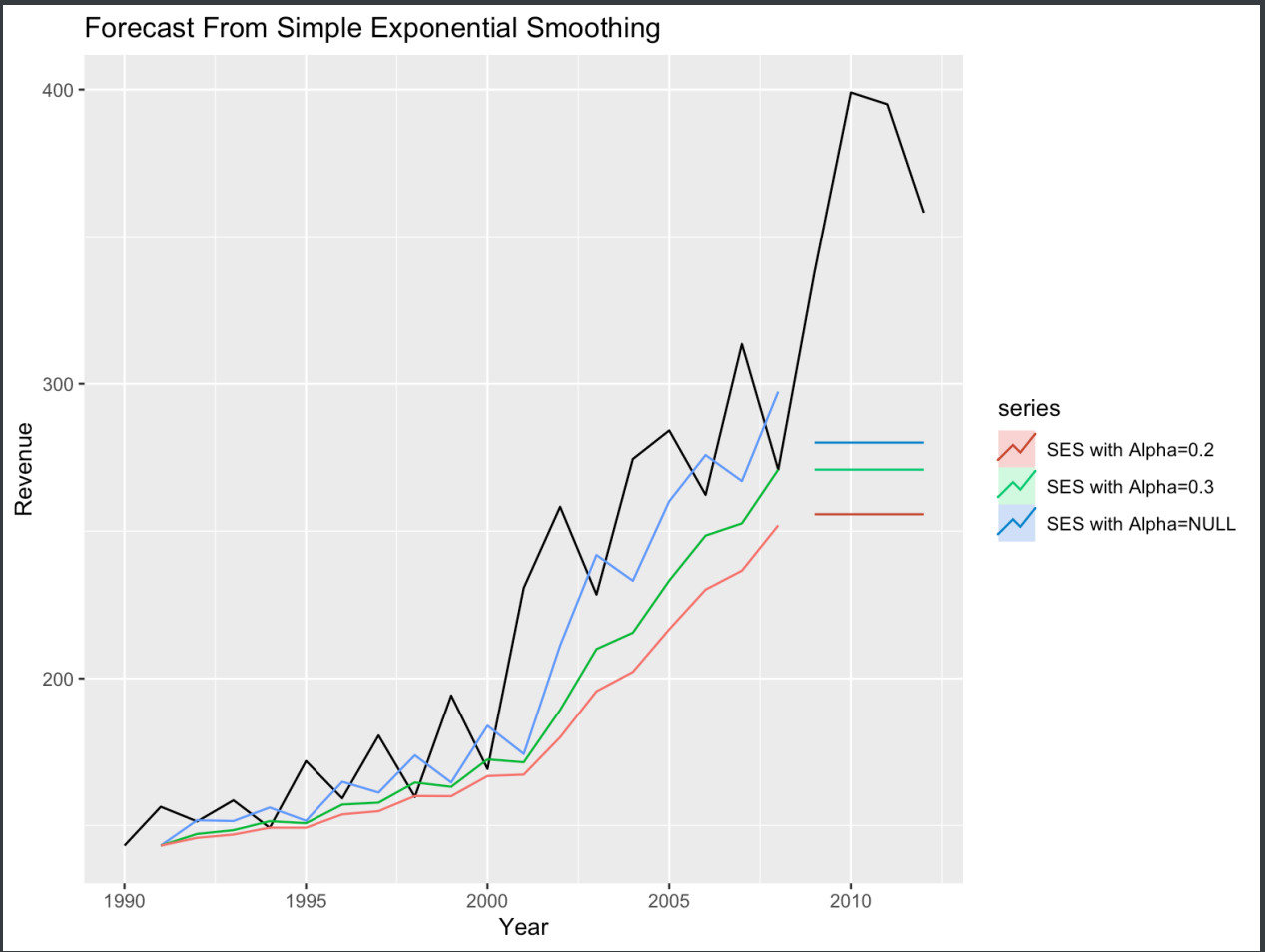
```
# attach data for question 2
attach(IA6_2)

#store the data in a time series object
IA6_2.ts <- ts(IA6_2$Revenue,start = 1990)
#IA6_2.ts <- log(IA6_2.ts) # add this for model B
#create a time series plot
```

```r
plot(IA6_2.ts)
hist(Revenue, breaks = 20)
hist(log(Revenue), breaks = 20) # add this for model B
# ndata = length(IA6_2$Revenue)
# print(ndata)

# split data
train.ts <- window(IA6_2.ts, start=1990, end=2008)
# print(train.ts)
test.ts<-window(IA6_2.ts, start=2009)
# print(test.ts)
```

Try SES to decide Alpha:

```r
# SES
# alpha=NULL
train.simple <- HoltWinters(train.ts, alpha = NULL, beta =
FALSE, gamma = FALSE)
simple.pred <- forecast(train.simple, h = 4, level = 0)
summary(simple.pred)

# alpha=0.3
train1.simple <- HoltWinters(train.ts, alpha = 0.3, beta =
FALSE, gamma = FALSE)
simple1.pred <- forecast(train1.simple, h = 4, level = 0)
summary(simple1.pred)

# alpha=0.2
train2.simple <- HoltWinters(train.ts, alpha = 0.2, beta =
FALSE, gamma = FALSE)
simple2.pred <- forecast(train2.simple, h = 4, level = 0)
summary(simple2.pred)

# autoplot to compare
autoplot(IA6_2.ts) +
  autolayer(simple.pred$fitted, series="SES with Alpha=NULL") +
```

```
  autolayer(simple.pred, series="SES with Alpha=NULL")+
  autolayer(simple1.pred$fitted, series="SES with Alpha=0.3") +
  autolayer(simple1.pred, series="SES with Alpha=0.3")+
  autolayer(simple2.pred$fitted, series="SES with Alpha=0.2") +
  autolayer(simple2.pred, series="SES with Alpha=0.2")+
  xlab("Year")+ylab("Revenue")+
  ggtitle("Forecast From Simple Exponential Smoothing")
```

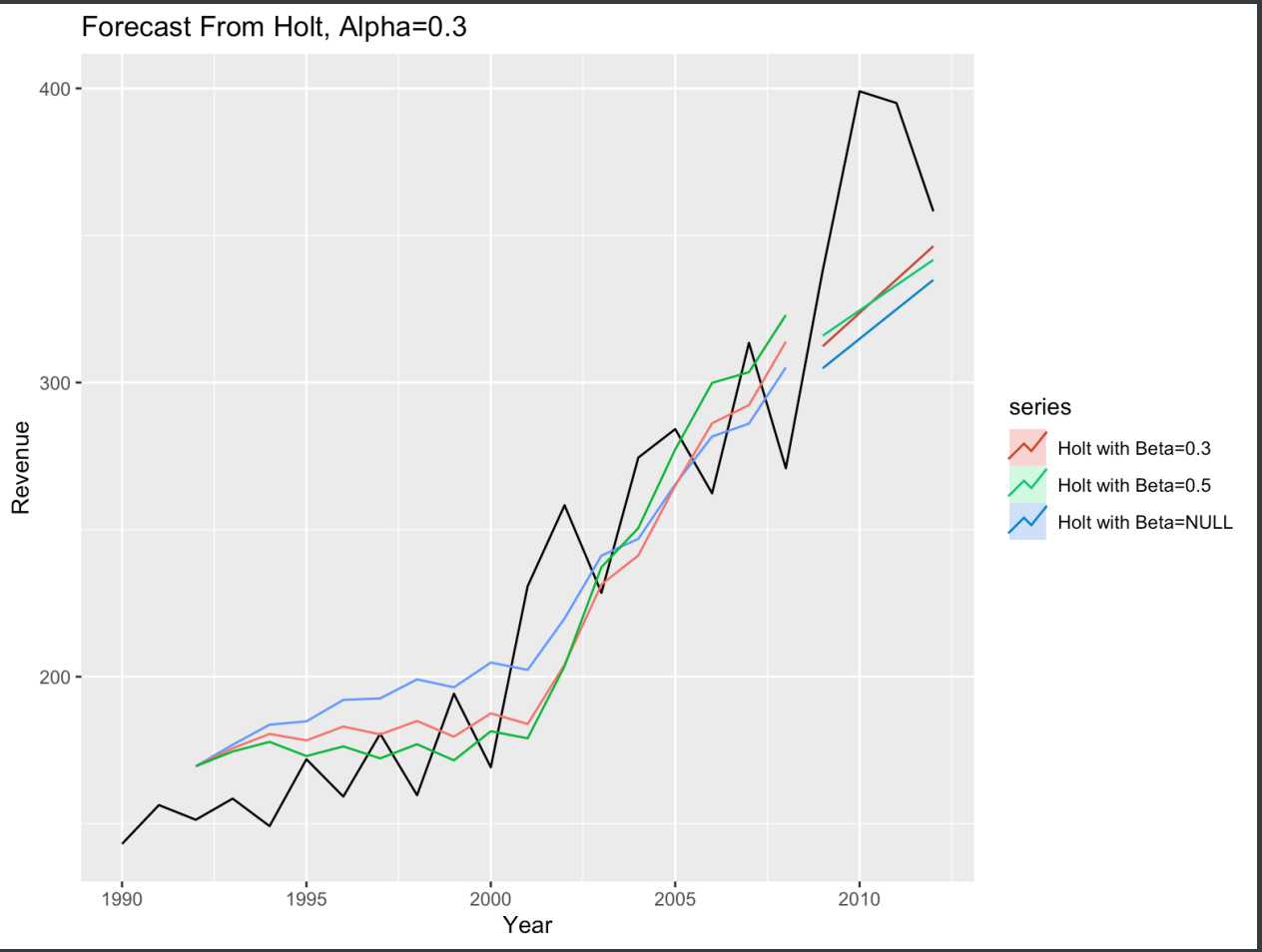


For the best performance, choose Alpha = **0.3**.

Then build Holt's as:

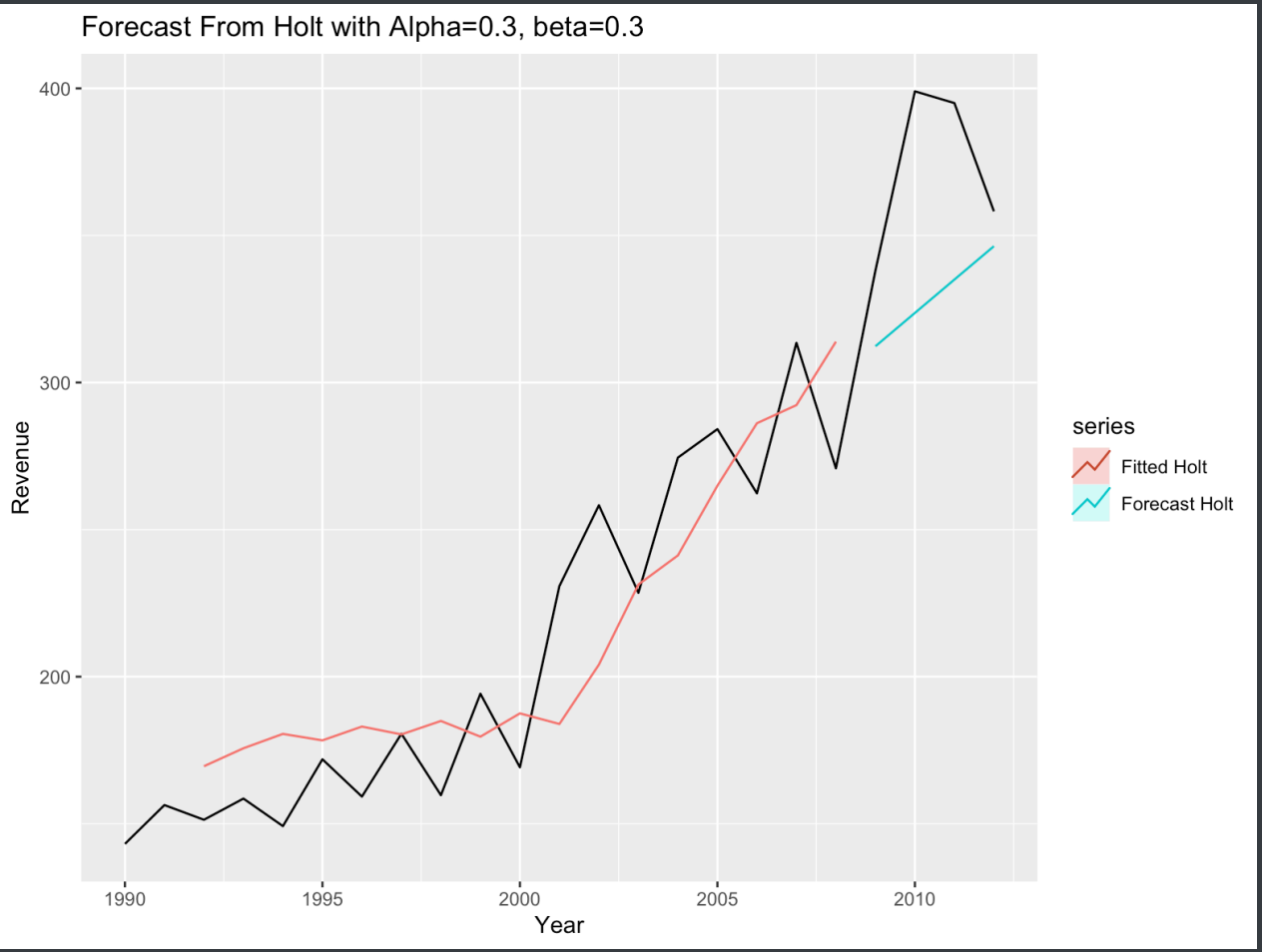
```
# Holt, alpha=0.3
# beta=NULL
train.Holt <- HoltWinters(train.ts, alpha = 0.3, beta = NULL,
gamma = FALSE)
Holt.pred <- forecast(train.Holt, h = 4, level = 0)
summary(Holt.pred)

```

```r
# beta=0.3
train1.Holt <- HoltWinters(train.ts, alpha = 0.3, beta = 0.3, gamma = FALSE)
Holt1.pred <- forecast(train1.Holt, h = 4, level = 0)
summary(Holt1.pred)

# beta=0.5
train2.Holt <- HoltWinters(train.ts, alpha = 0.3, beta = 0.5, gamma = FALSE)
Holt2.pred <- forecast(train2.Holt, h = 4, level = 0)
summary(Holt2.pred)

# autoplot to compare
autoplot(IA6_2.ts) +
  autolayer(Holt.pred$fitted, series="Holt with Beta=NULL") +
  autolayer(Holt.pred, series="Holt with Beta=NULL")+
  autolayer(Holt1.pred$fitted, series="Holt with Beta=0.3") +
  autolayer(Holt1.pred, series="Holt with Beta=0.3")+
  autolayer(Holt2.pred$fitted, series="Holt with Beta=0.5") +
  autolayer(Holt2.pred, series="Holt with Beta=0.5")+
  xlab("Year")+ylab("Revenue")+
  ggtitle("Forecast From Holt, Alpha=0.3")
```



For the best performance, choose Beta = **0.1**.

```
# choose beta=0.3 and plot
autoplot(IA6_2.ts) +
  autolayer(Holt1.pred$fitted, series="Fitted Holt") +
  autolayer(Holt1.pred, series="Forecast Holt")+
  xlab("Year")+ylab("Revenue")+
  ggtitle("Forecast From Holt with Alpha=0.3, beta=0.3")
```



2. What are the RMSE and MAPE of the trend model based on the validation data? Discuss the overall performance of you model.

   Solution:

   Compute the RMSE and MAPE as:

```
1  # compute rmse and mape
2  print(c(rmse(test.ts,Holt1.pred$mean),mape(test.ts,Holt1.pred$me
   an)))
3  [1] 50.1977504  0.1124855
```

   RMSE=50.1977504 and MAPE=0.1124855.

   My model's overall performance is quite great here, because the MAPE is relatively high, meaning that the predict isn't accurate enough, also the predict is not close to the test. This model, however, is already an optimal option without over-fitting or under-fitting, and the problem is due to the small size of data and mixed seasonality.

3. Fill in the table with your predictions for the following 4 years.

```
# forcast
train1.holtfore <- forecast(train1.Holt, h = 8, level = 0)
summary(train1.holtfore)
```

Year	Revenue
2013	357.6847
2014	369.0276
2015	380.3704
2016	391.7132

Model B:

Now, take the logarithmic transformation of the revenue (log(Revenue)).

a) (2 pts) Which exponential smoothing method would be the best (select one)?

1. Simple Exponential smoothing
2. Double (Holt's) exponential smoothing
3. Triple (Holt-Winter's) exponential smoothing

Solution:

I choose Double (Holt's) exponential smoothing, because the plot shows a trend, but the seasonality is mixed with 2 years and 3 years, **same as a)**

b) (5+2+2=9 pts) Build an appropriate exponential smoothing model (depending on your answer in part a). To validate your model, use the last 4 years as a validation data set.

1. 1. Copy and paste your R code and display the regression output.

   Solution:

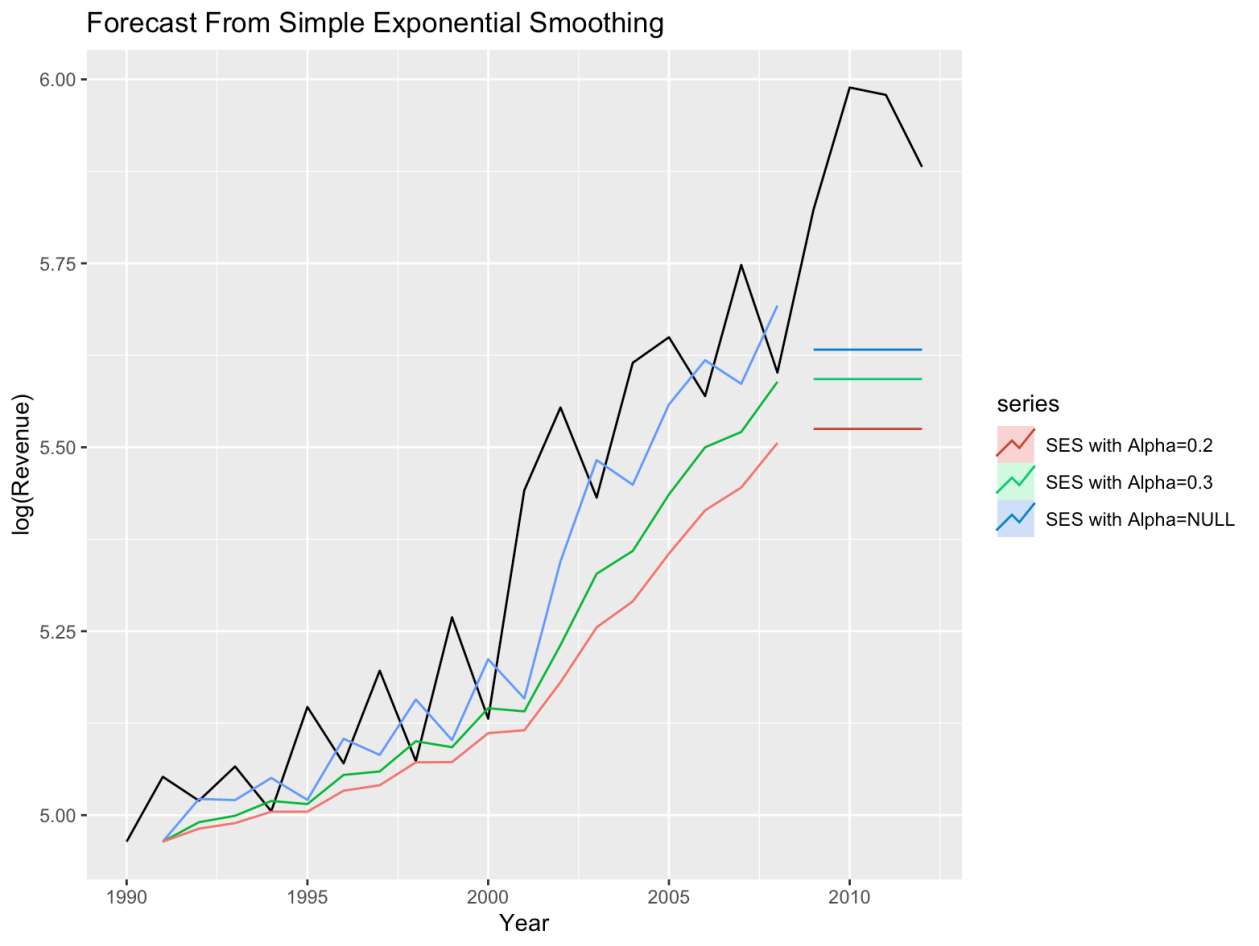
   Import data and split:

```r
# attach data for question 2
attach(IA6_2)

# store the data in a time series object
IA6_2.ts <- ts(IA6_2$Revenue,start = 1990)
IA6_2.ts <- log(IA6_2.ts) # add this for model B
# create a time series plot
plot(IA6_2.ts)
hist(Revenue, breaks = 20)
hist(log(Revenue), breaks = 20) # add this for model B
# ndata = length(IA6_2$Revenue)
# print(ndata)

# split data
train.ts <- window(IA6_2.ts, start=1990, end=2008)
# print(train.ts)
test.ts<-window(IA6_2.ts, start=2009)
# print(test.ts)
```

   Try SES to decide Alpha:

```r
# SES
# alpha=NULL
train.simple <- HoltWinters(train.ts, alpha = NULL, beta =
FALSE, gamma = FALSE)
simple.pred <- forecast(train.simple, h = 4, level = 0)
summary(simple.pred)

# alpha=0.3
train1.simple <- HoltWinters(train.ts, alpha = 0.3, beta =
FALSE, gamma = FALSE)
```

```r
simple1.pred <- forecast(train1.simple, h = 4, level = 0)
summary(simple1.pred)

# alpha=0.2
train2.simple <- HoltWinters(train.ts, alpha = 0.2, beta =
FALSE, gamma = FALSE)
simple2.pred <- forecast(train2.simple, h = 4, level = 0)
summary(simple2.pred)

# autoplot to compare
autoplot(IA6_2.ts) +
  autolayer(simple.pred$fitted, series="SES with Alpha=NULL") +
  autolayer(simple.pred, series="SES with Alpha=NULL")+
  autolayer(simple1.pred$fitted, series="SES with Alpha=0.3") +
  autolayer(simple1.pred, series="SES with Alpha=0.3")+
  autolayer(simple2.pred$fitted, series="SES with Alpha=0.2") +
  autolayer(simple2.pred, series="SES with Alpha=0.2")+
  xlab("Year")+ylab("log(Revenue)")+
  ggtitle("Forecast From Simple Exponential Smoothing")
```

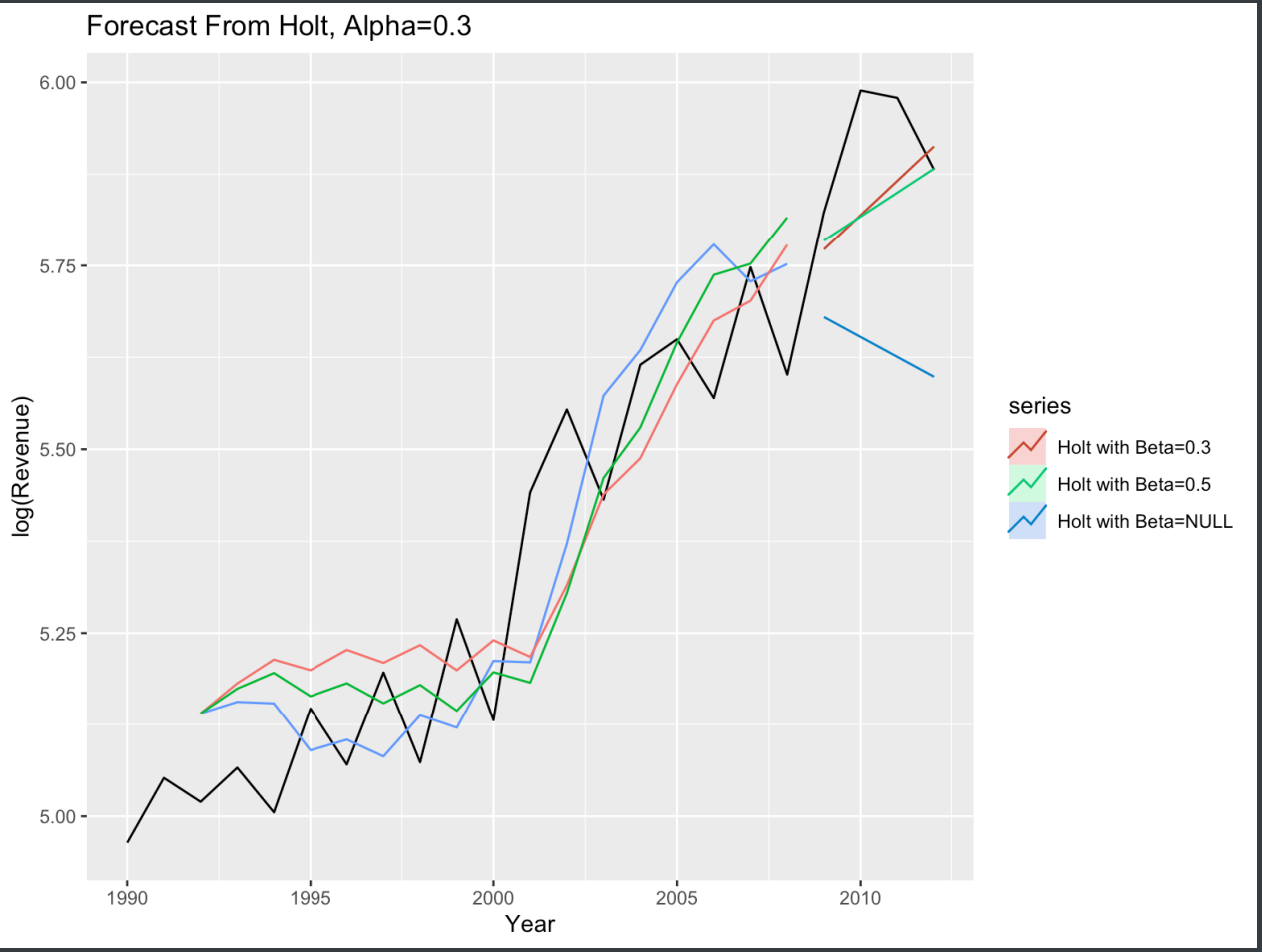


For the best performance, choose Alpha = **0.3**.

Then build Holt's as:

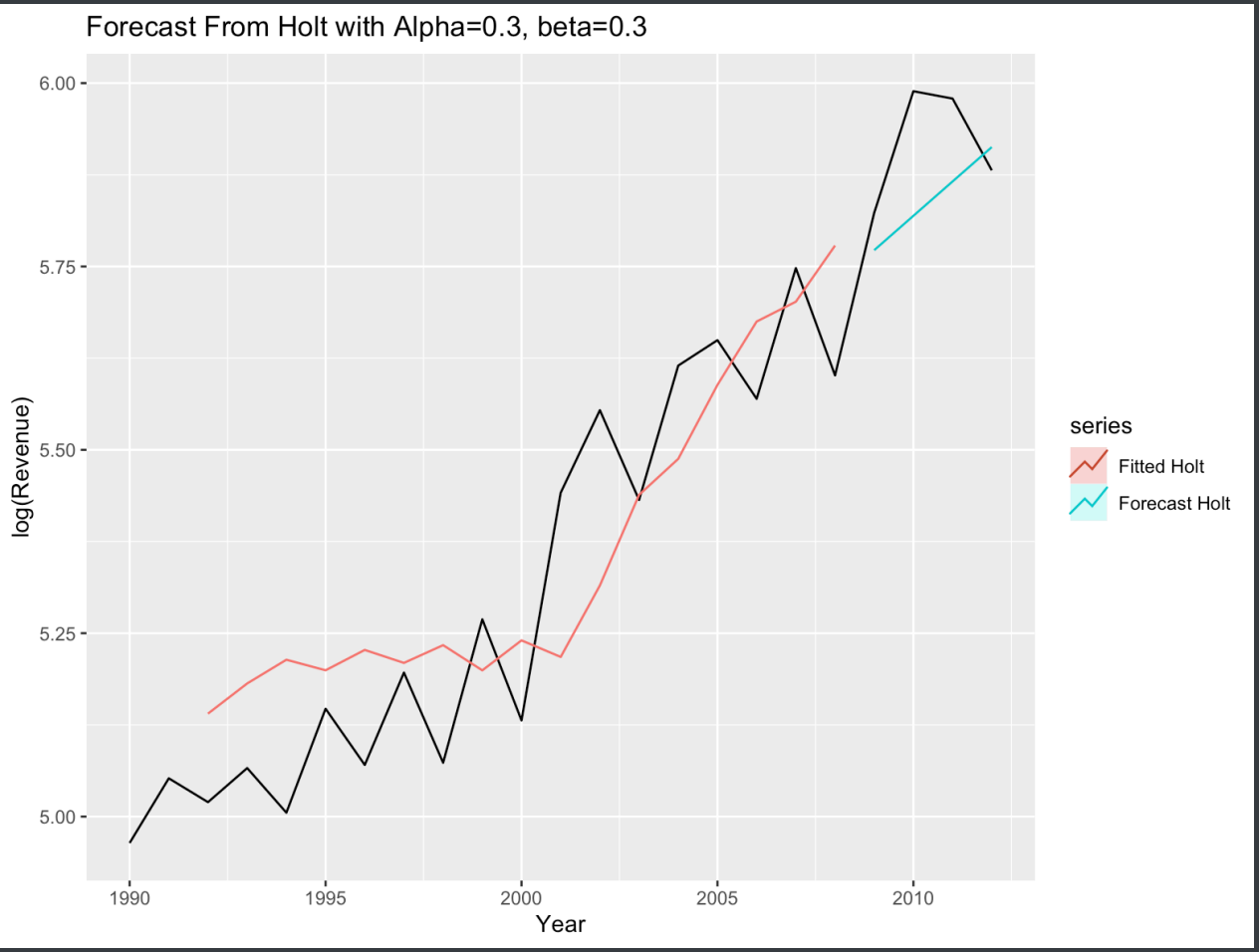
```
# Holt, alpha=0.3
# beta=NULL
train.Holt <- HoltWinters(train.ts, alpha = 0.3, beta = NULL,
gamma = FALSE)
Holt.pred <- forecast(train.Holt, h = 4, level = 0)
summary(Holt.pred)

# beta=0.3
train1.Holt <- HoltWinters(train.ts, alpha = 0.3, beta = 0.3,
gamma = FALSE)
Holt1.pred <- forecast(train1.Holt, h = 4, level = 0)
summary(Holt1.pred)

# beta=0.5
```

```r
train2.Holt <- HoltWinters(train.ts, alpha = 0.3, beta = 0.5,
gamma = FALSE)
Holt2.pred <- forecast(train2.Holt, h = 4, level = 0)
summary(Holt2.pred)

# autoplot to compare
autoplot(IA6_2.ts) +
  autolayer(Holt.pred$fitted, series="Holt with Beta=NULL") +
  autolayer(Holt.pred, series="Holt with Beta=NULL")+
  autolayer(Holt1.pred$fitted, series="Holt with Beta=0.3") +
  autolayer(Holt1.pred, series="Holt with Beta=0.3")+
  autolayer(Holt2.pred$fitted, series="Holt with Beta=0.5") +
  autolayer(Holt2.pred, series="Holt with Beta=0.5")+
  xlab("Year")+ylab("log(Revenue)")+
  ggtitle("Forecast From Holt, Alpha=0.3")
```



For the best performance, choose Beta = **0.1**.

```r
# choose beta=0.3 and plot
autoplot(IA6_2.ts) +
  autolayer(Holt1.pred$fitted, series="Fitted Holt") +
  autolayer(Holt1.pred, series="Forecast Holt")+
  xlab("Year")+ylab("log(Revenue)")+
  ggtitle("Forecast From Holt with Alpha=0.3, beta=0.3")
```



2. What are the RMSE and MAPE of the trend model based on the validation data? Discuss the overall performance of you model.

   Solution:

   Compute the RMSE and MAPE as:

```r
# compute rmse and mape
print(c(rmse(test.ts,Holt1.pred$mean),mape(test.ts,Holt1.pred$mean)))
[1] 0.10633263 0.01534862
```

RMSE=0.10633263 and MAPE=0.01534862.

Similarly to a)'s reason, my model's overall performance is still not quite great here, because predict isn't accurate enough. Here the RMSE and MAPE is quite small, because log function lower it down.

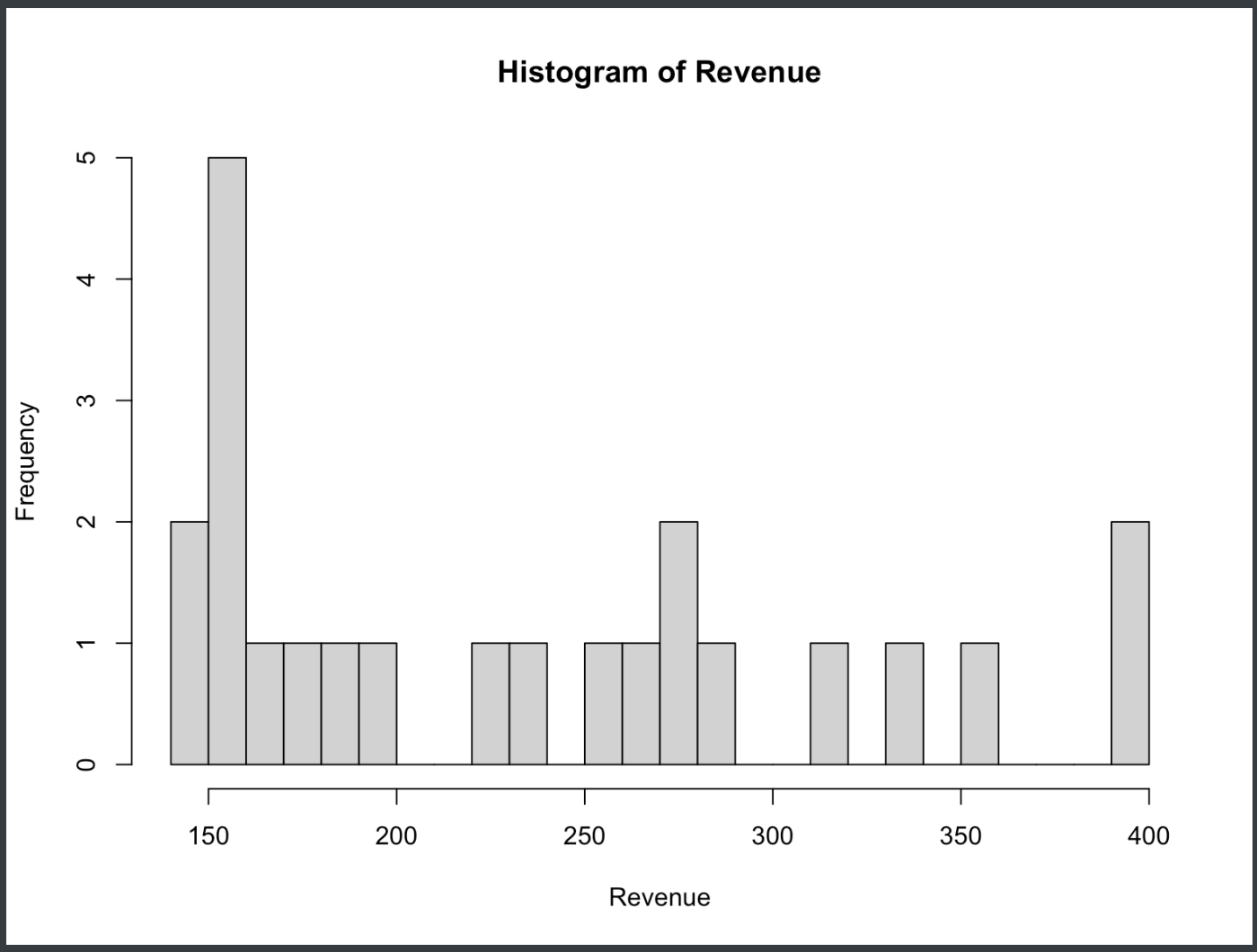
3. Fill in the table with your predictions for the following 4 years.

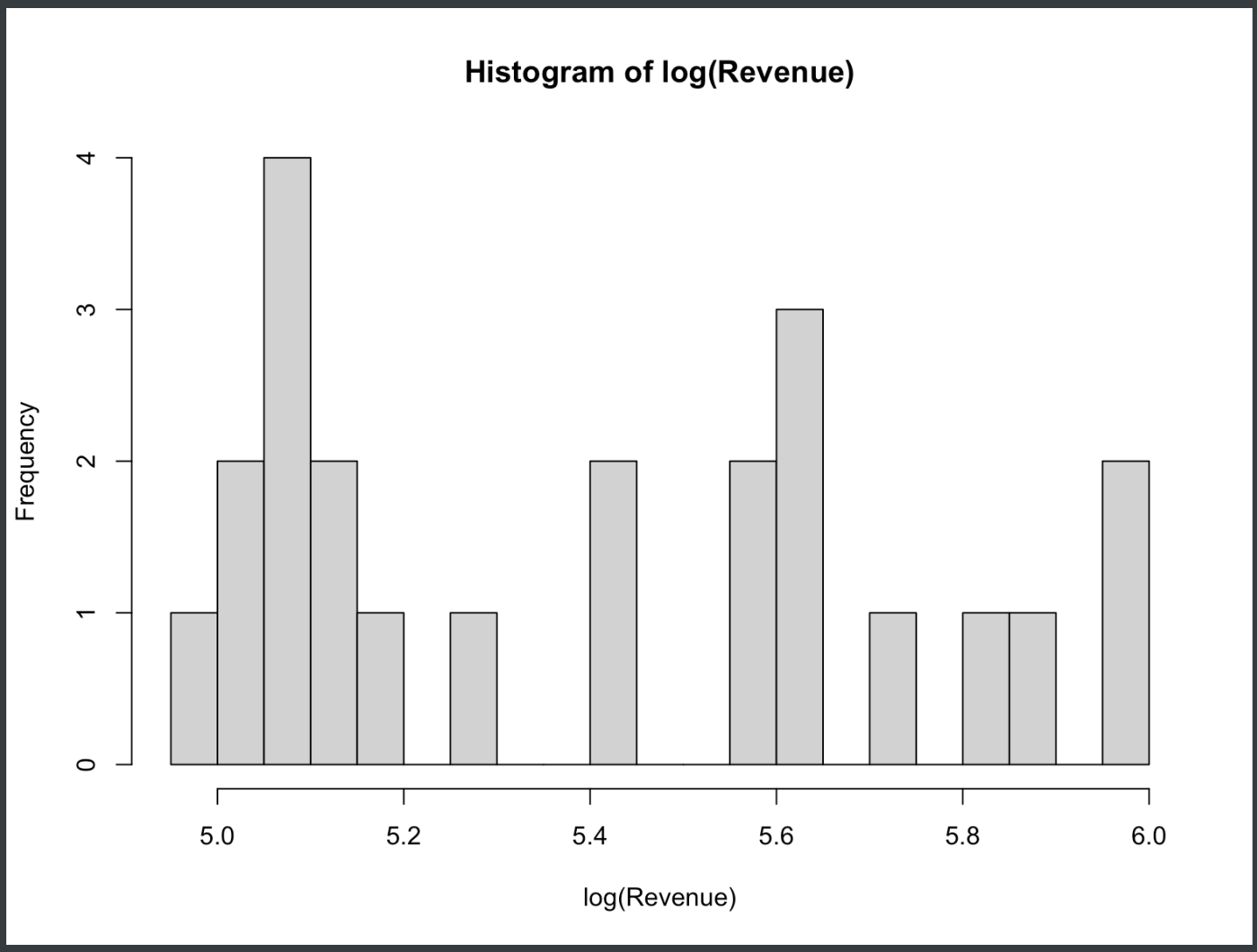
Year	Revenue
2013	387.447362
2014	406.026732
2015	425.496618
2016	445.900574

c) (3 pts) Between Model A and Model B, which model will you use? Explain your answer.

Solution:

I choose Model B. From the histogram, we see the original data is right-skewed:


Histogram of Revenue
Frequency
0
1
2
3
4
5
150
200
250
300
350
400
Revenue



The distribution become better after log, and the result is more accurate. But again, both of the model is not good, we need more data to draw strong predictions, **Model B** is just a better choice.