

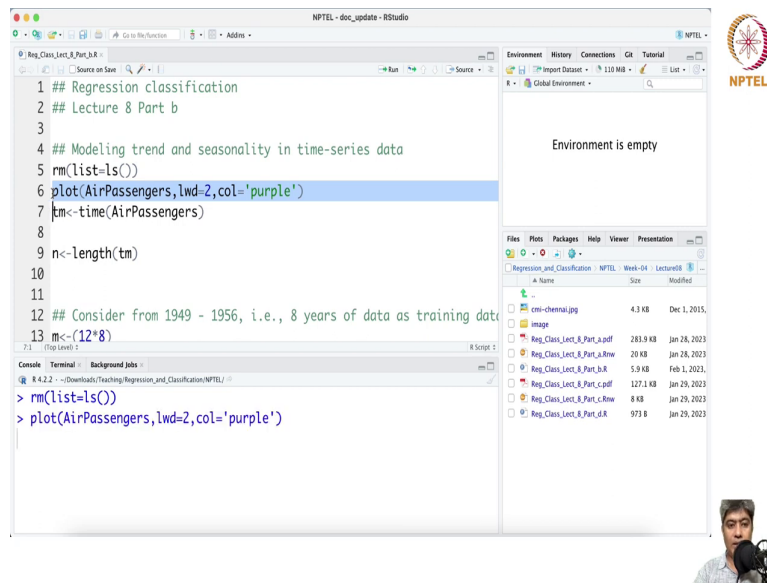
Predictive Analytics - Regression and Classification
Prof. Sourish Das
Department of Mathematics
Chennai Mathematical Institute

Lecture - 27
Hands on with R Part - 6

Hello all, welcome back to the part b of lecture series 8. In the previous video we discussed how to use simple linear regression model technique to model time series data in that case we consider two objectives; one was to a model a forecast a long trend or long term forecast using modeling the trained and seasonality and then short term forecast using autoregressive model.

In this video we are going to see how those models were actually fitted with the air passenger data.

(Refer Slide Time: 01:09)



The screenshot displays the RStudio interface with the following R code in the editor:

```
1 ## Regression classification
2 ## Lecture 8 Part b
3
4 ## Modeling trend and seasonality in time-series data
5 rm(list=ls())
6 plot(AirPassengers, lwd=2, col='purple')
7 tm<-time(AirPassengers)
8
9 n<-length(tm)
10
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<-(12*8)
```

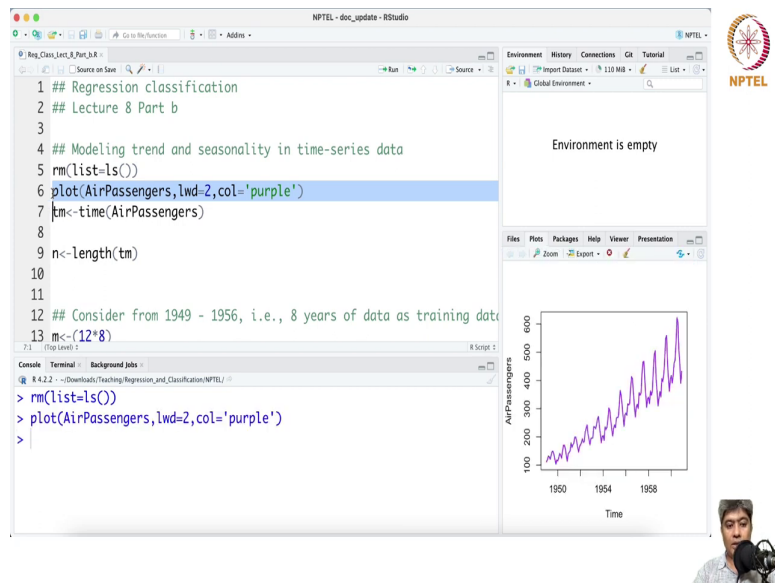
The console shows the execution of the following commands:

```
> rm(list=ls())
> plot(AirPassengers, lwd=2, col='purple')
```

The environment pane on the right indicates "Environment is empty". The file pane shows a list of files including "csi-chemical.jpg", "image", and several PDF files related to regression and classification.

So, let me go and start the r and I will share this code on the SWAYAM portal in the NPTEL portal. So, first line here is it is just remove the environment any variable are there. So, it will just remove the environment clean the environment. If you just run air passengers data it is available in the base data set.

(Refer Slide Time: 01:32)



The screenshot displays the RStudio interface. The main editor window contains the following R code:

```
1 ## Regression classification
2 ## Lecture 8 Part b
3
4 ## Modeling trend and seasonality in time-series data
5 rm(list=ls())
6 plot(AirPassengers, lwd=2, col='purple')
7 tm<-time(AirPassengers)
8
9 n<-length(tm)
10
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<-(12*8)
```

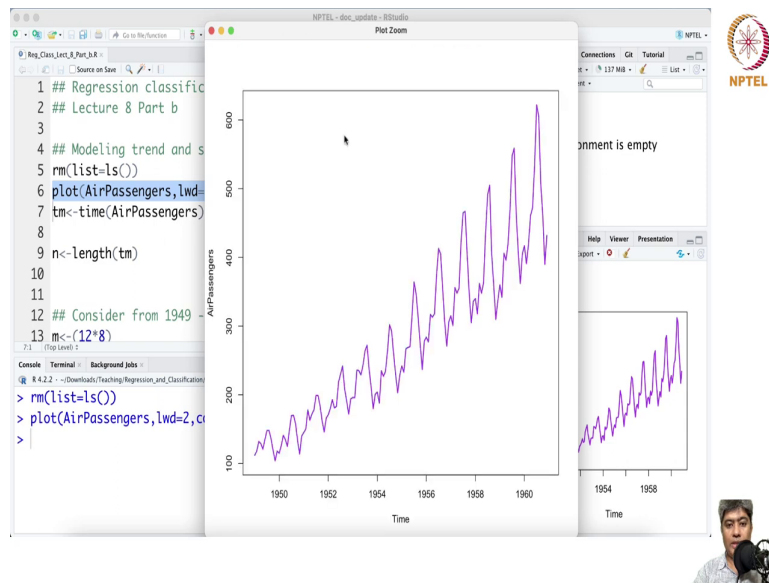
The console window shows the execution of the following commands:

```
> rm(list=ls())
> plot(AirPassengers, lwd=2, col='purple')
>
```

The environment pane on the right indicates "Environment is empty". The plot pane on the right shows a time-series plot of "AirPassengers" from 1950 to 1958. The y-axis is labeled "AirPassengers" and ranges from 100 to 600. The x-axis is labeled "Time" and shows years 1950, 1954, and 1958. The plot shows a clear upward trend with seasonal fluctuations, plotted in purple.

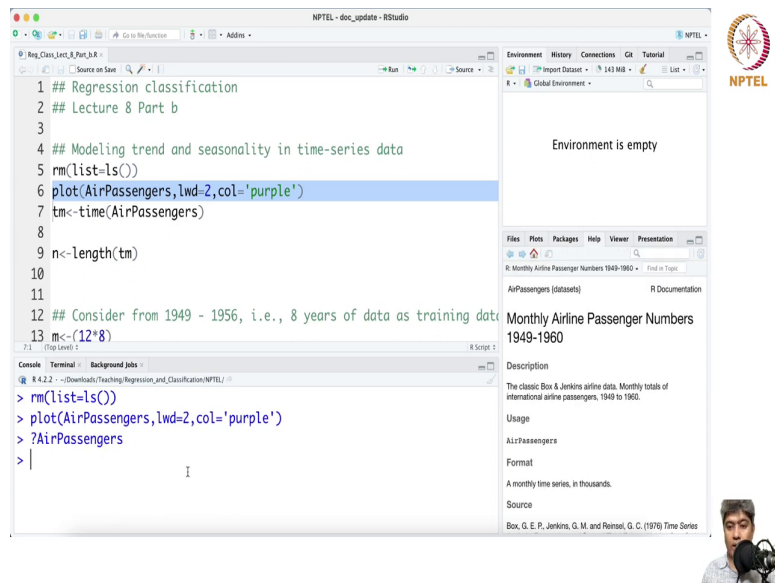
The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 01:37)



Then it will just plot the air passenger data set the data set is over the time period from 9 early 1950s to 1960s and here is the air passenger values.

(Refer Slide Time: 01:51)



The screenshot displays the RStudio interface with the following components:

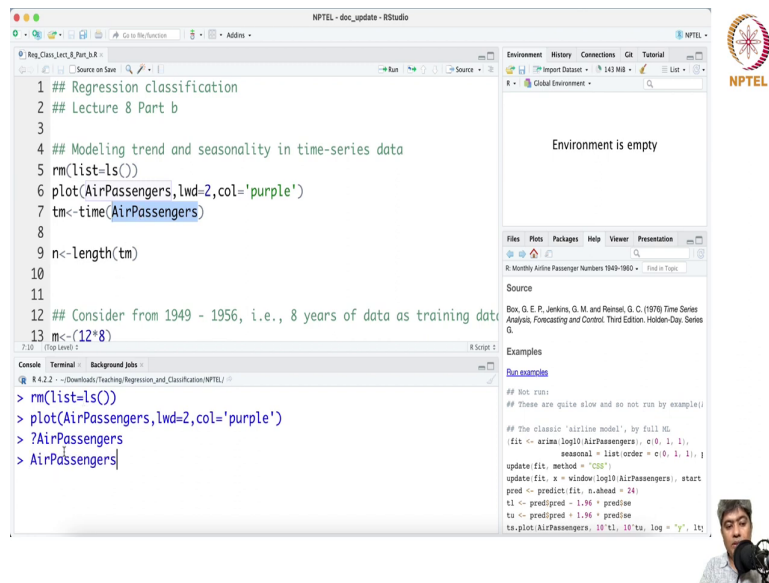
- Source Editor:** Contains R code for regression classification and time-series data analysis. The code includes comments and functions like `rm(list=ls())`, `plot(AirPassengers, lwd=2, col='purple')`, `tm<-time(AirPassengers)`, and `n<-length(tm)`. The code is as follows:

```
1 ## Regression classification
2 ## Lecture 8 Part b
3
4 ## Modeling trend and seasonality in time-series data
5 rm(list=ls())
6 plot(AirPassengers, lwd=2, col='purple')
7 tm<-time(AirPassengers)
8
9 n<-length(tm)
10
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<-tm[1:8]
```
- Environment Pane:** Shows "Environment is empty".
- Console:** Shows the execution of the code:

```
> rm(list=ls())
> plot(AirPassengers, lwd=2, col='purple')
> ?AirPassengers
> |
```
- Help Pane:** Shows the documentation for "Monthly Airline Passenger Numbers 1949-1960". The description states: "The classic Box & Jenkins airline data. Monthly totals of international airline passengers, 1949 to 1960." Usage: `AirPassengers`. Format: "A monthly time series, in thousands." Source: "Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1976) Time Series".

In fact, if you just do question mark air passengers here it is talking about the classic box gains airline data set, monthly total of international air passengers between 1949 and 1960.

(Refer Slide Time: 02:13)



```
1 ## Regression classification
2 ## Lecture 8 Part b
3
4 ## Modeling trend and seasonality in time-series data
5 rm(list=ls())
6 plot(AirPassengers,lwd=2,col='purple')
7 tm<-time(AirPassengers)
8
9 n<-length(tm)
10
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<-(12*8)
```

```
> rm(list=ls())
> plot(AirPassengers,lwd=2,col='purple')
> ?AirPassengers
> AirPassengers
```

Environment is empty

Source

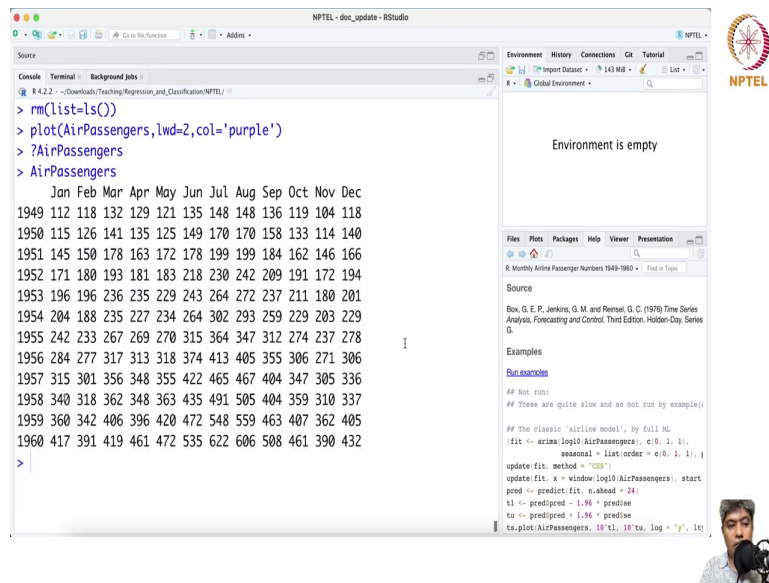
Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1976) Time Series Analysis, Forecasting and Control, Third Edition. Holden-Day, Series G.

Examples

```
## Not run:
## These are quite slow and do not run by example!
## The classic 'airline model', by full ML
fit <- arima(log10(AirPassengers), c(0, 1, 1),
             seasonal = list(order = c(0, 1, 1),
                             update:fit, method = "COP"))
update:fit, n = window(log10(AirPassengers), start
pred <- predict:fit, n.ahead = 24)
t1 <- predictpred - 1.96 * predise
t2 <- predictpred + 1.96 * predise
ts.plot(AirPassengers, 30:t1, 10:t2, log = "y", lty
```

So, here is the data and if you just run this in fact, this data just copy and paste it and just run it.

(Refer Slide Time: 02:21)



```
rm(list=ls())
> plot(AirPassengers, lwd=2, col='purple')
> ?AirPassengers
> AirPassengers
  Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1949 112 118 132 129 121 135 148 148 136 119 104 118
1950 115 126 141 135 125 149 170 170 158 133 114 140
1951 145 150 178 163 172 178 199 199 184 162 146 166
1952 171 180 193 181 183 218 230 242 209 191 172 194
1953 196 196 236 235 229 243 264 272 237 211 180 201
1954 204 188 235 227 234 264 302 293 259 229 203 229
1955 242 233 267 269 270 315 364 347 312 274 237 278
1956 284 277 317 313 318 374 413 405 355 306 271 306
1957 315 301 356 348 355 422 465 467 404 347 305 336
1958 340 318 362 348 363 435 491 505 404 359 310 337
1959 360 342 406 396 420 472 548 559 463 407 362 405
1960 417 391 419 461 472 535 622 606 508 461 390 432
>
```

Environment is empty

```
## Not run:
## These are quite slow and so not run by example()
## The classic 'airline model', by full ML
fit <- arima(log10(AirPassengers), c(0, 1, 1),
            seasonal = list(order = c(0, 1, 1),
                             update:fit, method = "CSP"))
update:fit, n = window(log10(AirPassengers), start
pred <- predict:fit, n.ahead = 24)
t1 <- predictpred - 1.96 * predictse
t2 <- predictpred + 1.96 * predictse
ts.plot(AirPassengers, 30:t1, 10:ts, log = "y", lty =
```

You can see the entire data set its a small data set where you can see the entire data set Jan, February, March, April, May up to December and then if each row sort of or a particular year. So, its a nice data set which you can see and now we are going to model it.

(Refer Slide Time: 02:46)



The screenshot shows the RStudio interface with the following code in the editor:

```
1 ## Regression classification
2 ## Lecture 8 Part b
3
4 ## Modeling trend and seasonality in time-series data
5 rm(list=ls())
6 plot(AirPassengers, lwd=2, col='purple')
7 tm<-time(AirPassengers)
8
9 n<-length(tm)
10
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<-(12*8)
```

The console output shows the following data:

```
R 4.2.2 - ~/Downloads/Teaching-Regression_and_Classification/NPTEL/ <
1956 284 277 317 313 318 374 413 405 355 306 271 306
1957 315 301 356 348 355 422 465 467 404 347 305 336
1958 340 318 362 348 363 435 491 505 404 359 310 337
1959 360 342 406 396 420 472 548 559 463 407 362 405
1960 417 391 419 461 472 535 622 606 508 461 390 432
> tm<-time(AirPassengers)
>
```

The right-hand side of the RStudio window shows the 'Values' pane with 'tm' as a 'Time-Series [1:144...]' and the 'Source' pane with a reference to Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1976) Time Series Analysis, Forecasting and Control, Third Edition. Holden-Day, Series G.



Now from here I am going to extract the time.

(Refer Slide Time: 02:49)

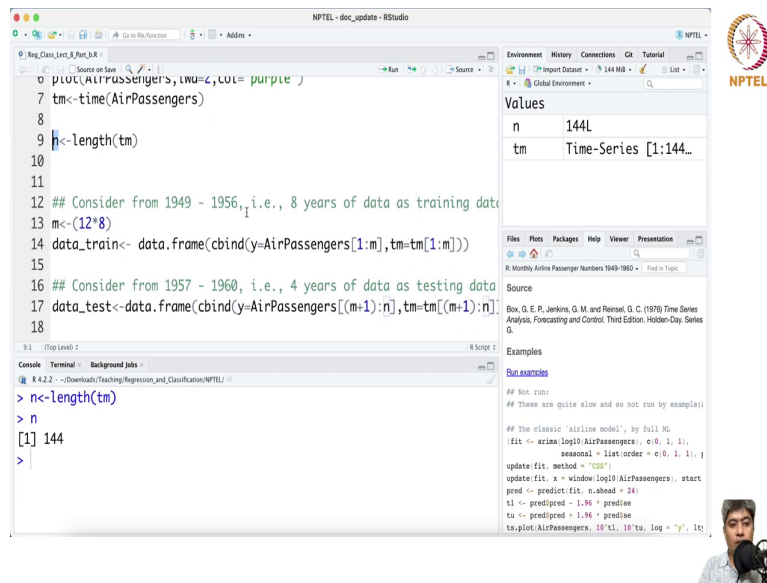
The screenshot shows the RStudio environment with the following components:

- Console:** Displays the execution of `tm <- time(AirPassengers)` and the resulting output of the `tm` object, which is a time series with monthly intervals from 1949 to 1960.
- Environment:** Shows the `tm` object as a Time-Series [1:144...].
- Source:** Contains R code for fitting an ARIMA model to the `tm` object and plotting the results.

```
1957 315 301 356 348 355 422 465 467 404 347 305 336
1958 340 318 362 348 363 435 491 505 404 359 310 337
1959 360 342 406 396 420 472 548 559 463 407 362 405
1960 417 391 419 461 472 535 622 606 508 461 390 432
> tm <- time(AirPassengers)
> tm
      Jan  Feb  Mar  Apr  May  Jun  Jul
1949 1949.000 1949.083 1949.167 1949.250 1949.333 1949.417 1949.500
1950 1950.000 1950.083 1950.167 1950.250 1950.333 1950.417 1950.500
1951 1951.000 1951.083 1951.167 1951.250 1951.333 1951.417 1951.500
1952 1952.000 1952.083 1952.167 1952.250 1952.333 1952.417 1952.500
1953 1953.000 1953.083 1953.167 1953.250 1953.333 1953.417 1953.500
1954 1954.000 1954.083 1954.167 1954.250 1954.333 1954.417 1954.500
1955 1955.000 1955.083 1955.167 1955.250 1955.333 1955.417 1955.500
1956 1956.000 1956.083 1956.167 1956.250 1956.333 1956.417 1956.500
1957 1957.000 1957.083 1957.167 1957.250 1957.333 1957.417 1957.500
1958 1958.000 1958.083 1958.167 1958.250 1958.333 1958.417 1958.500
1959 1959.000 1959.083 1959.167 1959.250 1959.333 1959.417 1959.500
1960 1960.000 1960.083 1960.167 1960.250 1960.333 1960.417 1960.500
      Aug  Sep  Oct  Nov  Dec
1960 1960.583 1960.667 1960.750 1960.833 1960.917
```

So, `tm` if I just plot the `tm`, you can see that Jan, Feb, March every month the time is being converted into a numerical value.

(Refer Slide Time: 03:08)



The screenshot shows an RStudio interface with the following code in the editor:

```
plot(AirPassengers, lwd=2, col= purple )
7 tm<-time(AirPassengers)
8
9 h<-length(tm)
10
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<-(12*8)
14 data_train<- data.frame(cbind(y=AirPassengers[1:m],tm=tm[1:m]))
15
16 ## Consider from 1957 - 1960, i.e., 4 years of data as testing data
17 data_test<-data.frame(cbind(y=AirPassengers[(m+1):n],tm=tm[(m+1):n])
18
```

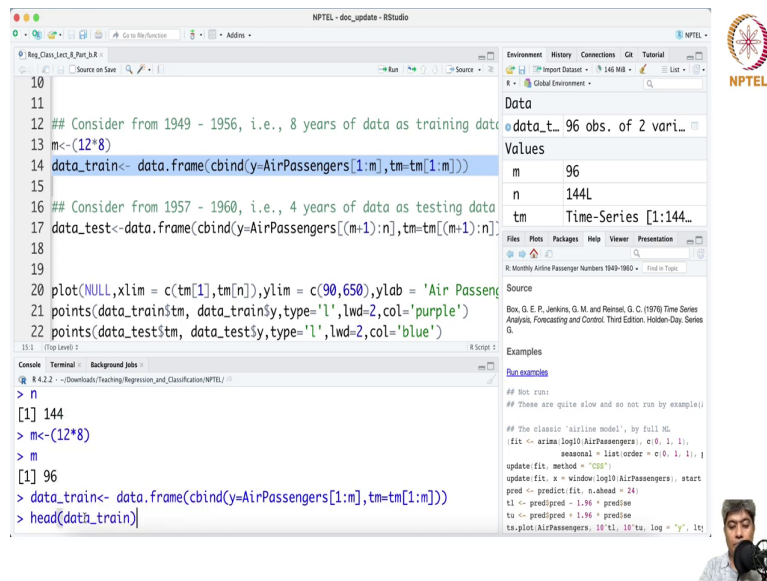
The console output shows:

```
> n
[1] 144
```

The Environment pane on the right shows the variable 'tm' as a Time-Series [1:144..]. The console also shows a small video feed of the presenter in the bottom right corner.

So, we call it tm and n is the length of the tm. So, n is so, about 144 values are there. So, we consider from 1949 to 1956 that is 8 years of data as training data. So, 12 into 8 every month we have 12 months of data over 8 years.

(Refer Slide Time: 03:35)



The screenshot displays the RStudio interface with the following R code in the script editor:

```
10
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<-(12*8)
14 data_train<- data.frame(cbind(y=AirPassengers[1:m],tm=tm[1:m]))
15
16 ## Consider from 1957 - 1960, i.e., 4 years of data as testing data
17 data_test<-data.frame(cbind(y=AirPassengers[(m+1):n],tm=tm[(m+1):n])
18
19
20 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air Passeng
21 points(data_train$tm, data_train$y,type='l',lwd=2,col='purple')
22 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
```

The console output shows the execution of the code:

```
> n
[1] 144
> m<-(12*8)
> m
[1] 96
> data_train<- data.frame(cbind(y=AirPassengers[1:m],tm=tm[1:m]))
> head(data_train)
```

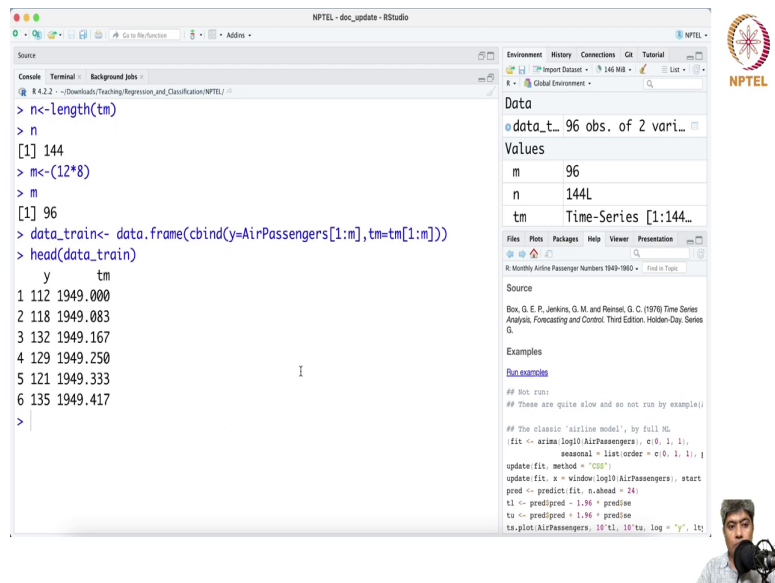
The right-hand side of the RStudio window shows the 'Data' pane with the following information:

Variable	Value
m	96
n	144L
tm	Time-Series [1:144_

The 'Source' pane shows the citation for the 'AirPassengers' dataset: Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1976) Time Series Analysis, Forecasting and Control, Third Edition. Holden-Day, Series G.

So, we use we are going to use 96 data points out of the; out of the 144 data points as training data. So, I am defining my training data as data frame where y equal to AirPassengers from 1 is to m and tm stands for time. So, if I just now write head of data sorry header data_train.

(Refer Slide Time: 04:09)



```
Source
> n<-length(tm)
> n
[1] 144
> m<-(12*8)
> m
[1] 96
> data_train<- data.frame(cbind(y=AirPassengers[1:m],tm=tm[1:m]))
> head(data_train)
  y      tm
1 112 1949.000
2 118 1949.083
3 132 1949.167
4 129 1949.250
5 121 1949.333
6 135 1949.417
>
```

Environment History Connections Git Tutorial

Import Dataset 146 MB List Global Environment

Data

data_train 96 obs. of 2 vari...

Values

m	96
n	144L
tm	Time-Series [1:144_

Files Plots Packages Help Viewer Presentation

R: Monthly Airline Passenger Numbers 1949-1960 Find in Topic

Source

Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1976) Time Series Analysis, Forecasting and Control, Third Edition, Holden-Day, Series G.

Examples

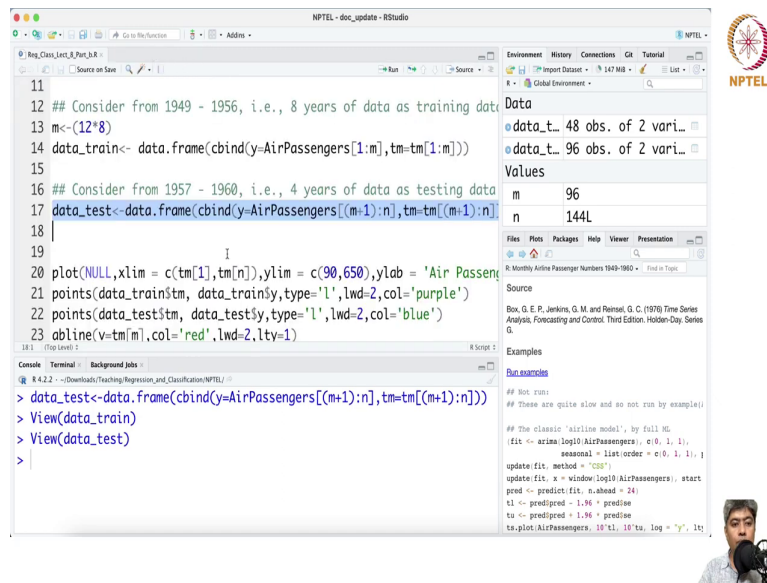
[Run examples](#)

```
## Not run:
## These are quite slow and so not run by example!

## The classic 'airline model', by full ML
fit <- arima(log10(AirPassengers), c(0, 1, 1),
             seasonal = list(order = c(0, 1, 1),
                              method = "CSP"))
update: fit: method = "CSP"
update: fit: n = window(log10(AirPassengers), start
pred <- predict(fit, n.ahead = 24)
t1 <- predictpred + 1.96 * predise
t2 <- predictpred - 1.96 * predise
ts.plot(AirPassengers, 10:t1, 10:t2, log = "y", lty
```

So, you can see that this is the data on the first few days of data, few months of data ok.

(Refer Slide Time: 04:20)



```
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<- (12*8)
14 data_train<- data.frame(cbind(y=AirPassengers[1:m],tm=tm[1:m]))
15
16 ## Consider from 1957 - 1960, i.e., 4 years of data as testing data
17 data_test<- data.frame(cbind(y=AirPassengers[(m+1):n],tm=tm[(m+1):n]))
18
19
20 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air Passengers',
21 points(data_train$tm, data_train$y,type='l',lwd=2,col='purple')
22 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
23 abline(v=tm[m],col='red',lwd=2,lty=1)
```

Environment: Global Environment

Variable	Value
m	96
n	144L

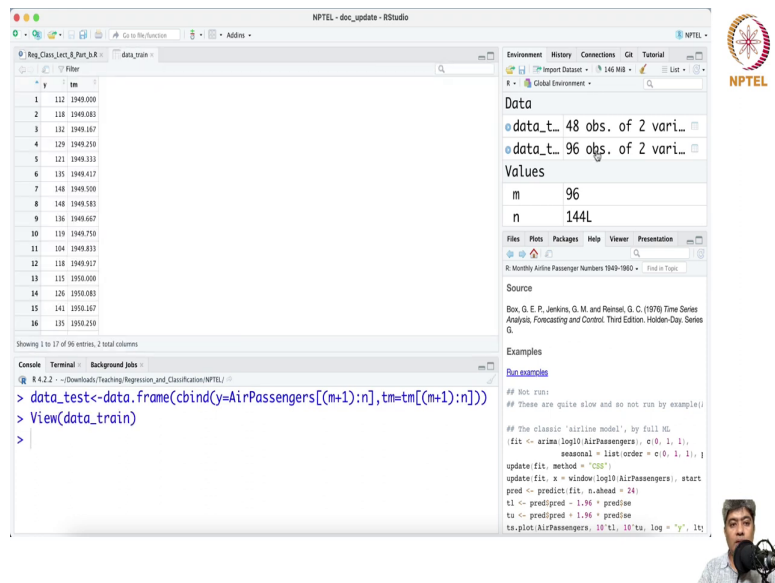
```
> data_test<-data.frame(cbind(y=AirPassengers[(m+1):n],tm=tm[(m+1):n]))
> View(data_train)
> View(data_test)
```

Examples

```
## Not run:
## These are quite slow and do not run by example!
## The classic 'airline model', by full ML
fit <- arima(log10(AirPassengers), c(0, 1, 1),
seasonal = list(order = c(0, 1, 1),
update:fit: method = "CSP")
update:fit: n = window(log10(AirPassengers), start
pred <- predict(fit, n.ahead = 24)
t1 <- pred$pred - 1.96 * pred$se
t2 <- pred$pred + 1.96 * pred$se
ts.plot(AirPassengers, 30:t1, 10:t2, log = "y", lty
```

Now, we are going to consider from 1957 to 1960 that is 57,58, 59 and 60 4 years of data as testing data. So, about one two third data we are going to use for training and one-third data we are going to use it for testing ok. So, here is the we are running it as a testing.

(Refer Slide Time: 04:44)



The screenshot shows the RStudio interface with the following components:

- Environment:** Shows 'data_train' with 48 observations of 2 variables and 'data_test' with 96 observations of 2 variables. Values for 'm' are 96 and 'n' are 144L.
- Source:** Cites 'Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1976) Time Series Analysis, Forecasting and Control, Third Edition, Holden-Day, Series G.'
- Console:** Contains the following R code:

```
> data_test <- data.frame(cbind(y=AirPassengers[(m+1):n], tm=tm[(m+1):n]))
> View(data_train)
>
```
- Examples:** Shows R code for fitting an ARIMA model:

```
## Not run:
## These are quite slow and do not run by example!
## The classic 'airline model', by full ML
fit <- arima(log10(AirPassengers), c(0, 1, 1),
             seasonal = list(order = c(0, 1, 1),
                             method = "CSP"))
update: fit ~ window(log10(AirPassengers), start
pred <- predict(fit, n.ahead = 24)
t1 <- predictpred + 1.96 * predise
t2 <- predictpred - 1.96 * predise
ts.plot(AirPassengers, 30:t1, 10:t2, log = "y", lty =
```

So, here is the test data set it is a trained data set and this is the test data set ok.

(Refer Slide Time: 04:51)

The screenshot displays the RStudio interface with the following components:

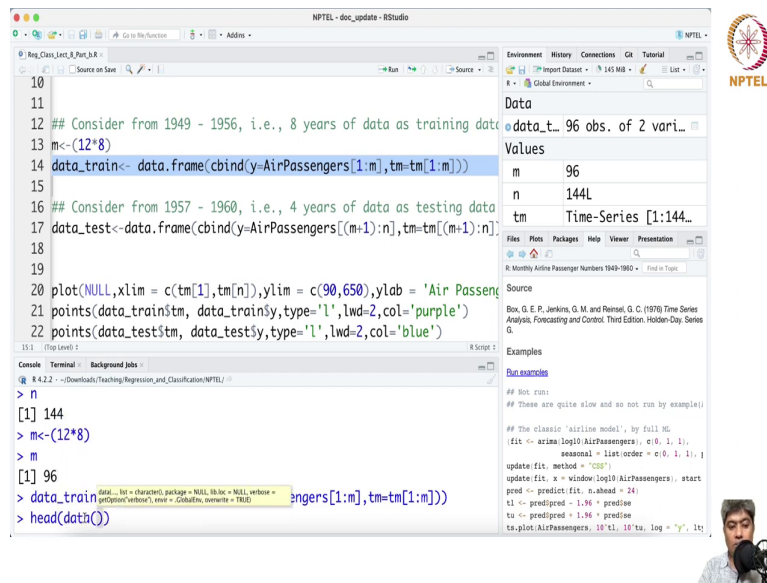
- Environment:** Shows 'data_train' with 48 observations and 2 variables, and 'data_test' with 96 observations and 2 variables.
- Console:** Contains the following R code:

```
> data_test <- data.frame(cbind(y=AirPassengers[(m+1):n], tm=tm[(m+1):n]))
> View(data_train)
> View(data_test)
>
```
- Source:** Shows the R script code for fitting an ARIMA model:

```
## Not run:
## These are quite slow and do not run by example!
## The classic 'airline model', by full ML
fit <- arima(log10(AirPassengers), c(0, 1, 1),
  seasonal = list(order = c(0, 1, 1),
update: fit: method = "COP")
update: fit: n = window(log10(AirPassengers), start
pred <- predict(fit, n.ahead = 24)
t1 <- predipred + 1.96 * predise
t2 <- predipred - 1.96 * predise
ts.plot(AirPassengers, 30:t1, 10:t2, log = "y", lty
```
- Terminal:** Shows the R version: 'R 4.2.2'.

The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 04:56)



The screenshot shows the RStudio interface with the following R code in the editor:

```
10
11
12 ## Consider from 1949 - 1956, i.e., 8 years of data as training data
13 m<- (12*8)
14 data_train<- data.frame(cbind(y=AirPassengers[1:m],tm=tm[1:m]))
15
16 ## Consider from 1957 - 1960, i.e., 4 years of data as testing data
17 data_test<-data.frame(cbind(y=AirPassengers[(m+1):n],tm=tm[(m+1):n])
18
19
20 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air Passeng
21 points(data_train$tm, data_train$y,type='l',lwd=2,col='purple')
22 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
```

The console output shows the following commands and results:

```
> n
[1] 144
> m<- (12*8)
> m
[1] 96
> data_train
data_train: data.frame(cbind(y=AirPassengers[1:m],tm=tm[1:m]))
> head(data_train)
```

The right-hand side of the RStudio window displays the 'Data' pane with the following information:

Variable	Value
m	96
n	144
tm	Time-Series [1:144]

The 'Source' pane shows the following text:

```
Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. (1976) Time Series
Analysis, Forecasting and Control, Third Edition. Holden-Day, Series
G.
```

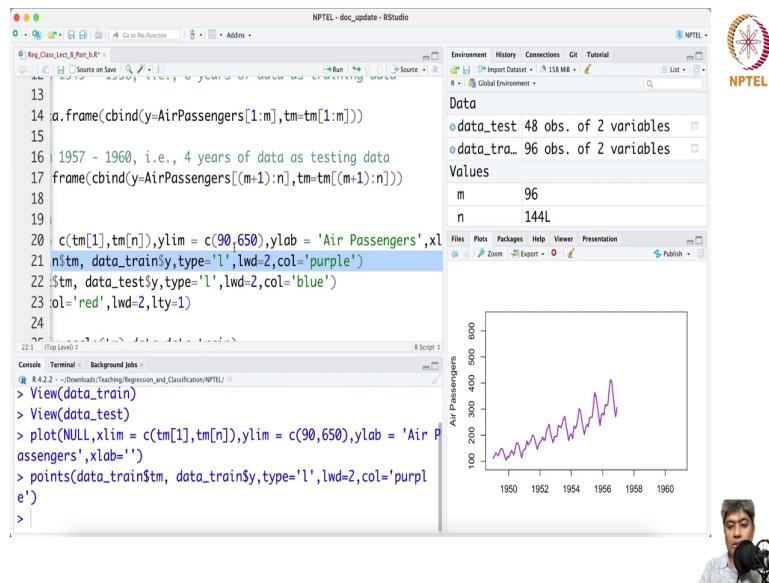
The 'Examples' pane shows the following code:

```
## Not run:
## These are quite slow and do not run by example!
## The classic 'airline model', by full ML
fit <- arima(log10(AirPassengers), c(0, 1, 1),
seasonal = list(order = c(0, 1, 1),
update:fit: method = "ML")
update:fit: n = window(log10(AirPassengers), start
pred <- predict(fit, n.ahead = 24)
t1 <- pred$pred - 1.96 * pred$se
t2 <- pred$pred + 1.96 * pred$se
ts.plot(AirPassengers, 30:t1, 10:t2, log = "y", lty
```

Now, this piece of line is going to create the just that frame of the plot I am going to create some visualization let us as our discussion you know that visualization is sometimes helps to understand the it helps us to understand the data and what kind of model can be used.

So, here I am just creating a framework just on the exact same thing do not plot anything just x axis will have the times and y axis will have the dates. Then from the time train you plot the data with purple color.

(Refer Slide Time: 05:46)



The screenshot displays the RStudio interface with the following components:

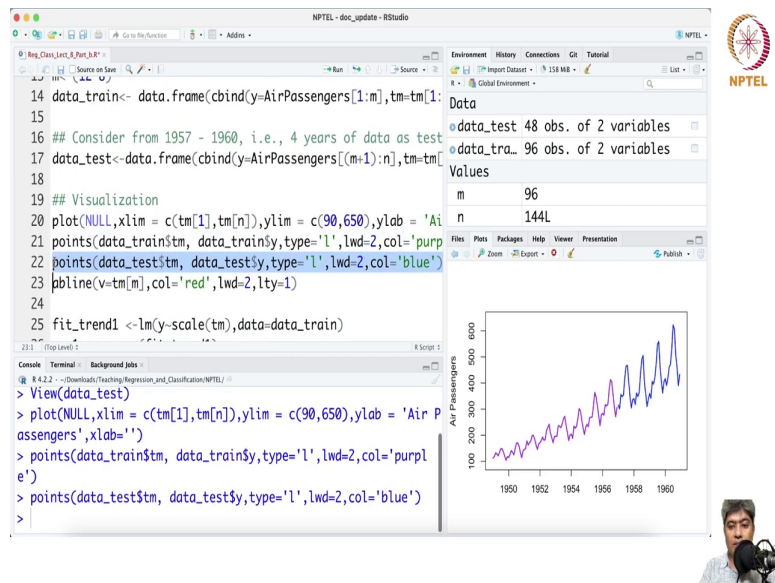
- Source Editor:** Contains R code for data manipulation:

```
13  
14 a.frame(cbind(y=AirPassengers[1:m],tm=tm[1:m]))  
15  
16 1957 - 1960, i.e., 4 years of data as testing data  
17 frame(cbind(y=AirPassengers[(m+1):n],tm=tm[(m+1):n]))  
18  
19  
20 c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air Passengers',xlab =  
21 tm, data_train$y,type='l',lwd=2,col='purple')  
22 $tm, data_test$y,type='l',lwd=2,col='blue')  
23 col='red',lwd=2,ty=1)  
24
```
- Environment:** Shows two data objects:
 - `data_test`: 48 observations of 2 variables.
 - `data_train`: 96 observations of 2 variables.
- Values:** A table showing the dimensions of the data objects:

Variable	Value
m	96
n	144L
- Console:** Shows the execution of the following commands:

```
> View(data_train)  
> View(data_test)  
> plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air Passengers',xlab='')  
> points(data_train$tm, data_train$y,type='l',lwd=2,col='purple')  
>
```
- Plot:** A line plot titled "Air Passengers" showing the number of passengers from 1950 to 1960. The y-axis ranges from 100 to 600. The plot shows a clear upward trend with seasonal fluctuations. The data points are plotted in purple.
- Background:** A small video inset shows a person speaking into a microphone.
- Logos:** The NPTEL logo is visible in the top right corner.

(Refer Slide Time: 05:51)



The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for data splitting and visualization:

```
14 data_train<- data.frame(cbind(y=AirPassengers[1:m],tm=tm[1:
15
16 ## Consider from 1957 - 1960, i.e., 4 years of data as test
17 data_test<-data.frame(cbind(y=AirPassengers[(m+1):n],tm=tm[
18
19 ## Visualization
20 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Ai
21 points(data_train$tm, data_train$y,type='l',lwd=2,col='purp
22 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
23 abline(v=tm[m],col='red',lwd=2,ltty=1)
24
25 fit_trend1 <-lm(y~scale(tm),data=data_train)
```
- Environment:** Shows the loaded data objects:
 - data_test: 48 obs. of 2 variables
 - data_train: 96 obs. of 2 variables
- Values:** A table showing the number of observations (n) for each variable (m and n):

Variable	Value
m	96
n	144L
- Console:** Shows the execution of the code:

```
> View(data_test)
> plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air P
assengers',xlab='')
> points(data_train$tm, data_train$y,type='l',lwd=2,col='purpl
e')
> points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
>
```
- Plot:** A line plot titled 'Air Passengers' showing the number of passengers over time. The x-axis represents years from 1950 to 1960, and the y-axis represents the number of passengers from 100 to 600. The plot shows a clear upward trend with seasonal fluctuations. A vertical red line is drawn at the year 1957, separating the training data (purple line) from the test data (blue line).

And then from the test data set you plot the data with the blue color.

(Refer Slide Time: 05:56)

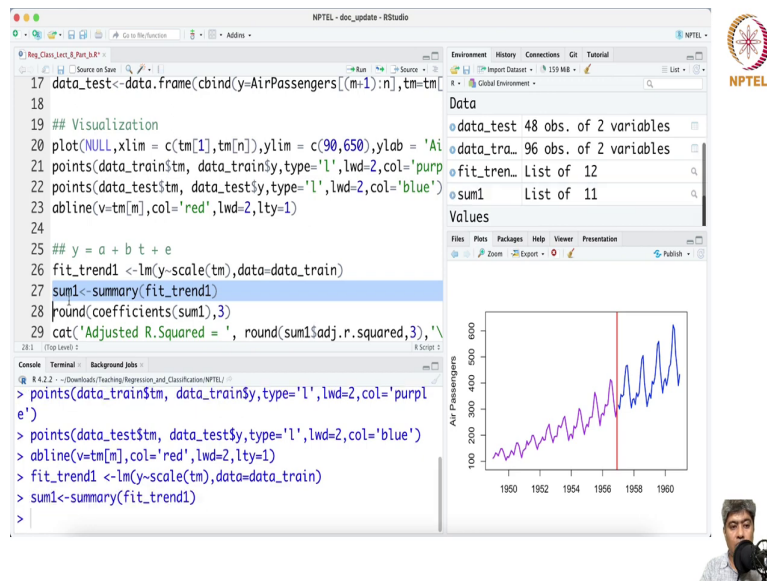
The screenshot displays an RStudio interface with the following components:

- Script Editor:** Contains R code for data manipulation and visualization:

```
17 data_test<-data.frame(cbind(y=AirPassengers[(m+1):n],tm=tm)
18
19 ## Visualization
20 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air P
21 points(data_train$tm, data_train$y,type='l',lwd=2,col='purpl
22 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
23 abline(v=tm[m],col='red',lwd=2,lty=1)
24
25 fit_trend1 <-lm(y~scale(tm),data=data_train)
26 sum1<-summary(fit_trend1)
27 round(coefficients(sum1),3)
28 cat('Adjusted R.Squared = ', round(sum1$adj.r.squared,3), '\n'
29
```
- Environment:** Shows two data objects: `data_test` (48 obs. of 2 variables) and `data_train` (96 obs. of 2 variables).
- Values:** A table showing the number of observations: `m = 96` and `n = 144L`.
- Plot:** A line plot titled 'Air Passengers' showing the number of passengers from 1950 to 1960. The y-axis ranges from 100 to 600. The plot features purple lines for training data (1950-1956), blue lines for test data (1957-1960), and a vertical red line at the year 1956.5.
- Console:** Shows the execution of the plot commands from the script.

And this is the red line that we draw. So, this part of the data we are going to use as a train data and this part of the data we are going to use as a test data.

(Refer Slide Time: 06:14)



The screenshot displays an RStudio interface with the following components:

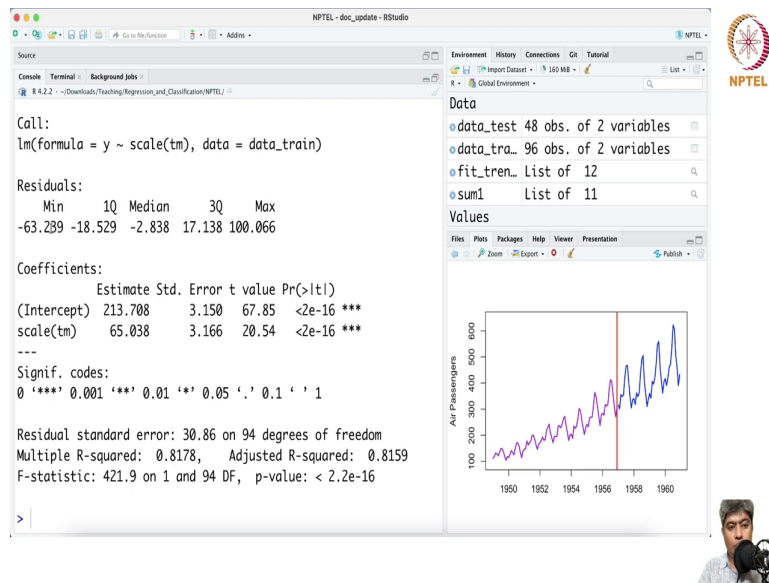
- Source Editor:** Contains R code for data preparation and visualization. The code includes:

```
17 data_test<-data.frame(cbind(y=AirPassengers[(m+1):n],tm=tm)
18
19 ## Visualization
20 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air
21 points(data_train$tm, data_train$y,type='l',lwd=2,col='purpl
22 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
23 abline(v=tm[m],col='red',lwd=2,lty=1)
24
25 ## y = a + b t + e
26 fit_trend1 <-lm(y~scale(tm),data=data_train)
27 sum1<-summary(fit_trend1)
28 round(coefficients(sum1),3)
29 cat('Adjusted R.Squared = ', round(sum1$adj.r.squared,3), '\n')
```
- Environment:** Lists objects in the workspace: data_test (48 obs. of 2 variables), data_tra_ (96 obs. of 2 variables), fit_tren_ (List of 12), and sum1 (List of 11).
- Console:** Shows the execution of the code from the source editor, including the output of the summary function.

```
> points(data_train$tm, data_train$y,type='l',lwd=2,col='purpl
e')
> points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
> abline(v=tm[m],col='red',lwd=2,lty=1)
> fit_trend1 <-lm(y~scale(tm),data=data_train)
> sum1<-summary(fit_trend1)
>
```
- Plots:** A line plot titled 'Air Passengers' showing the number of passengers from 1950 to 1960. The y-axis ranges from 100 to 600. The plot features a purple line for training data, a blue line for test data, and a vertical red line at the year 1956. A linear regression line is also visible, showing a positive trend.
- Background:** The NPTEL logo is visible in the top right corner, and a small video feed of a person is in the bottom right corner.

So, the first model we are going to fit is y equal to a plus b time t plus error ok, now we just fit the data summary.

(Refer Slide Time: 06:31)



The screenshot displays the RStudio interface. The console window shows the following output for a linear regression model:

```
Call:
lm(formula = y ~ scale(tm), data = data_train)

Residuals:
    Min       1Q   Median       3Q      Max
-63.289 -18.529  -2.838  17.138 100.066



Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  213.708     3.150   67.85  <2e-16 ***
scale(tm)    65.038     3.166   20.54  <2e-16 ***
---
Signif. codes:
  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 30.86 on 94 degrees of freedom
Multiple R-squared:  0.8178,    Adjusted R-squared:  0.8159
F-statistic: 421.9 on 1 and 94 DF,  p-value: < 2.2e-16
```

The environment pane on the right shows the following data objects:

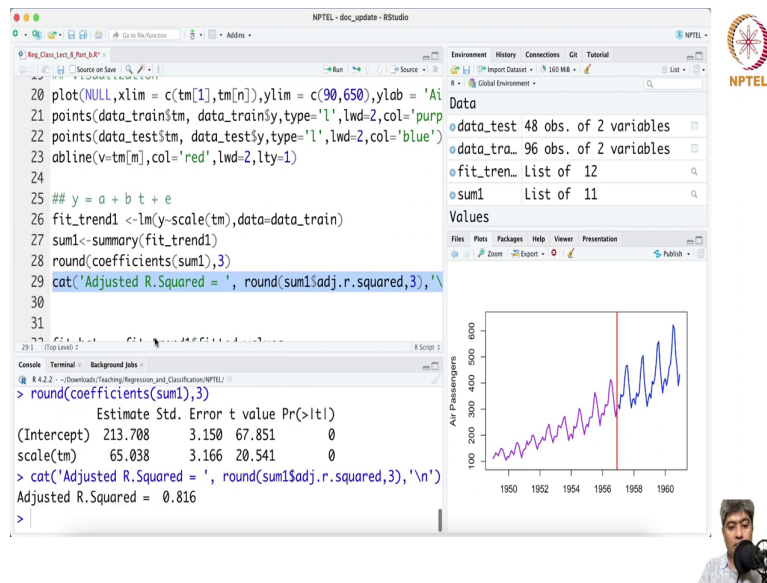
- data_test: 48 obs. of 2 variables
- data_tra: 96 obs. of 2 variables
- fit_tren: List of 12
- sum1: List of 11

The plot pane shows a time series plot of 'Air Passengers' from 1950 to 1960. The y-axis ranges from 100 to 600. The plot shows a clear upward trend with seasonal fluctuations. A vertical red line is drawn at approximately 1956.5.



And this is the summary. So, this is what we have seen it in our presentation also.

(Refer Slide Time: 06:41)



The screenshot shows an RStudio window with the following code in the script editor:

```
20 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air  
21 points(data_train$tm, data_train$y,type='l',lwd=2,col='purple'  
22 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')  
23 abline(v=tm[m],col='red',lwd=2,lty=1)  
24  
25 ## y = a + b t + e  
26 fit_trend1 <-lm(y~scale(tm),data=data_train)  
27 sum1<-summary(fit_trend1)  
28 round(coefficients(sum1),3)  
29 cat('Adjusted R.Squared = ', round(sum1$adj.r.squared,3), '\n'  
30  
31
```

The console output shows the following results:

```
> round(coefficients(sum1),3)  
              Estimate Std. Error t value Pr(>|t|)  
(Intercept) 213.708      3.150  67.851    0  
scale(tm)    65.038      3.166  20.541    0  
> cat('Adjusted R.Squared = ', round(sum1$adj.r.squared,3), '\n')  
Adjusted R.Squared = 0.816  
>
```

The plot shows 'Air Passengers' on the y-axis (ranging from 100 to 600) and years on the x-axis (ranging from 1950 to 1960). The data points are shown as purple and blue lines, and a red vertical line is drawn at approximately 1956.5.

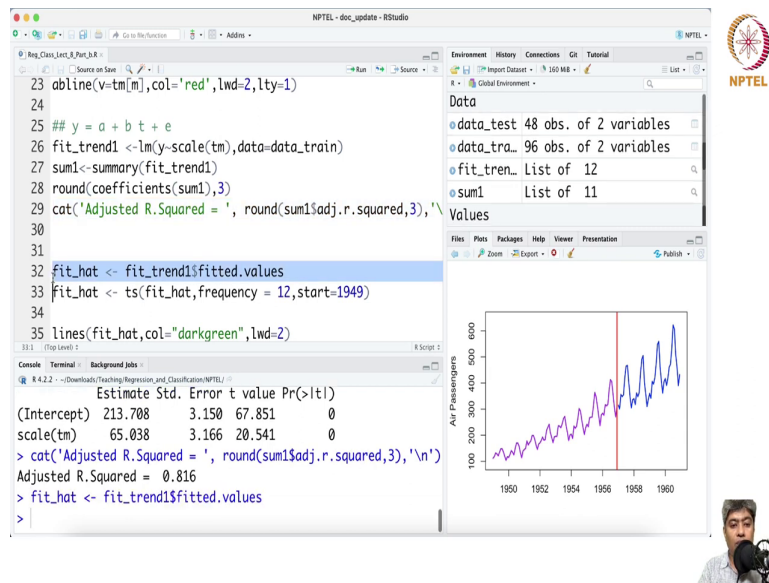
The Environment pane on the right shows the following objects:

- data_test: 48 obs. of 2 variables
- data_train: 96 obs. of 2 variables
- fit_trend1: List of 12
- sum1: List of 11

The NPTEL logo is visible in the top right corner of the RStudio window.

If we just so, this is what we have seen and adjusted r square is 81.6 percent so; that means, 81.6 percent of the variability in the air passenger data can be simply explained by the simple linear trend.

(Refer Slide Time: 07:05)



The screenshot displays the RStudio interface. The main editor window contains the following R code:

```
23 abline(v=tm[col='red',lwd=2,lt=1])
24
25 ## y = a + b t + e
26 fit_trend1 <- lm(y=scale(tm),data=data_train)
27 sum1<-summary(fit_trend1)
28 round(coefficients(sum1),3)
29 cat('Adjusted R.Squared = ', round(sum1$adj.r.squared,3), '\n')
30
31
32 fit_hat <- fit_trend1$fitted.values
33 fit_hat <- ts(fit_hat,frequency = 12,start=1949)
34
35 lines(fit_hat,col="darkgreen",lwd=2)
```

The console window shows the output of the code:

```
Estimate Std. Error t value Pr(>|t|)
(Intercept) 213.708      3.150  67.851    0
scale(tm)   65.038      3.166  20.541    0
> cat('Adjusted R.Squared = ', round(sum1$adj.r.squared,3), '\n')
Adjusted R.Squared = 0.816
> fit_hat <- fit_trend1$fitted.values
>
```

The Environment pane on the right shows the following objects:

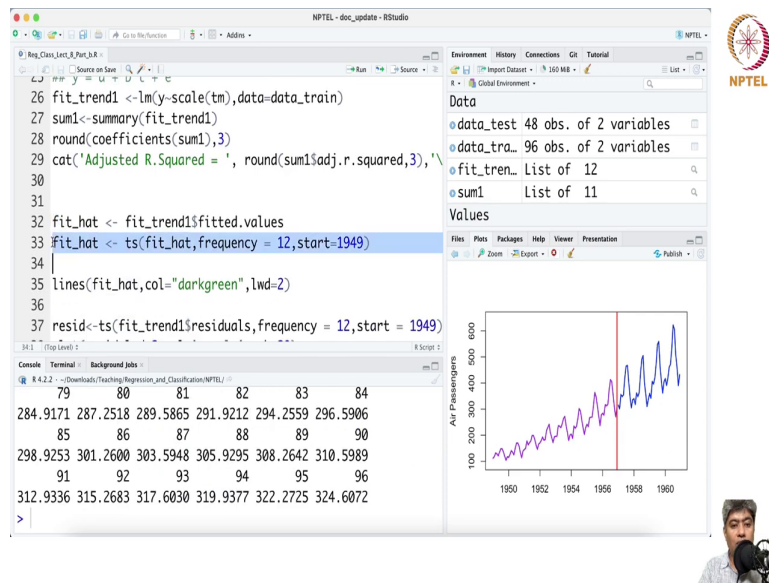
- data_test: 48 obs. of 2 variables
- data_tra: 96 obs. of 2 variables
- fit_tren: List of 12
- sum1: List of 11

The Plots pane shows a time series plot of 'Air Passengers' from 1950 to 1960. The y-axis ranges from 100 to 600. A vertical red line is drawn at the year 1949. The data points are shown in purple, and a dark green line represents the fitted values.

The NPTEL logo is visible in the top right corner of the RStudio window.

Now from that I can compute the fitted values.

(Refer Slide Time: 07:06)



The screenshot displays the RStudio interface. The console window shows the following R code and its output:

```
26 fit_trend1 <- lm(y~scale(tm), data=data_train)
27 sum1 <- summary(fit_trend1)
28 round(coefficients(sum1), 3)
29 cat('Adjusted R.Squared = ', round(sum1$adj.r.squared, 3), '\n')
30
31
32 fit_hat <- fit_trend1$fitted.values
33 fit_hat <- ts(fit_hat, frequency = 12, start = 1949)
34
35 lines(fit_hat, col = "darkgreen", lwd = 2)
36
37 resid <- ts(fit_trend1$residuals, frequency = 12, start = 1949)
```



The console output shows the following table:

79	80	81	82	83	84
284.9171	287.2518	289.5865	291.9212	294.2559	296.5906
85	86	87	88	89	90
298.9253	301.2600	303.5948	305.9295	308.2642	310.5989
91	92	93	94	95	96
312.9336	315.2683	317.6030	319.9377	322.2725	324.6072

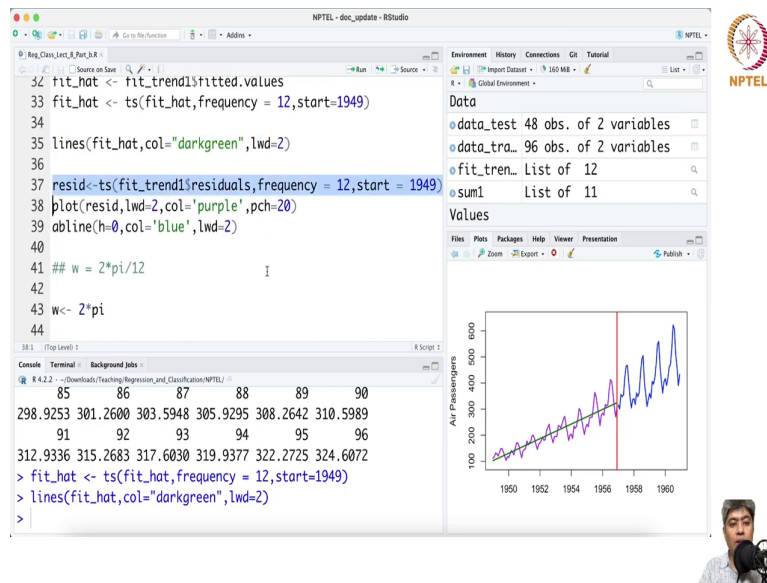
The Environment pane on the right shows the following data objects:

- data_test: 48 obs. of 2 variables
- data_tra: 96 obs. of 2 variables
- fit_tren: List of 12
- sum1: List of 11

The Plots pane shows a time series plot titled "Air Passengers". The y-axis is labeled "Air Passengers" and ranges from 100 to 600. The x-axis shows years from 1950 to 1960. The plot displays a blue line representing the fitted values and a purple line representing the residuals. A vertical red line is drawn at the year 1949.



(Refer Slide Time: 07:09)



The screenshot displays the RStudio interface. The script editor contains the following R code:

```
32 fit_hat <- fit_trend$fitted.values
33 fit_hat <- ts(fit_hat,frequency = 12,start=1949)
34
35 lines(fit_hat,col="darkgreen",lwd=2)
36
37 resid<-ts(fit_trend$residuals,frequency = 12,start = 1949)
38 plot(resid,lwd=2,col='purple',pch=20)
39 abline(h=0,col='blue',lwd=2)
40
41 ## w = 2*pi/12
42
43 w<- 2*pi
44
```

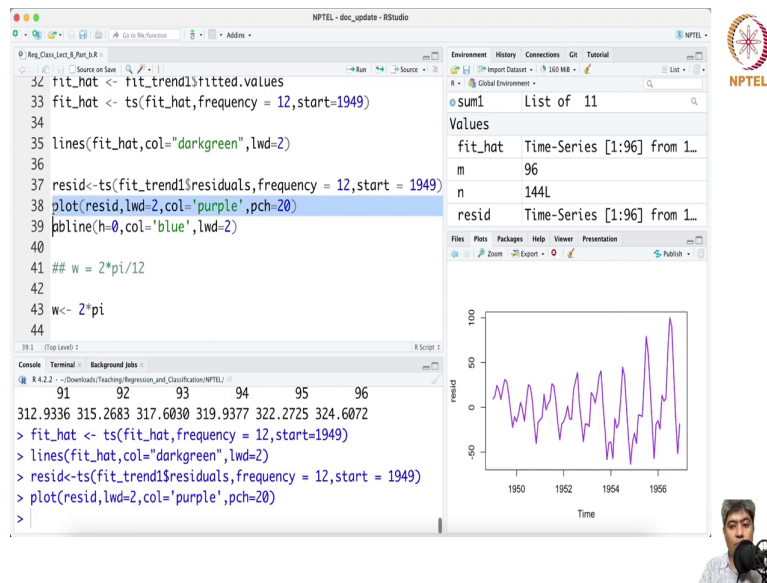
The console shows the output of the `lines` function, displaying a grid of values for years 85 through 96. The plot window shows a time series plot of 'Air Passengers' from 1950 to 1960. The y-axis ranges from 100 to 600. The plot features a purple line with circular markers representing the residuals, a dark green line representing the fitted trend, and a horizontal blue line at zero. A vertical red line is drawn at the year 1949.

Environment: R 4.2.2, Global Environment. Data: data_test (48 obs. of 2 variables), data_tra_ (96 obs. of 2 variables), fit_tren_ (List of 12), sum1 (List of 11).

NPTEL logo is visible in the top right corner.

And I just define it as a time series and plot the line. So, this is the fitted values and from there we from the fit trend.

(Refer Slide Time: 07:24)



The screenshot displays the RStudio interface. The script editor on the left contains the following R code:

```
32 fit_hat <- fit_trend1$fitted.values
33 fit_hat <- ts(fit_hat,frequency = 12,start=1949)
34
35 lines(fit_hat,col="darkgreen",lwd=2)
36
37 resid<-ts(fit_trend1$residuals,frequency = 12,start = 1949)
38 plot(resid,lwd=2,col='purple',pch=20)
39 pblime(h=0,col='blue',lwd=2)
40
41 ## w = 2*pi/12
42
43 w<- 2*pi
44
```

The console on the bottom left shows the execution of the code:

```
312.9336 315.2683 317.6030 319.9377 322.2725 324.6072
> fit_hat <- ts(fit_hat,frequency = 12,start=1949)
> lines(fit_hat,col="darkgreen",lwd=2)
> resid<-ts(fit_trend1$residuals,frequency = 12,start = 1949)
> plot(resid,lwd=2,col='purple',pch=20)
>
```

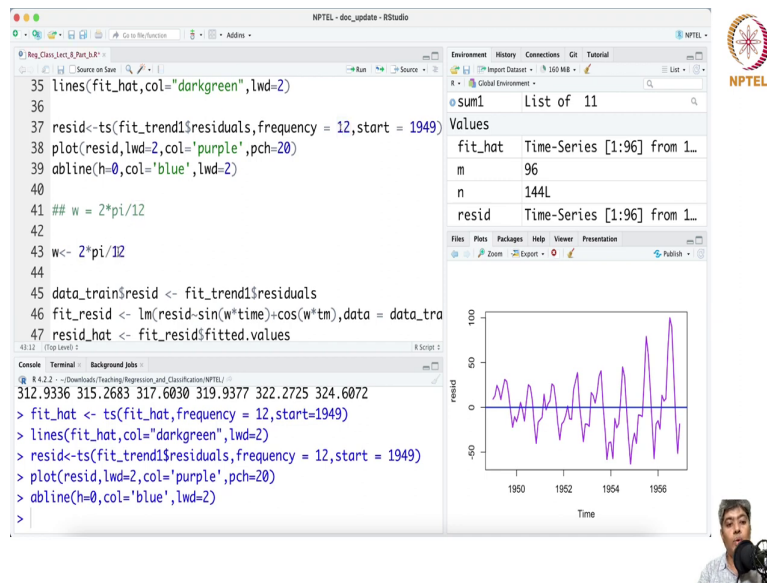
The Environment pane on the right shows a list of objects:

Object	Class
sum1	List of 11
fit_hat	Time-Series [1:96] from 1949 to 1966
m	96
n	144L
resid	Time-Series [1:96] from 1949 to 1966

The Plots pane on the right shows a time series plot of the residuals. The x-axis is labeled 'Time' and ranges from 1950 to 1956. The y-axis is labeled 'resid' and ranges from -50 to 100. The plot shows a highly volatile time series with a purple line and purple dots representing the residuals.

We just extract the residuals as frequencies as time series data and plot them and this is these are the residuals you can see this is the residuals.

(Refer Slide Time: 07:37)



The screenshot displays the RStudio interface. The script editor on the left contains the following R code:

```
35 lines(fit_hat,col="darkgreen",lwd=2)
36
37 resid<-ts(fit_trend$residuals,frequency = 12,start = 1949)
38 plot(resid,lwd=2,col='purple',pch=20)
39 abline(h=0,col='blue',lwd=2)
40
41 ## w = 2*pi/12
42
43 w<- 2*pi/12
44
45 data_train$resid <- fit_trend$residuals
46 fit_resid <- lm(resid-sin(w*time)+cos(w*tm),data = data_tra
47 resid_hat <- fit_resid$fitted.values
```

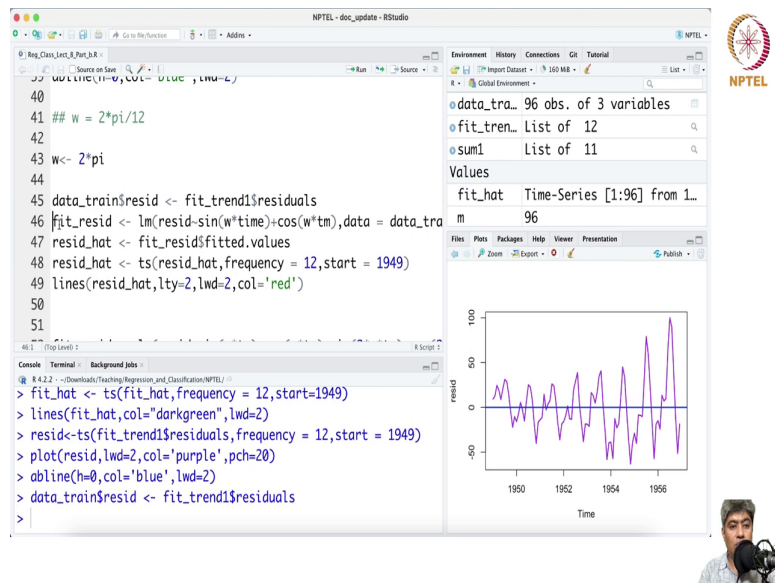
The console on the left shows the execution of the following commands:

```
> fit_hat <- ts(fit_hat,frequency = 12,start=1949)
> lines(fit_hat,col="darkgreen",lwd=2)
> resid<-ts(fit_trend$residuals,frequency = 12,start = 1949)
> plot(resid,lwd=2,col='purple',pch=20)
> abline(h=0,col='blue',lwd=2)
>
```

The Environment pane on the right shows a list of objects: `sum1` (List of 11), `fit_hat` (Time-Series [1:96] from 1...), `m` (96), `n` (144L), and `resid` (Time-Series [1:96] from 1...). Below the Environment pane is a plot of the residuals. The x-axis is labeled 'Time' and ranges from 1950 to 1956. The y-axis is labeled 'resid' and ranges from -50 to 100. The plot shows a purple line representing the residuals, which exhibits a clear seasonal pattern. A horizontal blue line is drawn at `resid = 0`. The NPTEL logo is visible in the top right corner of the RStudio window.

And at we plot a 0 at a at 0 the sum will be negative sum will be positive and we can see that residuals have the seasonality. Now we now we plot $2\pi\omega$ by 2π now we do not need to define it as a 12 because in the we have defined it as a time series data and here frequency is already defined as 12. So, that is why I do not need to divide it by 12.

(Refer Slide Time: 08:12)



The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for fitting a model and plotting residuals. The code includes:

```
40  
41 ## w = 2*pi/12  
42  
43 w<- 2*pi  
44  
45 data_train$resid <- fit_trend1$residuals  
46 fit_resid <- lm(resid~sin(w*time)+cos(w*tm),data = data_tra  
47 resid_hat <- fit_resid$fitted.values  
48 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)  
49 lines(resid_hat,lty=2,lwd=2,col='red')  
50  
51
```
- Environment:** Lists objects: `data_tra_` (96 obs. of 3 variables), `fit_tren_` (List of 12), and `sum1` (List of 11).
- Console:** Shows the execution of the following commands:

```
> fit_hat <- ts(fit_hat,frequency = 12,start=1949)  
> lines(fit_hat,col="darkgreen",lwd=2)  
> resid<-ts(fit_trend1$residuals,frequency = 12,start = 1949)  
> plot(resid,lwd=2,col='purple',pch=20)  
> abline(h=0,col='blue',lwd=2)  
> data_train$resid <- fit_trend1$residuals  
>
```
- Plots:** A time-series plot of residuals. The x-axis is labeled 'Time' with ticks at 1950, 1952, 1954, and 1956. The y-axis is labeled 'resid' with ticks at -50, 0, 50, and 100. The plot shows a purple line with circular markers oscillating around a horizontal blue line at y=0.
- NPTEL Logo:** Located in the top right corner of the RStudio window.
- Speaker:** A small video feed of a person speaking is visible in the bottom right corner.

So, because data it is already being defined as a frequency of 12.

(Refer Slide Time: 08:25)

The screenshot displays the RStudio interface with the following components:

- Environment:** Shows objects `data_train` (96 obs. of 3 variables), `fit_trend` (List of 12), and `sum1` (List of 11).
- Values:** Shows `fit_hat` as a Time-Series [1:96] from 1949 to 1956, with a mean value of 96.
- Console:** Contains the following R code:

```
> lines(fit_hat,col="darkgreen",lwd=2)
> resid<-ts(fit_trend$residuals,frequency = 12,start = 1949)
> plot(resid,lwd=2,col='purple',pch=20)
> abline(h=0,col='blue',lwd=2)
> data_train$resid <- fit_trend$residuals
> View(data_train)
>
```
- Plot:** A time-series plot of residuals. The x-axis is labeled 'Time' with major ticks at 1950, 1952, 1954, and 1956. The y-axis is labeled 'resid' with ticks at -50, 0, 50, and 100. The plot shows a purple line with circular markers fluctuating around a horizontal blue line at y=0.

The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 08:31)

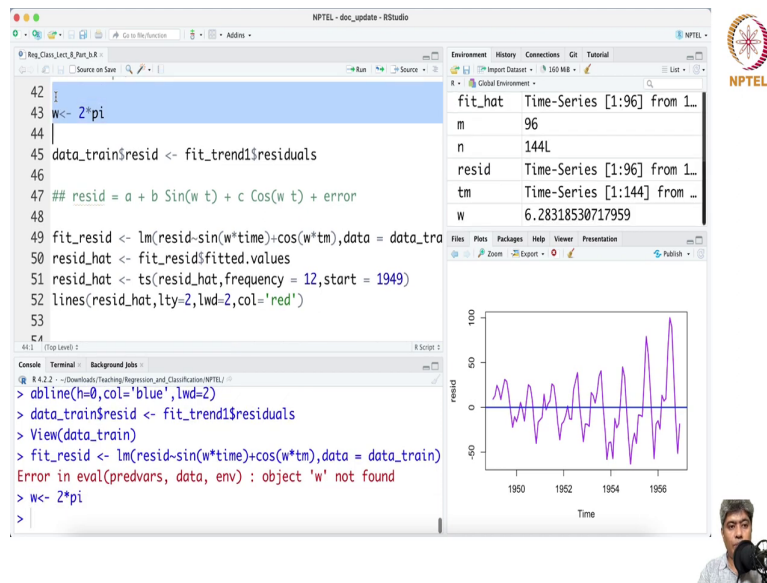
The screenshot displays the RStudio interface. The top-left pane shows a data table with columns 'tm' and 'resid'. The console pane contains the following R code:

```
R 4.2.2 - ~/Downloads/Teaching-Regression_and_Classification/NPTEL/ > lines(fit_hat,col="darkgreen",lwd=2) > resid<-ts(fit_trend1$residuals,frequency = 12,start = 1949) > plot(resid,lwd=2,col='purple',pch=20) > abline(h=0,col='blue',lwd=2) > data_train$resid <- fit_trend1$residuals > View(data_train) >
```

The top-right pane shows the Environment window with objects: data_tra_ (96 obs. of 3 variables), fit_tren_ (List of 12), and sum1 (List of 11). The bottom-right pane shows a plot of residuals over time, with a purple line and a blue horizontal line at zero. The x-axis is labeled 'Time' and ranges from 1950 to 1956. The y-axis is labeled 'resid' and ranges from -50 to 100. A small inset video of the presenter is visible in the bottom right corner.

So, data set is residuals and now in the data set train you see I have extracted the residuals ok extracted the residuals and fit the residuals as a function of $\sin \omega t$ plus $\cos \omega t$ time ok.

(Refer Slide Time: 08:45)



The screenshot shows the RStudio interface. The script editor contains the following R code:

```
42 |  
43 w<- 2*pi  
44 |  
45 data_train$resid <- fit_trend1$residuals  
46 |  
47 ## resid = a + b Sin(w t) + c Cos(w t) + error  
48 |  
49 fit_resid <- lm(resid-sin(w*time)+cos(w*tm),data = data_train)  
50 resid_hat <- fit_resid$fitted.values  
51 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)  
52 lines(resid_hat,lty=2,lwd=2,col='red')  
53 |
```

The console shows the execution of the code, including the error message: "Error in eval(predvars, data, env) : object 'w' not found".

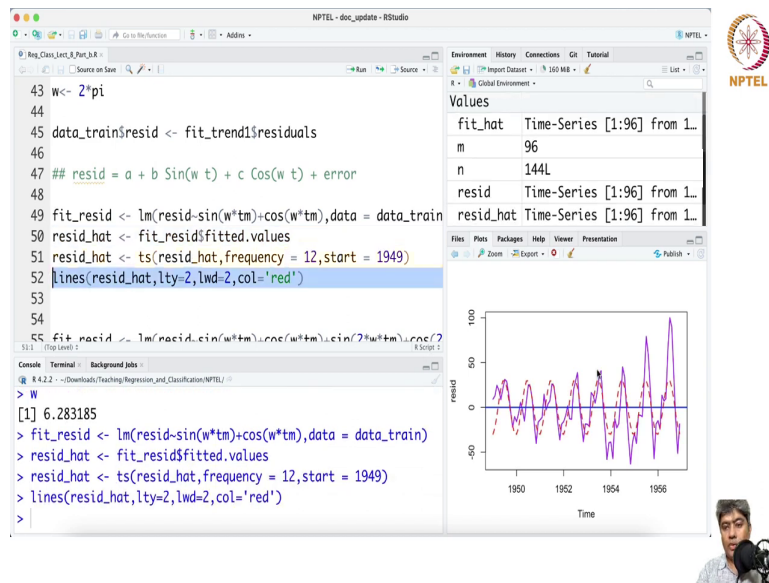
The Environment pane shows the following objects:

fit_hat	Time-Series [1:96] from 1...
m	96
n	144L
resid	Time-Series [1:96] from 1...
tm	Time-Series [1:144] from ...
w	6.28318530717959

The Plot pane shows a time series plot of the residuals. The x-axis is labeled "Time" and ranges from 1950 to 1956. The y-axis is labeled "resid" and ranges from -50 to 100. A red line represents the residuals, and a blue horizontal line is drawn at y=0.

Now, this is the model what we want to fit residual as a function of a plus b times Sin omega t plus c times cos omega t plus error this is the model we want to fit. So, we fit this model sorry we need to run this guy yeah.

(Refer Slide Time: 09:19)



The screenshot displays the RStudio interface with the following R code in the editor:

```
43 w<- 2*pi
44 data_train$resid <- fit_trend1$residuals
45
46 ## resid = a + b Sin(w t) + c Cos(w t) + error
47
48
49 fit_resid <- lm(resid~sin(w*tm)+cos(w*tm),data = data_train)
50 resid_hat <- fit_resid$fitted.values
51 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
52 lines(resid_hat,lty=2,lwd=2,col='red')
53
54
55 fit_resid <- lm(resid~sin(w*tm)+cos(w*tm)+sin(2*w*tm)+cos(2*w*tm),data = data_train)
56 resid_hat <- fit_resid$fitted.values
57 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
58 lines(resid_hat,lty=2,lwd=2,col='red')
```



The console shows the execution of the code:

```
> w
[1] 6.283185
> fit_resid <- lm(resid~sin(w*tm)+cos(w*tm),data = data_train)
> resid_hat <- fit_resid$fitted.values
> resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
> lines(resid_hat,lty=2,lwd=2,col='red')
```

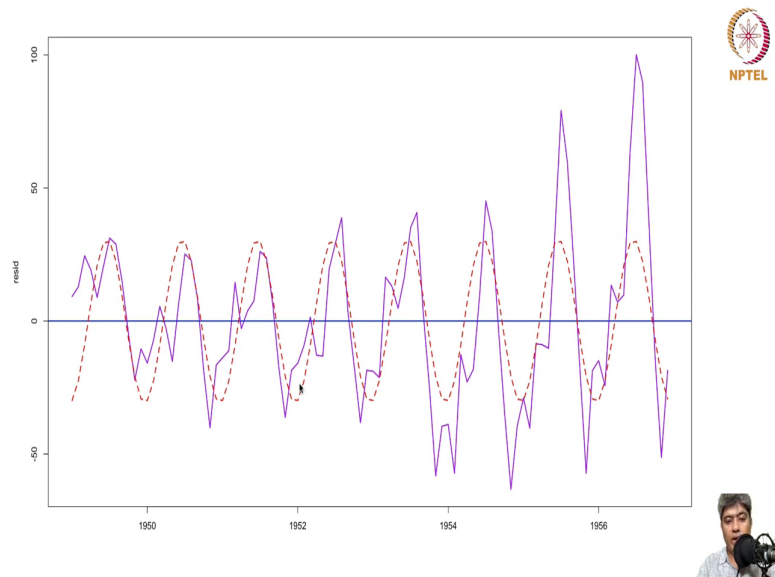
The Environment pane shows the following variables:

Variable	Value
fit_hat	Time-Series [1:96] from 1...
m	96
n	144L
resid	Time-Series [1:96] from 1...
resid_hat	Time-Series [1:96] from 1...

The Plots pane shows a time series plot of the residuals. The x-axis is labeled 'Time' and ranges from 1950 to 1956. The y-axis is labeled 'resid' and ranges from -50 to 100. The plot shows a highly oscillatory time series with a red dashed line representing the fitted model.

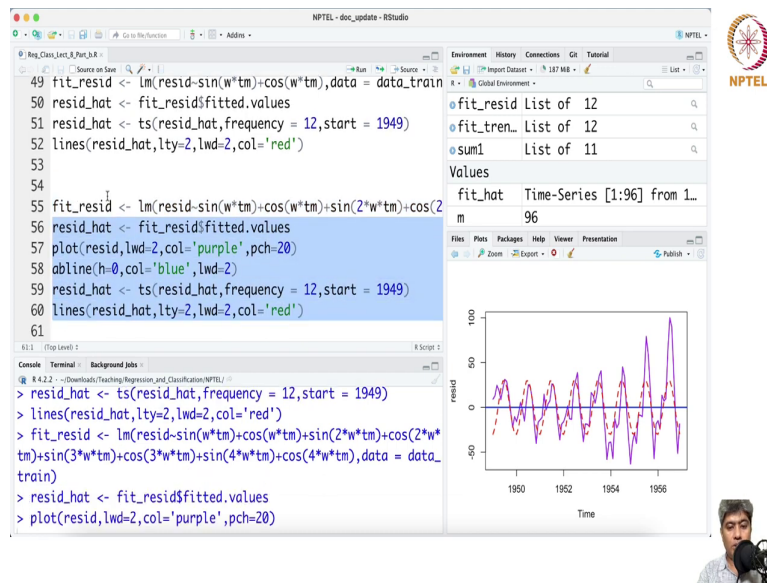


(Refer Slide Time: 09:27)



Now you can see that let me just enlarge this thing. So, now, you can see that this sin cosine is trying to capture the seasonality annual seasonality of the data, but it misses the like you know highs and lows of the data.

(Refer Slide Time: 09:51)



The screenshot displays the RStudio interface with the following R code in the script editor:

```
49 fit_resid <- lm(resid-sin(w*tm)+cos(w*tm),data = data_train)
50 resid_hat <- fit_resid$fitted.values
51 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
52 lines(resid_hat,lty=2,lwd=2,col='red')
53
54
55 fit_resid <- lm(resid-sin(w*tm)+cos(w*tm)+sin(2*w*tm)+cos(2*
56 resid_hat <- fit_resid$fitted.values
57 plot(resid,lwd=2,col='purple',pch=20)
58 abline(h=0,col='blue',lwd=2)
59 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
60 lines(resid_hat,lty=2,lwd=2,col='red')
61
```

The console shows the execution of the following commands:

```
> resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
> lines(resid_hat,lty=2,lwd=2,col='red')
> fit_resid <- lm(resid-sin(w*tm)+cos(w*tm)+sin(2*w*tm)+cos(2*w*
tm)+sin(3*w*tm)+cos(3*w*tm)+sin(4*w*tm)+cos(4*w*tm),data = data_
train)
> resid_hat <- fit_resid$fitted.values
> plot(resid,lwd=2,col='purple',pch=20)
```

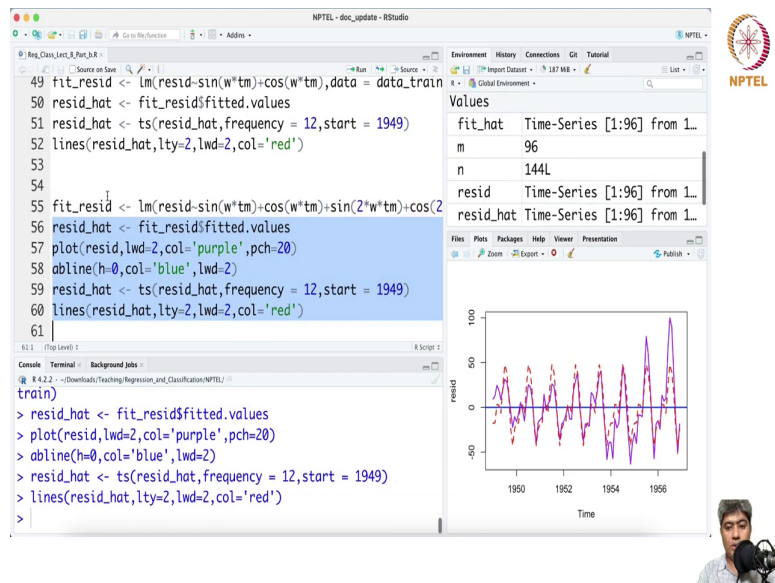
The environment pane on the right shows the following objects:

- fit_resid List of 12
- fit_tren_ List of 12
- sum1 List of 11

The plot pane displays a time-series plot of residuals from 1949 to 1956. The y-axis is labeled 'resid' and ranges from -50 to 100. The x-axis is labeled 'Time' and shows years from 1950 to 1956. The plot features a purple line with circular markers representing the residuals, a solid red line representing the fitted values, and a horizontal blue line at zero.

Now, it for then we fit a second order residual Fourier transform.

(Refer Slide Time: 09:57)



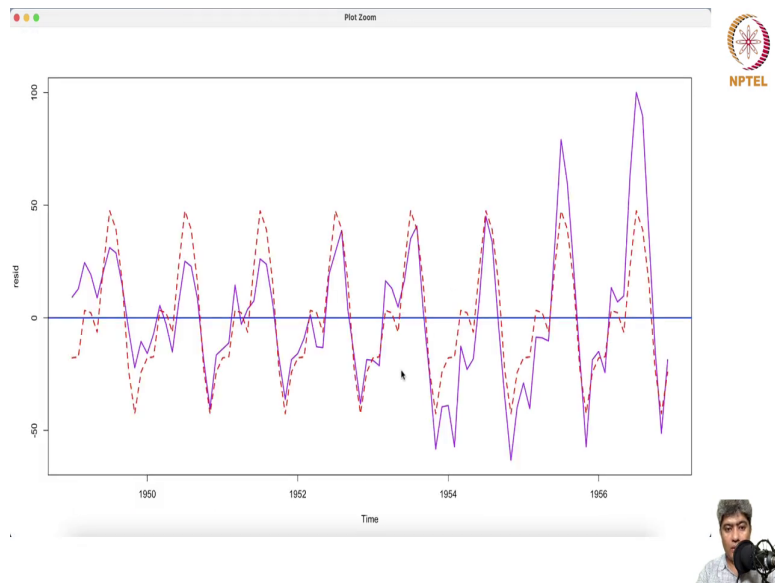
The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for fitting a model and plotting residuals. The code is as follows:

```
49 fit_resid <- lm(resid~sin(w*tm)+cos(w*tm),data = data_train)
50 resid_hat <- fit_resid$fitted.values
51 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
52 lines(resid_hat,lty=2,lwd=2,col='red')
53
54
55 fit_resid <- lm(resid~sin(w*tm)+cos(w*tm)+sin(2*w*tm)+cos(2
56 resid_hat <- fit_resid$fitted.values
57 plot(resid,lwd=2,col='purple',pch=20)
58 abline(h=0,col='blue',lwd=2)
59 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
60 lines(resid_hat,lty=2,lwd=2,col='red')
61
```
- Environment:** Shows variables `fit_hat` (Time-Series [1:96] from 1...), `m` (96), `n` (144L), `resid` (Time-Series [1:96] from 1...), and `resid_hat` (Time-Series [1:96] from 1...).
- Console:** Shows the execution of the code, including the `train` function and the `plot` command.

```
> resid_hat <- fit_resid$fitted.values
> plot(resid,lwd=2,col='purple',pch=20)
> abline(h=0,col='blue',lwd=2)
> resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
> lines(resid_hat,lty=2,lwd=2,col='red')
>
```
- Plots:** A time-series plot of the residuals is displayed, showing a purple line with circular markers and a red line representing the fitted model. The x-axis is labeled 'Time' and ranges from 1950 to 1956. The y-axis is labeled 'resid' and ranges from -50 to 100. A horizontal blue line is drawn at `y=0`.

(Refer Slide Time: 09:59)



And we plot this and now we can see that this red curve is actually trying to capture the this local seasonality this local seasonality is trying to be captured by the you know the second term Fourier.

(Refer Slide Time: 10:17)

The screenshot shows an RStudio interface with the following components:

- Source Editor:** Contains R code for plotting residuals and fitting a trend and seasonality model. The code includes:

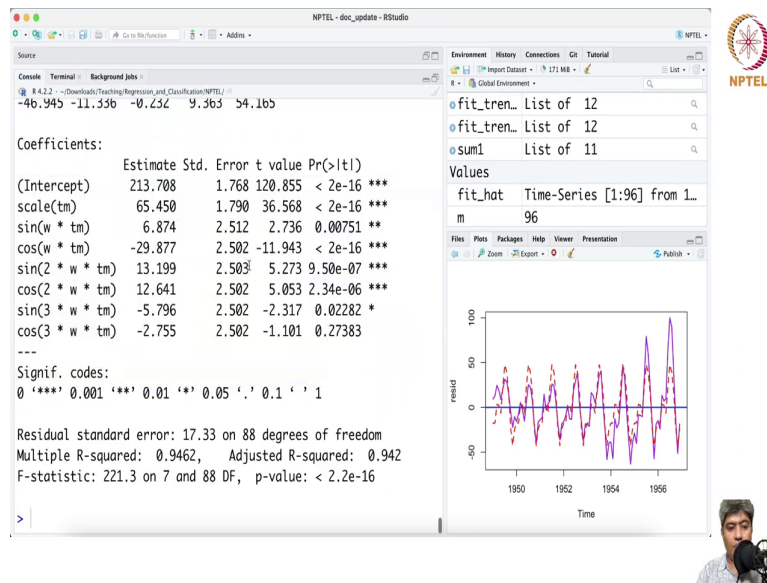
```
58 abline(h=0,col='blue',lwd=2)
59 resid_hat <- ts(resid_hat,frequency = 12,start = 1949)
60 lines(resid_hat,lty=2,lwd=2,col='red')
61
62 I
63 fit_trend_season<-lm(y-scale(tm)+sin(w*tm)+cos(w*tm)
64                   +sin(2*w*tm)+cos(2*w*tm)
65                   +sin(3*w*tm)+cos(3*w*tm)
66                   ,data=data_train)
67 summary(fit_trend_season)
68
69 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Ai
70 noints(data_train$tm, data_train$y, type='l', lwd=2, col='purple')
```
- Console:** Displays the output of the `summary` function:

```
0 **** 0.001 *** 0.01 ** 0.05 . 0.1 ' ' 1

Residual standard error: 17.33 on 88 degrees of freedom
Multiple R-squared: 0.9462, Adjusted R-squared: 0.942
F-statistic: 221.3 on 7 and 88 DF, p-value: < 2.2e-16
```
- Environment:** Lists objects: `fit_trend_season` (lm), `fit_hat` (Time-Series [1:96] from 1...), and `m` (96).
- Plot:** A time-series plot showing residuals (y-axis, -50 to 100) against time (x-axis, 1950 to 1956). The plot features a blue horizontal line at zero, a purple line representing the residuals, and a red line representing the fitted model. The residuals show a clear periodic pattern.

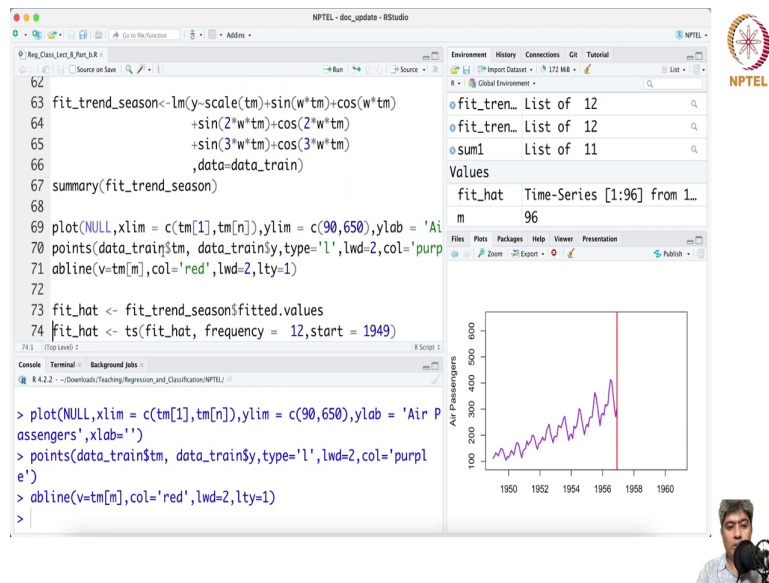
So, now if we fit a trend and seasonality together what we are getting is.

(Refer Slide Time: 10:31)



And then almost so, that you know the time that you know the sin cosine everything is kind of extremely significant and adjusted R square is 0.942. So, 94 percent of the x variability get explained by the model.

(Refer Slide Time: 10:52)



The screenshot displays the RStudio interface with the following R code in the editor:

```
62 fit_trend_season <- lm(y ~ scale(tm) + sin(w*tm) + cos(w*tm)
63                       + sin(2*w*tm) + cos(2*w*tm)
64                       + sin(3*w*tm) + cos(3*w*tm)
65                       , data = data_train)
66 summary(fit_trend_season)
67
68
69 plot(NULL, xlim = c(tm[1], tm[n]), ylim = c(90, 650), ylab = 'Air P
70 points(data_train$tm, data_train$y, type = 'l', lwd = 2, col = 'purpl
71 abline(v = tm[m], col = 'red', lwd = 2, lty = 1)
72
73 fit_hat <- fit_trend_season$fitted.values
74 fit_hat <- ts(fit_hat, frequency = 12, start = 1949)
```



The console shows the execution of the plotting commands:

```
> plot(NULL, xlim = c(tm[1], tm[n]), ylim = c(90, 650), ylab = 'Air P
> assengers', xlab = '')
> points(data_train$tm, data_train$y, type = 'l', lwd = 2, col = 'purpl
> e')
> abline(v = tm[m], col = 'red', lwd = 2, lty = 1)
>
```

The Environment pane on the right shows the following objects:

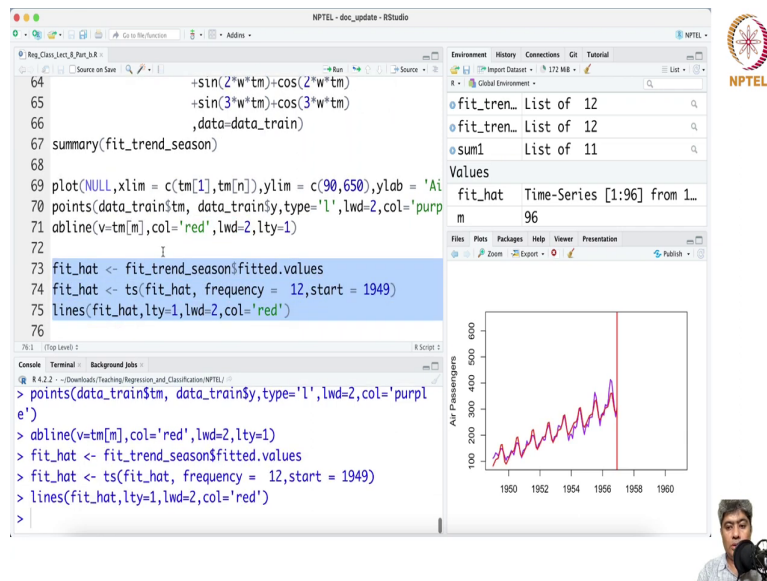
- fit_tren... List of 12
- fit_tren... List of 12
- sum1 List of 11
- Values
- fit_hat Time-Series [1:96] from 1_
- m 96

The Plots pane shows a time series plot of 'Air Passengers' from 1950 to 1960. The y-axis ranges from 100 to 600. The plot shows a purple line representing the data points and a red vertical line at approximately 1957. The plot is titled 'Air Passengers'.



And then we plot the model and then fit hat.

(Refer Slide Time: 10:59)



The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for fitting a seasonal model. The code includes:

```
64 +sin(2*w*tm)+cos(2*w*tm)
65 +sin(3*w*tm)+cos(3*w*tm)
66 ,data=data_train)
67 summary(fit_trend_season)
68
69 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air
70 points(data_train$tm, data_train$y,type='l',lwd=2,col='purple'
71 abline(v=tm[m],col='red',lwd=2,ty=1)
72
73 fit_hat <- fit_trend_season$fitted.values
74 fit_hat <- ts(fit_hat, frequency = 12,start = 1949)
75 lines(fit_hat,ty=1,lwd=2,col='red')
76
```
- Environment:** Lists objects: fit_tren... List of 12, fit_tren... List of 12, and sum1 List of 11.
- Values:** Shows fit_hat as a Time-Series [1:96] from 1949 to 1960 with a value of 96.
- Console:** Shows the execution of the code:

```
> points(data_train$tm, data_train$y,type='l',lwd=2,col='purple'
e')
> abline(v=tm[m],col='red',lwd=2,ty=1)
> fit_hat <- fit_trend_season$fitted.values
> fit_hat <- ts(fit_hat, frequency = 12,start = 1949)
> lines(fit_hat,ty=1,lwd=2,col='red')
>
```
- Plots:** A line plot titled 'Air Passengers' showing data from 1950 to 1960. The y-axis ranges from 100 to 600. The plot shows a clear upward trend with seasonal fluctuations. A vertical red line is drawn at the year 1957.

So, we can in the in sample the fitting is quite good.

(Refer Slide Time: 11:04)

The screenshot displays the RStudio interface with the following components:

- Script Editor:** Contains R code for fitting a trend and season model to historical data and predicting out-of-sample values.
- Environment:** Shows the objects created in the workspace, including a list of 11 values.
- Console:** Shows the execution of the R code, including the plotting commands.
- Plot:** A time series plot titled "Air Passengers" showing data from 1950 to 1960. The y-axis ranges from 100 to 600. A vertical red line is drawn at approximately 1957, separating the in-sample data (purple points and red line) from the out-of-sample prediction (blue points and blue line).

```
70 points(data_train$tm, data_train$y, type='l', lwd=2, col='purple')
71 abline(v=tm[m], col='red', lwd=2, lty=1)
72
73 fit_hat <- fit_trend_season$fitted.values
74 fit_hat <- ts(fit_hat, frequency = 12, start = 1949)
75 lines(fit_hat, lty=1, lwd=2, col='red')
76
77 pred <- predict(fit_trend_season, newdata = data_test)
78 pred <- ts(pred, frequency = 12, start = tm[(m+1)])
79 lines(pred, lty=1, lwd=2, col='green')
80 points(data_test$tm, data_test$y, type='l', lwd=2, col='blue')
81
82
```

Environment: Global Environment

sum1	List of 11
fit_hat	Time-Series [1:96] from 1...
m	96
n	144L
pred	Named num [1:48] 308 320 ...

Console:

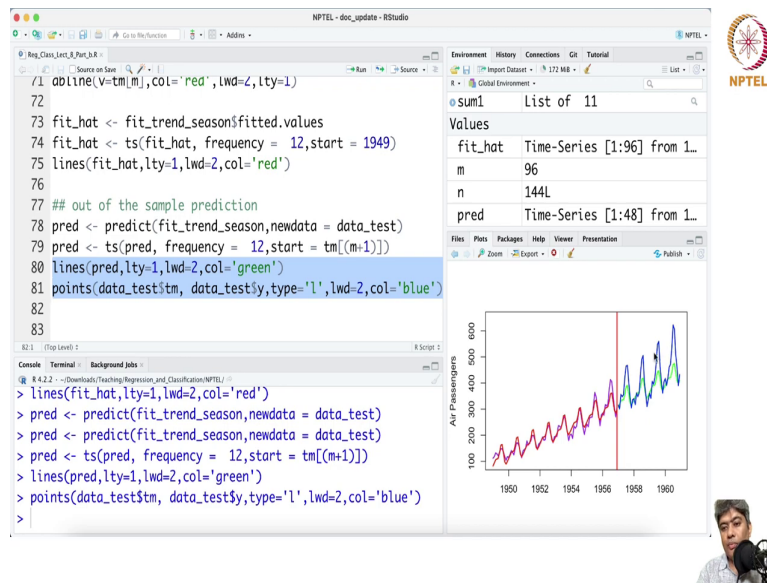
```
> abline(v=tm[m], col='red', lwd=2, lty=1)
> fit_hat <- fit_trend_season$fitted.values
> fit_hat <- ts(fit_hat, frequency = 12, start = 1949)
> lines(fit_hat, lty=1, lwd=2, col='red')
> pred <- predict(fit_trend_season, newdata = data_test)
>
```

Plot: Air Passengers

Year	Air Passengers
1950	100
1951	150
1952	200
1953	250
1954	300
1955	350
1956	400
1957	450
1958	500
1959	550
1960	600

And let us do the out of the sample prediction this is out of the sample prediction: out of the sample prediction.

(Refer Slide Time: 11:10)



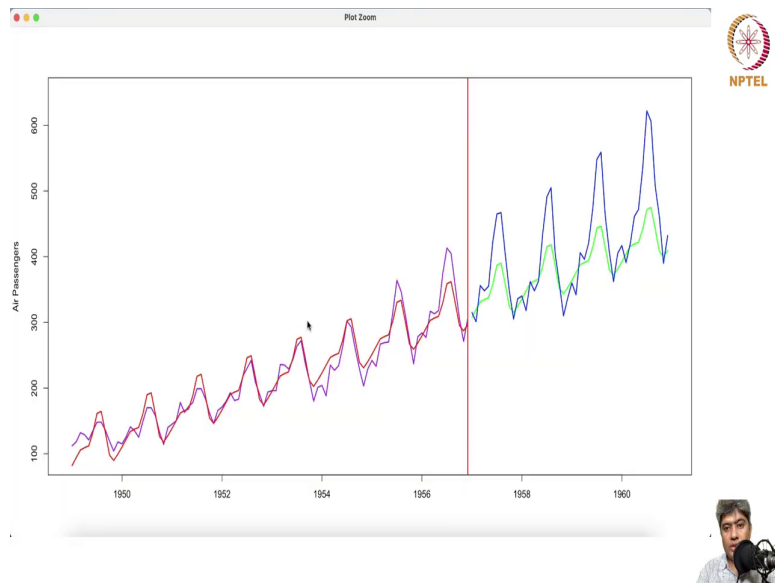
The screenshot displays an RStudio interface with the following components:

- Script Editor:** Contains R code for fitting a trend-season model and making predictions. The code includes:

```
1 abline(v=tm[m],col='red',lwd=2, lty=1)
72
73 fit_hat <- fit_trend_season$fitted.values
74 fit_hat <- ts(fit_hat, frequency = 12, start = 1949)
75 lines(fit_hat, lty=1, lwd=2, col='red')
76
77 ## out of the sample prediction
78 pred <- predict(fit_trend_season, newdata = data_test)
79 pred <- ts(pred, frequency = 12, start = tm[(m+1)])
80 lines(pred, lty=1, lwd=2, col='green')
81 points(data_test$tm, data_test$y, type='l', lwd=2, col='blue')
82
83
```
- Environment:** Shows a list of 11 objects, including 'fit_hat' (Time-Series [1:96] from 1949) and 'pred' (Time-Series [1:48] from 1957).
- Console:** Displays the execution output of the code, showing the creation of the fitted model and the prediction series.
- Plot:** A time series plot titled 'Air Passengers' showing the number of passengers from 1949 to 1960. The y-axis ranges from 100 to 600. A vertical red line is drawn at the end of the training data (1956). The fitted model is shown as a red line, the prediction as a green line, and the test data as blue points.

So, we do that and then we can see that you know its definitely missing the if you just let me just.

(Refer Slide Time: 11:36)



So, this reds and this greens are out of the sample forecast and clearly it is missing the highs of the you know max the peak seasons.

(Refer Slide Time: 11:53)

The screenshot displays the RStudio interface. The editor window contains the following R code:

```
84 ## take log-transformation on Y and run step-wise variable
85 ## selection to choose best engineered feature to capture t
86 ## seasonality
87 |
88 |
89 fit_trend_season_transform<-step(lm(log(y)-tm+I(tm^2)
90 +sin(w*tm)+cos(w*tm)
91 +sin(2*w*tm)+cos(2*w*tm)
92 +sin(3*w*tm)+cos(3*w*tm)
93 +sin(4*w*tm)+cos(4*w*tm)
94 +sin(5*w*tm)+cos(5*w*tm)
95 +sin(6*w*tm)+cos(6*w*tm)
```

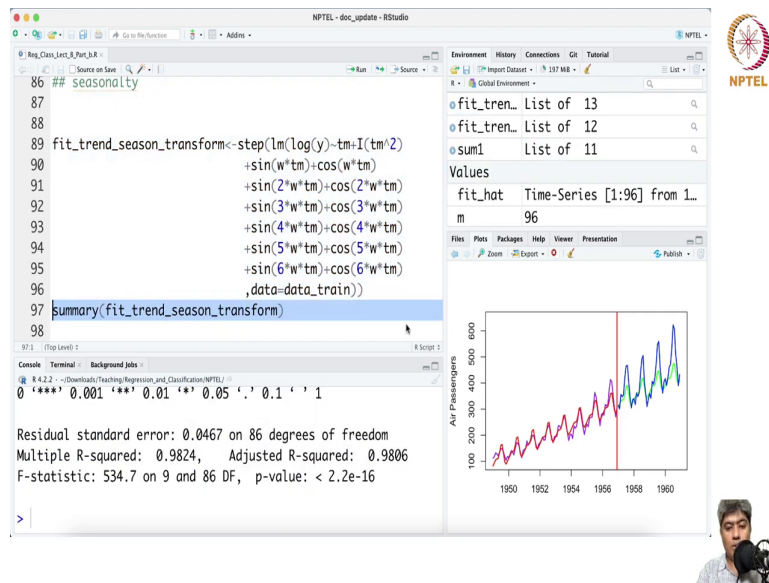
The console window shows the following commands and their output:

```
> lines(fit_hat, lty=1, lwd=2, col='red')
> pred <- predict(fit_trend_season, newdata = data_test)
> pred <- predict(fit_trend_season, newdata = data_test)
> pred <- ts(pred, frequency = 12, start = tm[(m+1)])
> lines(pred, lty=1, lwd=2, col='green')
> points(data_test$tm, data_test$y, type='l', lwd=2, col='blue')
>
```

The environment pane shows a list of 11 values for the fit_hat object, including m (96) and n (144L). The plot pane shows a time series plot of Air Passengers from 1950 to 1960, with a red line representing the fit and a green line representing the prediction. A vertical red line is drawn at approximately 1956.5.

So, we take the log transformation we trend and seasonality with log transformation and then we on that we run a step by selection transformation and run step wise variable selection to choose best engineered feature to capture the seasonality dt ok. So, let me just run this. So, you can see sin cos now all the cos terms of fourth and fifth order cos terms are being dropped ok.

(Refer Slide Time: 12:42)



The image displays the RStudio interface for a regression analysis. The main editor window shows the following R code:

```
86 ## seasonality
87
88
89 fit_trend_season_transform<-step(lm(log(y)~tm+I(tm^2)
90 +sin(w*tm)+cos(w*tm)
91 +sin(2*w*tm)+cos(2*w*tm)
92 +sin(3*w*tm)+cos(3*w*tm)
93 +sin(4*w*tm)+cos(4*w*tm)
94 +sin(5*w*tm)+cos(5*w*tm)
95 +sin(6*w*tm)+cos(6*w*tm)
96 ,data=data_train))
97 summary(fit_trend_season_transform)
98
```

The console window displays the following output:

```
97.1 (Top Level) >
R 4.2.2 - ...Downloads\Teaching\Regression_and_Classification\NPTEL\
0 **** 0.001 *** 0.01 ** 0.05 . 0.1 ' ' 1

Residual standard error: 0.0467 on 86 degrees of freedom
Multiple R-squared: 0.9824, Adjusted R-squared: 0.9806
F-statistic: 534.7 on 9 and 86 DF, p-value: < 2.2e-16
```

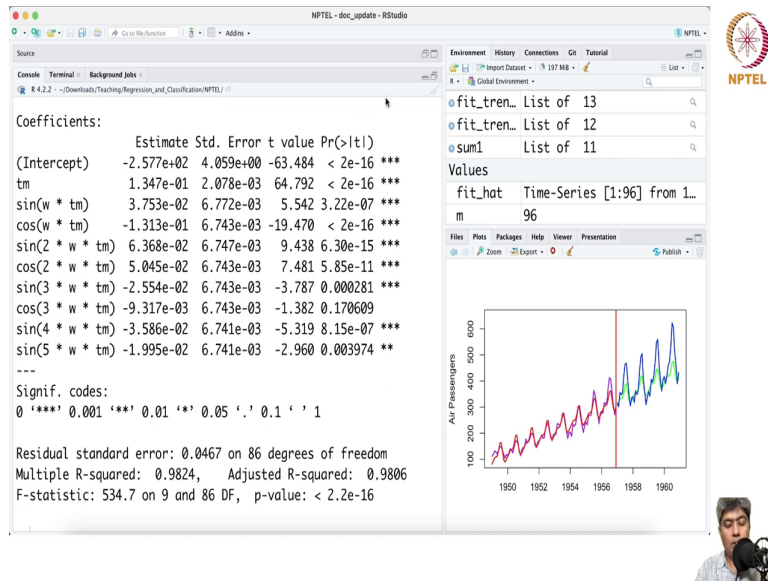
The Environment pane on the right shows the following objects:

- fit_tren_ List of 13
- fit_tren_ List of 12
- sum1 List of 11
- Values
- fit_hat Time-Series [1:96] from 1_
- m 96

The Plots pane shows a time-series plot of Air Passengers from 1950 to 1960. The y-axis is labeled 'Air Passengers' and ranges from 100 to 600. The x-axis is labeled with years from 1950 to 1960. The plot shows a clear upward trend with seasonal fluctuations. A vertical red line is drawn at approximately 1956.5. The data points are colored in a gradient from purple to blue.

The NPTEL logo is visible in the top right corner of the RStudio window.

(Refer Slide Time: 12:47)



The screenshot displays the RStudio interface with the following content:

Console:



```
Coefficients:
      Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.577e+02  4.059e+00 -63.484 < 2e-16 ***
tm           1.347e-01  2.078e-03  64.792 < 2e-16 ***
sin(w * tm)  3.753e-02  6.772e-03  5.542 3.22e-07 ***
cos(w * tm) -1.313e-01  6.743e-03 -19.470 < 2e-16 ***
sin(2 * w * tm) 6.368e-02  6.747e-03  9.438 6.30e-15 ***
cos(2 * w * tm) 5.045e-02  6.743e-03  7.481 5.85e-11 ***
sin(3 * w * tm) -2.554e-02  6.743e-03 -3.787 0.000281 ***
cos(3 * w * tm) -9.317e-03  6.743e-03 -1.382 0.170609
sin(4 * w * tm) -3.586e-02  6.741e-03 -5.319 8.15e-07 ***
sin(5 * w * tm) -1.995e-02  6.741e-03 -2.960 0.003974 **
---
Signif. codes:
  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0467 on 86 degrees of freedom
Multiple R-squared:  0.9824,    Adjusted R-squared:  0.9806
F-statistic: 534.7 on 9 and 86 DF,  p-value: < 2.2e-16
```

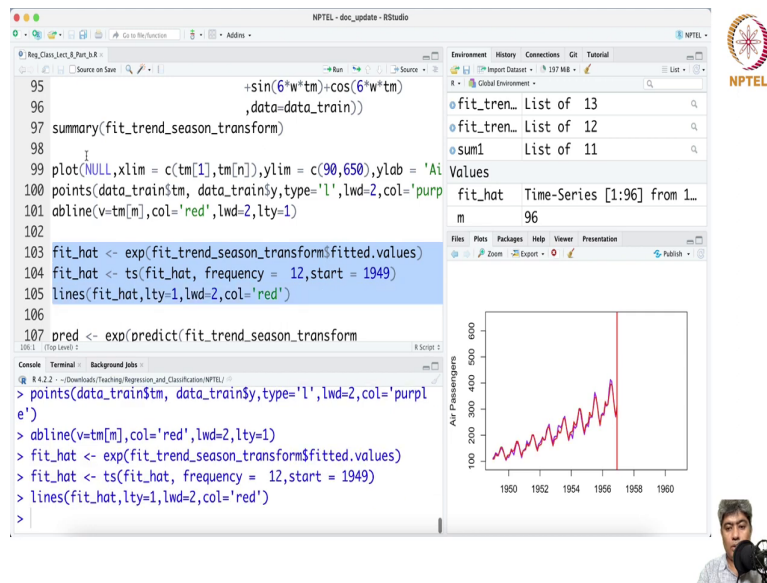
Environment: fit_tren_ List of 13, fit_tren_ List of 12, sum1 List of 11

Values: fit_hat Time-Series [1:96] from 1_ m 96

Plots: A time-series plot titled "Air Passengers" showing data from 1950 to 1960. The y-axis ranges from 100 to 600. The plot shows a clear upward trend with seasonal fluctuations. A vertical red line is drawn at approximately 1956.5. The plot includes a fitted model line and several data series in different colors (red, blue, green, purple).



(Refer Slide Time: 12:59)



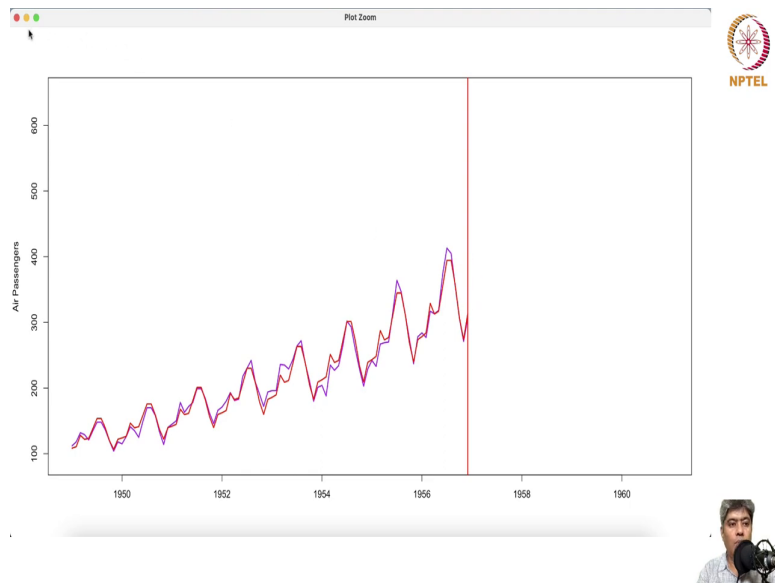
The screenshot shows an RStudio interface with the following components:

- Source Editor:** Contains R code for fitting a seasonal model to 'Air Passengers' data. The code includes:

```
95 +sin(6*w*tm)+cos(6*w*tm)
96 ,data=data_train))
97 summary(fit_trend_season_transform)
98
99 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air
100 passengers',data_train$tm, data_train$y,type='l',lwd=2,col='purple',
101 abline(v=tm[m],col='red',lwd=2,ty=1)
102
103 fit_hat <- exp(fit_trend_season_transform$fitted.values)
104 fit_hat <- ts(fit_hat, frequency = 12,start = 1949)
105 lines(fit_hat, lty=1,lwd=2,col='red')
106
107 pred <- exp(predict(fit_trend_season_transform
```
- Environment:** Shows objects like 'fit_trend_season_transform', 'fit_hat', and 'pred'.
- Console:** Displays the execution of the code from the source editor.
- Plots:** A time series plot titled 'Air Passengers' showing data from 1949 to 1960. The y-axis ranges from 100 to 600. A vertical red line is drawn at the year 1956.
- NPTEL Logo:** Located in the top right corner of the RStudio window.
- Speaker:** A small video feed of a person speaking is visible in the bottom right corner.

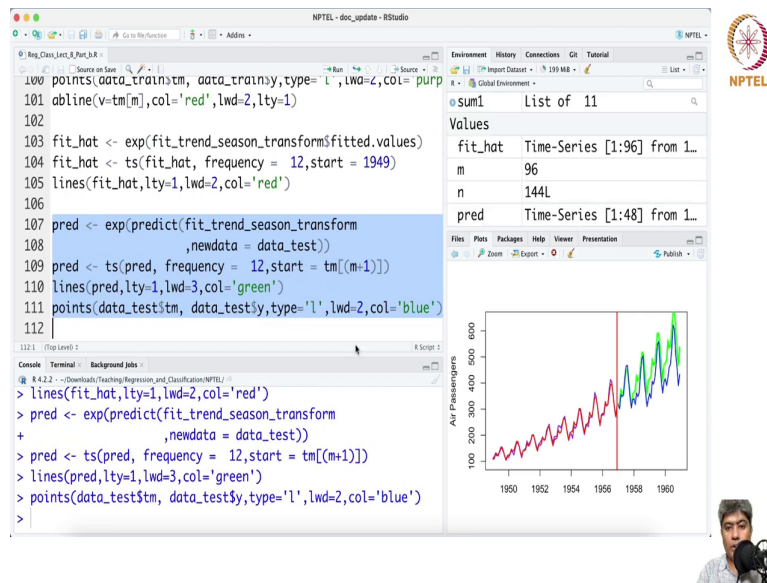
So, and sin 6 and cos 6 also being dropped. So, now, if we draw the plot.

(Refer Slide Time: 13:12)



So, this is in sample fitting which is pretty much picking up the entire model enter data nicely doing.

(Refer Slide Time: 13:18)



The screenshot displays the RStudio interface with the following code in the script editor:

```
100 points(data_train$tm, data_train$y,type='l',lwd=2,col='purple')
101 abline(v=tm[m],col='red',lwd=2,ty=1)
102
103 fit_hat <- exp(fit_trend_season_transform$fitted.values)
104 fit_hat <- ts(fit_hat, frequency = 12,start = 1949)
105 lines(fit_hat,ty=1,lwd=2,col='red')
106
107 pred <- exp(predict(fit_trend_season_transform
108 ,newdata = data_test))
109 pred <- ts(pred, frequency = 12,start = tm[(m+1)])
110 lines(pred,ty=1,lwd=3,col='green')
111 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
112
```

The console shows the execution of the following commands:

```
> lines(fit_hat,ty=1,lwd=2,col='red')
> pred <- exp(predict(fit_trend_season_transform
+ ,newdata = data_test))
> pred <- ts(pred, frequency = 12,start = tm[(m+1)])
> lines(pred,ty=1,lwd=3,col='green')
> points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
>
```

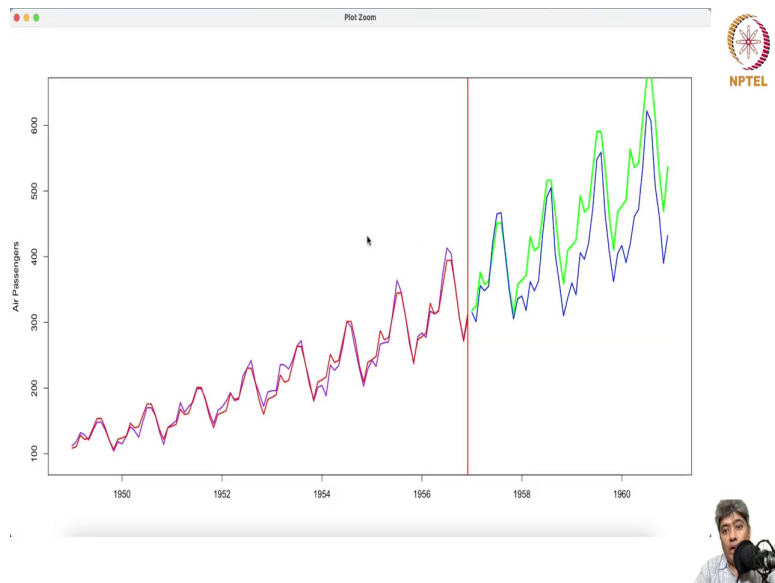
The environment pane shows the following variables:

sum1	List of 11
fit_hat	Time-Series [1:96] from 1_
m	96
n	144L
pred	Time-Series [1:48] from 1_

The plot shows 'Air Passengers' on the y-axis (ranging from 100 to 600) and years on the x-axis (1950 to 1960). A vertical red line is drawn at approximately 1956.5. The data points are blue, the in-sample fit is a green line, and the out-of-sample prediction is a blue line.

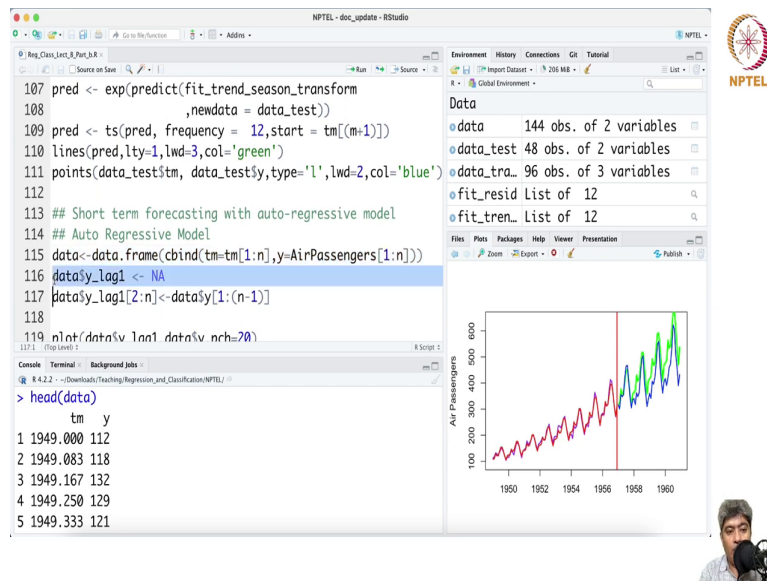
And this is out of the sample.

(Refer Slide Time: 13:24)



Forecasting the green is out of the sample forecasting it is slightly actually over estimating now.

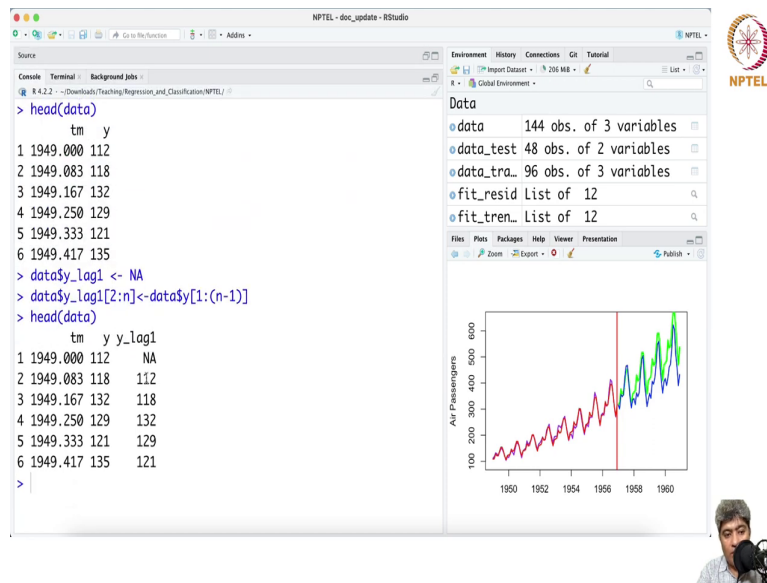
(Refer Slide Time: 13:31)



So, anyway so, this was the modeling long term modeling remember that actually why I am calling it long term modeling I am doing the entire forecast just standing here. Entire green forecast is being done while you standing in the 1957 remember that. So, it is I am doing the entire 4 year forecast while standing in the 1957 and it is doing reasonably good.

You can see it is yes of course, farther away the error will be higher, but at least the 1st year and 2nd year it has done p t decent job and it has picked up the nature of the trend and seasonality to a great extent. Now, we are going to do the short term forecasting short term forecasting with auto-regressive model ok. So, here is the data simple basic data that we have. So, basic time and the y ok and then we are taking the lag ones just created the lag ones.

(Refer Slide Time: 15:00)



The screenshot shows the RStudio interface. The console on the left displays the following R code and its output:

```
> head(data)
  tm y
1 1949.000 112
2 1949.083 118
3 1949.167 132
4 1949.250 129
5 1949.333 121
6 1949.417 135
> data$y_lag1 <- NA
> data$y_lag1[2:n] <- data$y[1:(n-1)]
> head(data)
  tm y y_lag1
1 1949.000 112 NA
2 1949.083 118 112
3 1949.167 132 118
4 1949.250 129 132
5 1949.333 121 129
6 1949.417 135 121
>
```

The Environment pane on the right shows the following objects:

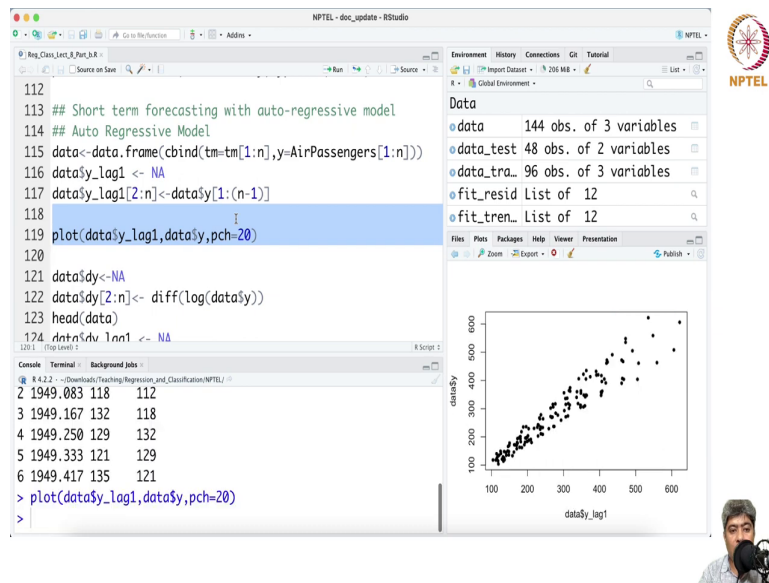
- data: 144 obs. of 3 variables
- data_test: 48 obs. of 2 variables
- data_tra_: 96 obs. of 3 variables
- fit_resid: List of 12
- fit_tren_: List of 12

The Plots pane shows a time series plot titled "Air Passengers" with the y-axis labeled "Air Passengers" ranging from 100 to 600 and the x-axis showing years from 1950 to 1960. The plot displays a red line representing the original data and a green line representing the lagged data, with a vertical red line at approximately 1956.5.

The NPTEL logo is visible in the top right corner of the RStudio window.

So, why so, 112 is being just placed here. So, if it is 112 on the February of 1949 the lag is 112 it has been kept here. If 132 is the value for the March in 112 18 is the lag value and that is being kept here similarly 132 is being kept here. So, that is how you create the lag variable ok.

(Refer Slide Time: 15:35)



The screenshot shows the RStudio interface with the following R code in the editor:

```
112
113 ## Short term forecasting with auto-regressive model
114 ## Auto Regressive Model
115 data<-data.frame(cbind(tm=tm[1:n],y=AirPassengers[1:n]))
116 data$y_lag1 <- NA
117 data$y_lag1[2:n]<-data$y[1:(n-1)]
118
119 plot(data$y_lag1,data$y,pch=20)
120
121 data$dy<-NA
122 data$dy[2:n]<- diff(log(data$y))
123 head(data)
124 data$y_lag1[1] <- NA
```

The console output shows the first six rows of the data frame:

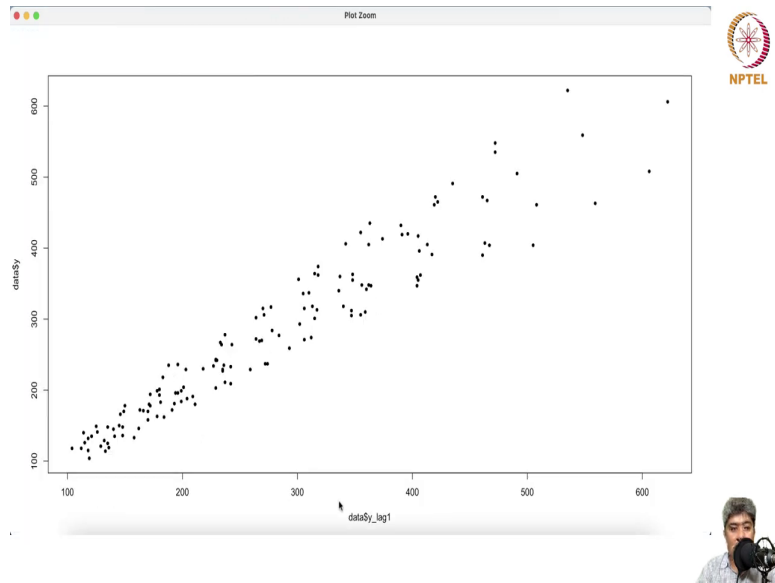
```
1 1949.083 118 112
2 1949.083 118 112
3 1949.167 132 118
4 1949.250 129 132
5 1949.333 121 129
6 1949.417 135 121
```

The plot on the right shows a scatter plot of `data$y_lag1` on the x-axis versus `data$y` on the y-axis. Both axes range from 100 to 600. The data points form a strong positive linear correlation, indicating that the lagged variable is highly correlated with the original variable.

The NPTEL logo is visible in the top right corner of the RStudio window.

Once you create the lag variable now you plot the lag variable versus the actual variable.

(Refer Slide Time: 15:43)



So, now, we can see this is the lag variable versus this is the original variable and there ok.

(Refer Slide Time: 15:50)

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for an auto-regressive model:

```
113 ## Short term forecasting with auto-regressive model
114 ## Auto Regressive Model
115 data<-data.frame(cbind(tm=tm[1:n],y=AirPassengers[1:n]))
116 data$y_lag1 <- NA
117 data$y_lag1[2:n]<-data$y[1:(n-1)]
118
119 plot(data$y_lag1,data$y,pch=20)
120
121 data$dy<-NA
122 data$dy[2:n]<- diff(log(data$y))
123 head(data)
124 data$y_lag1 <- NA
125 data$dy_lag1[2:n]<-data$dy[1:(n-1)]
```
- Environment:** Lists data objects: data (144 obs. of 4 variables), data_test (48 obs. of 2 variables), data_tra_ (96 obs. of 3 variables), fit_resid (List of 12), and fit_tren_ (List of 12).
- Console:** Shows the execution of the code from line 115 to 125, including the output of head(data) and the execution of the plot command.
- Plot:** A scatter plot with 'data\$lag1' on the x-axis and 'data\$dy' on the y-axis. Both axes range from 100 to 600. The plot shows a positive linear correlation between the lagged values and the log-differenced values.
- NPTEL Logo:** Located in the top right corner of the RStudio window.
- Video Feed:** A small inset in the bottom right corner shows a person speaking into a microphone.

Now I am going to create a log difference of the values.

(Refer Slide Time: 16:00)

The screenshot shows the RStudio interface with the following R code in the script editor:

```
113 ## Short term forecasting with auto-regressive model
114 ## Auto Regressive Model
115 data<-data.frame(cbind(tm=tm[1:n],y=AirPassengers[1:n]))
116 data$lag1 <- NA
117 data$lag1[2:n]<-data$y[1:(n-1)]
118
119 plot(data$lag1,data$y,pch=20)
120
121 data$dy<-NA
122 data$dy[2:n]<- diff(log(data$y))
123 head(data)
124 data$lag1 <- NA
125 data$lag1[2:n]<-data$dy[1:(n-1)]
```

The console output shows the following data:

1	1949.000	112	NA	NA	
2	1949.083	118	112	0.05218575	
3	1949.167	132	118	0.11211730	
4	1949.250	129	132	-0.02298952	
5	1949.333	121	129	-0.06402186	
6	1949.417	135	121	0.10948423	

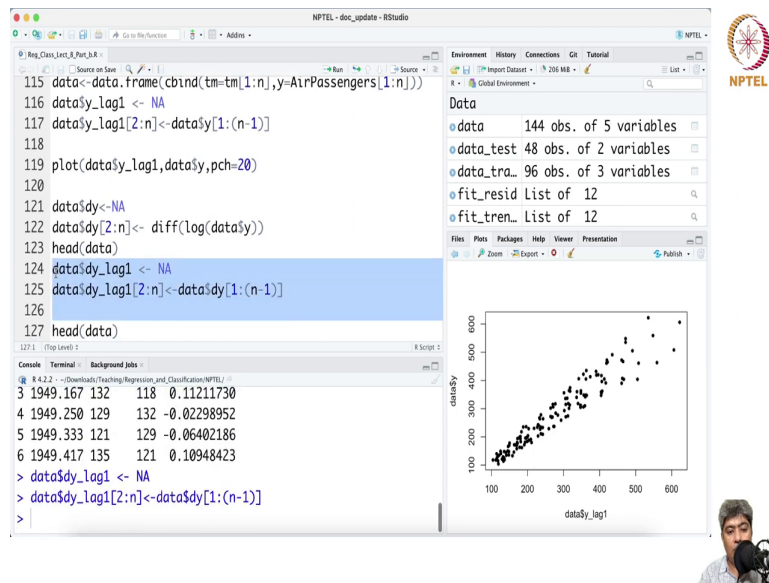
The Environment pane on the right shows the following data objects:

- data: 144 obs. of 4 variables
- data_test: 48 obs. of 2 variables
- data_tra: 96 obs. of 3 variables
- fit_resid: List of 12
- fit_tren: List of 12

The plot shows a scatter plot of data\$dy (y-axis) versus data\$lag1 (x-axis). The data points show a positive correlation, indicating that the log differences of the time series are related to their lagged values.

And then. So, if you now you see these are the delta y log differences.

(Refer Slide Time: 16:07)



The screenshot shows an RStudio interface with the following components:

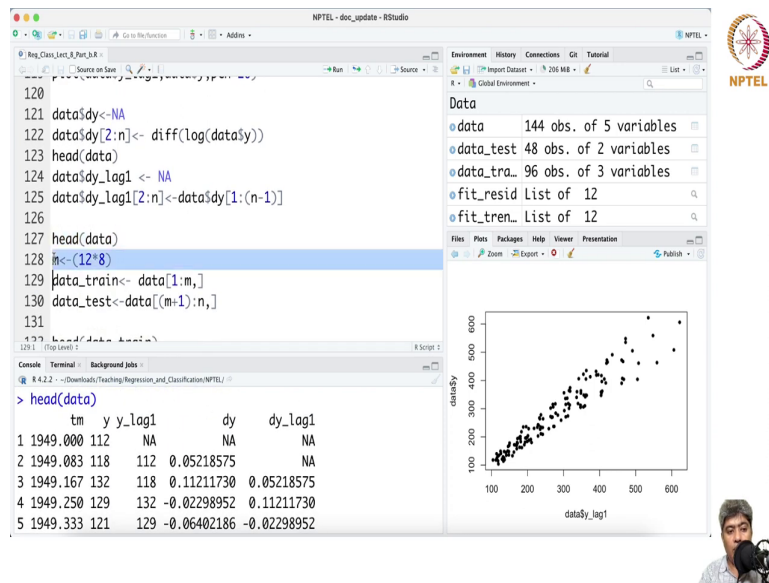
- Source Editor:** Contains R code for creating a data frame, lagging variables, and plotting. The code includes:

```
115 data<-data.frame(cbind(tm=tm[1:n],y=AirPassengers[1:n]))
116 data$y_lag1 <- NA
117 data$y_lag1[2:n]<-data$y[1:(n-1)]
118
119 plot(data$y_lag1,data$y,pch=20)
120
121 data$dy<-NA
122 data$dy[2:n]<- diff(log(data$y))
123 head(data)
124 data$dy_lag1 <- NA
125 data$dy_lag1[2:n]<-data$dy[1:(n-1)]
126
127 head(data)
```
- Environment:** Lists objects in the workspace: data (144 obs. of 5 variables), data_test (48 obs. of 2 variables), data_tra_ (96 obs. of 3 variables), fit_resid (List of 12), and fit_tren_ (List of 12).
- Console:** Shows the output of the code execution, including the first few rows of the data frame and the execution of the lagging code.

```
3 1949.167 132 118 0.11211730
4 1949.250 129 132 -0.02298952
5 1949.333 121 129 -0.06402186
6 1949.417 135 121 0.10948423
> data$dy_lag1 <- NA
> data$dy_lag1[2:n]<-data$dy[1:(n-1)]
>
```
- Plots:** A scatter plot titled 'density' vs 'data\$y_lag1' is displayed, showing a positive correlation between the two variables. The x-axis ranges from 100 to 600, and the y-axis ranges from 100 to 600.
- UI Elements:** The NPTEL logo is visible in the top right corner, and a small video feed of the presenter is in the bottom right corner.

And then lag of the log differences.

(Refer Slide Time: 16:09)



The screenshot shows an RStudio session with the following R code in the editor:

```
120
121 data$dy<-NA
122 data$dy[2:n]<- diff(log(data$y))
123 head(data)
124 data$dy_lag1 <- NA
125 data$dy_lag1[2:n]<-data$dy[1:(n-1)]
126
127 head(data)
128 m<-(12*8)
129 data_train<- data[1:m,]
130 data_test<-data[(m+1):n,]
131
```

The console output shows the result of `head(data)`:

```
> head(data)
   tm y_y_lag1    dy dy_lag1
1 1949.000 112    NA      NA
2 1949.083 118  112 0.05218575
3 1949.167 132  118 0.11211730 0.05218575
4 1949.250 129  132 -0.02298952 0.11211730
5 1949.333 121  129 -0.06402186 -0.02298952
```

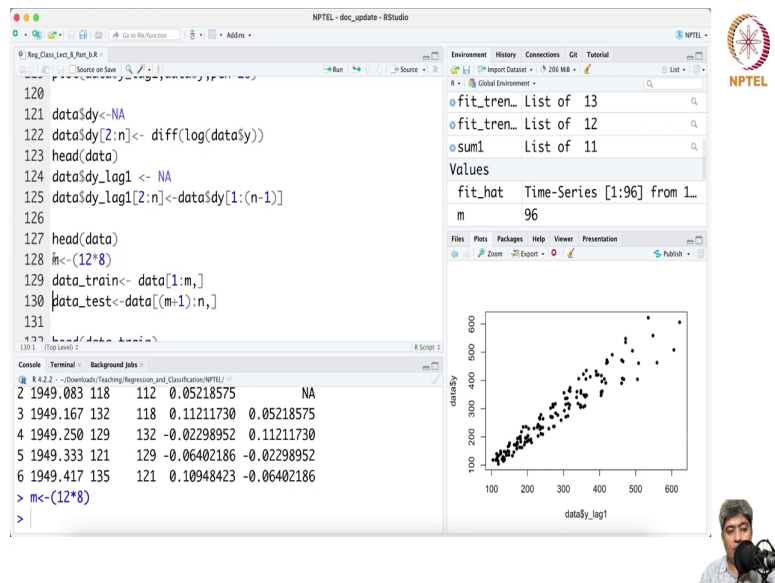
The Environment pane on the right shows the following objects:

- data: 144 obs. of 5 variables
- data_test: 48 obs. of 2 variables
- data_tra_: 96 obs. of 3 variables
- fit_resid: List of 12
- fit_tren_: List of 12

A scatter plot is displayed in the bottom right, showing the relationship between `data$y` (y-axis) and `data$y_lag1` (x-axis). Both axes range from 100 to 600. The plot shows a strong positive linear correlation between the current value of `y` and its lagged value.

I have created lag of the log differences.

(Refer Slide Time: 16:15)



The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for data manipulation and model fitting.
- Environment:** Lists objects like `fit_tren_` and `sum1`.
- Console:** Shows the execution of the code and the resulting data frame.
- Plots:** A scatter plot of `data$dy` vs `data$dy_lag1`.

```
120
121 data$dy<-NA
122 data$dy[2:n]<- diff(log(data$y))
123 head(data)
124 data$dy_lag1 <- NA
125 data$dy_lag1[2:n]<-data$dy[1:(n-1)]
126
127 head(data)
128 m<-(12*8)
129 data_train<- data[1:m,]
130 data_test<-data[(m+1):n,]
131
```

Console Output:

```
R 4.2.2 - (Downloads/Teaching/Regression_and_Classification/NPTEL) /
2 1949.083 118 112 0.05218575 NA
3 1949.167 132 118 0.11211730 0.05218575
4 1949.250 129 132 -0.02298952 0.11211730
5 1949.333 121 129 -0.06402186 -0.02298952
6 1949.417 135 121 0.10948423 -0.06402186
> m<-(12*8)
>
```

Environment:

- `fit_tren_` List of 13
- `fit_tren_` List of 12
- `sum1` List of 11

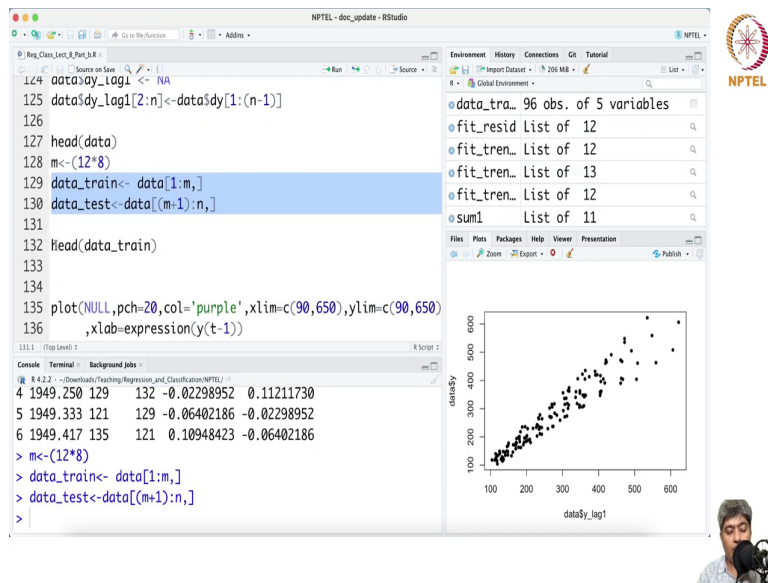
Values:

```
fit_hat Time-Series [1:96] from 1...
m
96
```

Plots:

Scatter plot showing `data$dy` (y-axis) versus `data$dy_lag1` (x-axis). Both axes range from 100 to 600. The plot shows a positive linear correlation between the two variables.

(Refer Slide Time: 16:18)



The screenshot shows an RStudio interface with the following components:

- Source Editor:** Contains R code for data manipulation and plotting. Lines 125-136 are visible, including the creation of `data_train` and `data_test` subsets, and a scatter plot of `density` vs `data$y_lag1`.
- Environment:** Lists objects in the workspace, including `data_train` (96 obs. of 5 variables), `fit_resid`, `fit_tren`, and `sum1`.
- Console:** Shows the execution of the code, including the output of `m` and the creation of `data_train` and `data_test`.
- Plots:** A scatter plot titled `density` vs `data$y_lag1` showing a positive correlation between the two variables.

```
125 data$y_lag1[2:n] <- data$y[1:(n-1)]
126
127 head(data)
128 m <- (12*8)
129 data_train <- data[1:m,]
130 data_test <- data[(m+1):n,]
131
132 head(data_train)
133
134
135 plot(NULL, pch=20, col='purple', xlim=c(90,650), ylim=c(90,650)
136       , xlab=expression(y(t-1)))
```

```
R 4.2.2 - ~/Downloads/Teaching/Regression_and_Classification/NPTEL/
4 1949.250 129 132 -0.02298952 0.11211730
5 1949.333 121 129 -0.06402186 -0.02298952
6 1949.417 135 121 0.10948423 -0.06402186
> m <- (12*8)
> data_train <- data[1:m,]
> data_test <- data[(m+1):n,]
>
```

(Refer Slide Time: 16:21)

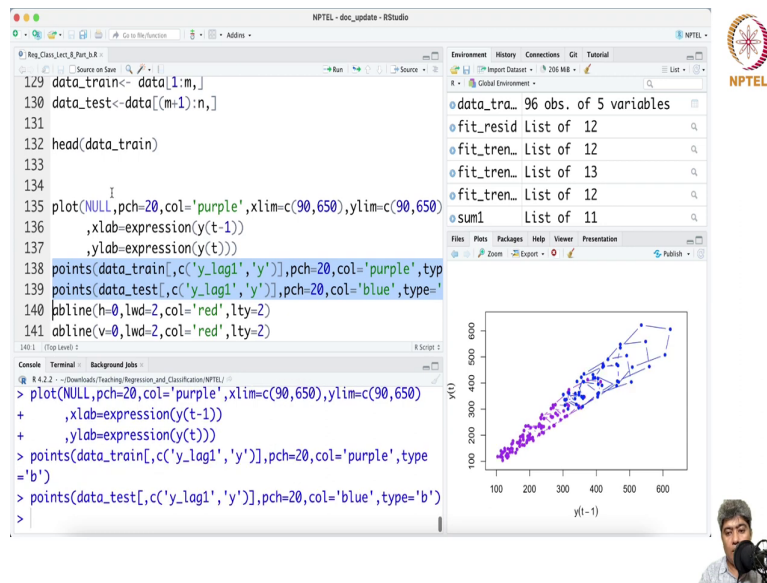
The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for data manipulation and plotting.

```
124 data$dy_lag1 <- NA
125 data$dy_lag1[2:n] <- data$dy[1:(n-1)]
126
127 head(data)
128 m <- (12^8)
129 data_train <- data[1:m,]
130 data_test <- data[(m+1):n,]
131
132 head(data_train)
133
134
135 plot(NULL, pch=20, col='purple', xlim=c(90,650), ylim=c(90,650)
136       , xlab=expression(y(t-1)))
```
- Environment:** Lists objects in the workspace:
 - data_tra_ 96 obs. of 5 variables
 - fit_resid_ List of 12
 - fit_tren_ List of 12
 - fit_tren_ List of 13
 - fit_tren_ List of 12
 - sum1 List of 11
- Console:** Shows the output of the `head(data_train)` command:

1	1949.000	112	NA	NA	NA
2	1949.083	118	112	0.05218575	NA
3	1949.167	132	118	0.11211730	0.05218575
4	1949.250	129	132	-0.02298952	0.11211730
5	1949.333	121	129	-0.06402186	-0.02298952
6	1949.417	135	121	0.10948423	-0.06402186
- Plots:** A scatter plot titled 'density' vs 'data\$y_lag1'. The x-axis ranges from 100 to 600, and the y-axis ranges from 100 to 600. The plot shows a positive linear correlation between the variables.

(Refer Slide Time: 16:27)



Again I am going to take the first 8 years of data as training data and next 4 years of data as the 4 years of data as the test data now I am going to plot this y_t versus y_{t-1} .

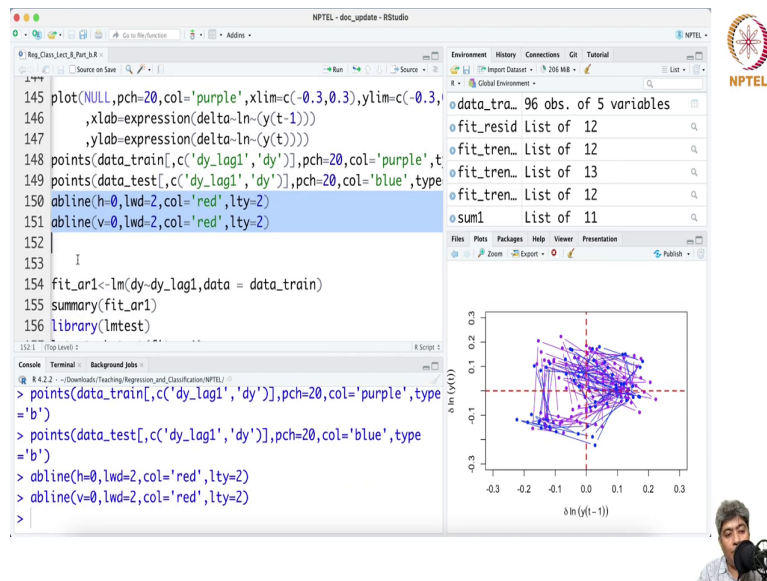
(Refer Slide Time: 16:40)

The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for plotting. Lines 137-144 define the plot's y-axis label and data points. Lines 145-147 define the plot's x-axis label and y-axis limits. Lines 148-149 plot the training and testing data points.
- Environment:** Lists objects such as `data_tra_` (96 obs. of 5 variables), `fit_resid`, `fit_tren_`, and `sum1`.
- Console:** Shows the execution of the R code, including the `plot` function call and the resulting plot.
- Plot:** A scatter plot with `y(t-1)` on the x-axis and `y(t)` on the y-axis. The plot shows a positive linear relationship between the two variables, with data points colored purple and blue. A red trend line is also visible.

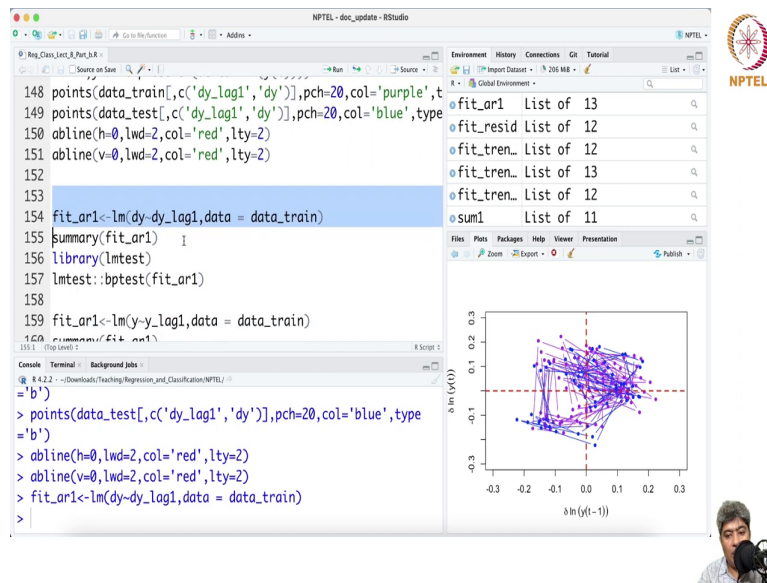
So, you can see this you know sort of a points ok.

(Refer Slide Time: 16:46)



And then also this plot you can see the this is a delta log y t minus 1 versus delta log y t and you can see that there is a sort of a hole in the data right and this whole represent the seasonality ok. So, question is there a second hole there or is it just why this case is so tense. So, this is something the topological data analysis people do consider why it is happening why the way it is happening this topology what is the topology of the data alright.

(Refer Slide Time: 17:30)



The screenshot displays the RStudio interface. The main editor window contains the following R code:

```
148 points(data_train[,c('dy_lag1', 'dy')], pch=20, col='purple', t
149 points(data_test[,c('dy_lag1', 'dy')], pch=20, col='blue', type
150 abline(h=0, lwd=2, col='red', lty=2)
151 abline(v=0, lwd=2, col='red', lty=2)
152
153
154 fit_ar1<-lm(dy-dy_lag1,data = data_train)
155 summary(fit_ar1)
156 library(lmtest)
157 lmtest::bptest(fit_ar1)
158
159 fit_ar1<-lm(y-y_lag1,data = data_train)
```

The console window shows the execution of the following commands:

```
> points(data_test[,c('dy_lag1', 'dy')], pch=20, col='blue', type
=> abline(h=0, lwd=2, col='red', lty=2)
=> abline(v=0, lwd=2, col='red', lty=2)
=> fit_ar1<-lm(dy-dy_lag1,data = data_train)
>
```

The Environment pane on the right lists several objects: fit_ar1 (List of 13), fit_resid (List of 12), fit_tren_ (List of 12), fit_tren_ (List of 13), fit_tren_ (List of 12), and sum1 (List of 11).

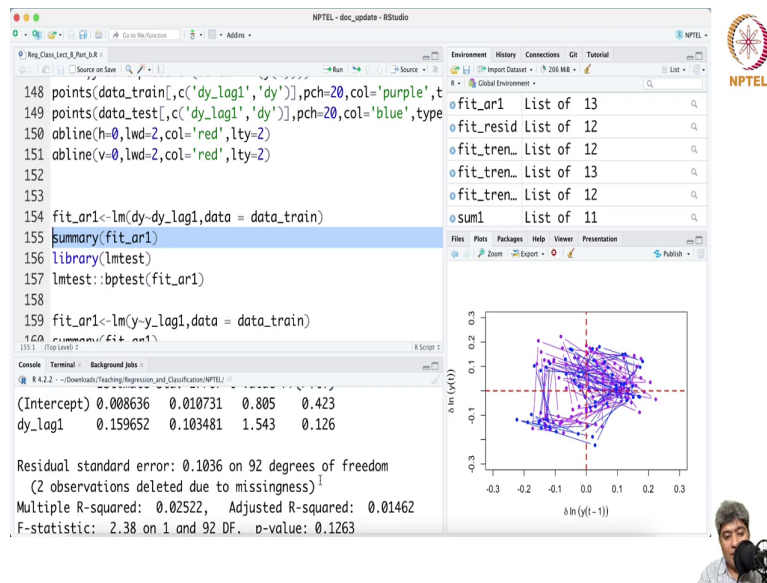
The Plots pane shows a scatter plot with the following axes:

- Y-axis: $\Delta \ln(y(t))$
- X-axis: $\Delta \ln(y(t-1))$

The plot displays purple data points for training data and blue points for test data. Two red dashed lines are drawn at $y=0$ and $x=0$. A small inset image of a person is visible in the bottom right corner of the RStudio window.

So, now, I am going to fit this model in delta y as a function of delta log y lag of y.

(Refer Slide Time: 17:39)



The screenshot displays the RStudio interface. The script editor contains the following R code:

```
148 points(data_train[,c('dy_lag1','dy')],pch=20,col='purple',t
149 points(data_test[,c('dy_lag1','dy')],pch=20,col='blue',type
150 abline(h=0,lwd=2,col='red',lty=2)
151 abline(v=0,lwd=2,col='red',lty=2)
152
153
154 fit_ar1<-lm(dy~dy_lag1,data = data_train)
155 summary(fit_ar1)
156 library(lmtest)
157 lmtest::bptest(fit_ar1)
158
159 fit_ar1<-lm(y~y_lag1,data = data_train)
```

The console output shows the results of the linear model fit:

```
(Intercept) 0.008636 0.010731 0.805 0.423
dy_lag1 0.159652 0.103481 1.543 0.126
```

Residual standard error: 0.1036 on 92 degrees of freedom
(2 observations deleted due to missingness)¹
Multiple R-squared: 0.02522, Adjusted R-squared: 0.01462
F-statistic: 2.38 on 1 and 92 DF. p-value: 0.1263

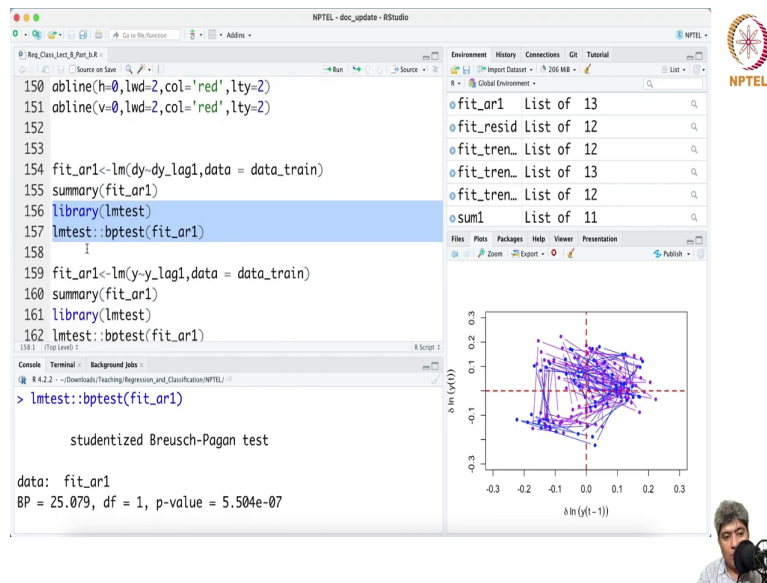
The environment pane on the right lists several objects: fit_ar1 (List of 13), fit_resid (List of 12), fit_tren_ (List of 12), fit_tren_ (List of 13), fit_tren_ (List of 12), and sum1 (List of 11).

A scatter plot is visible in the bottom right, showing $\Delta \ln(y(t))$ on the y-axis versus $\Delta \ln(y(t-1))$ on the x-axis. The plot contains purple and blue points, with red dashed lines at $y=0$ and $x=0$.

The NPTEL logo is located in the top right corner of the RStudio window.

Then summary interestingly you see they are not very really not very significant none of them are really adjusted us, but not very high.

(Refer Slide Time: 17:53)



The screenshot displays the RStudio interface. The script editor contains the following R code:

```
150 abline(h=0,lwd=2,col='red',lty=2)
151 abline(v=0,lwd=2,col='red',lty=2)
152
153
154 fit_ar1<-lm(dy-dy_lag1,data = data_train)
155 summary(fit_ar1)
156 library(lmtest)
157 lmtest::bptest(fit_ar1)
158
159 fit_ar1<-lm(y-y_lag1,data = data_train)
160 summary(fit_ar1)
161 library(lmtest)
162 lmtest::bptest(fit_ar1)
```

The console output shows the results of the Breusch-Pagan test:

```
> lmtest::bptest(fit_ar1)

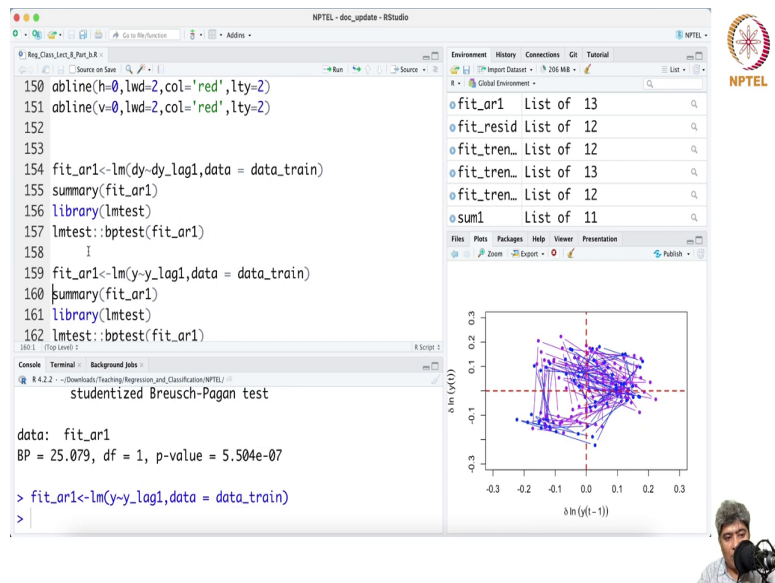
studentized Breusch-Pagan test

data: fit_ar1
BP = 25.079, df = 1, p-value = 5.504e-07
```

The Environment pane on the right lists several objects: fit_ar1 (List of 13), fit_resid (List of 12), fit_tren_ (List of 12), fit_tren_ (List of 13), fit_tren_ (List of 12), and sum1 (List of 11). A plot of the residuals is shown in the bottom right, with the x-axis labeled $\delta \ln(y(t-1))$ and the y-axis labeled $\delta \ln(y(t))$. The plot shows a scatter of points with a grid of red dashed lines at $x=0$ and $y=0$.

So, if I just run the bptest Breusch-Pagan test they are not also homogeneous. So, probably I will not use this model at delta low delta y level.

(Refer Slide Time: 18:07)



The screenshot displays the RStudio interface with the following R code in the editor:

```
150 abline(h=0,lwd=2,col='red',lty=2)
151 abline(v=0,lwd=2,col='red',lty=2)
152
153
154 fit_ar1<-lm(dy-dy_lag1,data = data_train)
155 summary(fit_ar1)
156 library(lmtest)
157 lmtest::bptest(fit_ar1)
158 I
159 fit_ar1<-lm(y-y_lag1,data = data_train)
160 summary(fit_ar1)
161 library(lmtest)
162 lmtest::bptest(fit_ar1)
```

The console output shows the results of the Breusch-Pagan test:

```
data: fit_ar1
BP = 25.079, df = 1, p-value = 5.504e-07
> fit_ar1<-lm(y-y_lag1,data = data_train)
>
```

The environment pane on the right lists several objects: fit_ar1 (List of 13), fit_resid (List of 12), fit_tren_ (List of 12), fit_tren_ (List of 13), fit_tren_ (List of 12), and sum1 (List of 11). A scatter plot in the bottom right corner shows the relationship between $\delta \ln(y(t))$ on the y-axis and $\delta \ln(y(t-1))$ on the x-axis, with red dashed lines at zero for both axes.

So, if you just model fit this model x simply y versus $\log y$ as a function of lag y auto simple auto regressive model.

(Refer Slide Time: 18:15)

The screenshot displays the RStudio interface for an AR(1) model. The script editor contains the following code:

```
150 abline(h=0,lwd=2,col='red',lty=2)
151 abline(v=0,lwd=2,col='red',lty=2)
152
153
154 fit_ar1<-lm(dy-dy_lag1,data = data_train)
155 summary(fit_ar1)
156 library(lmtest)
157 lmtest::bptest(fit_ar1)
158 I
159 fit_ar1<-lm(y-y_lag1,data = data_train)
160 summary(fit_ar1)
161 library(lmtest)
162 lmtest::bptest(fit_ar1)
```

The console output shows the following statistics:

```
Residual standard error: 23.15 on 93 degrees of freedom
(1 observation deleted due to missingness)
Multiple R-squared: 0.8963, Adjusted R-squared: 0.8952
F-statistic: 804 on 1 and 93 DF, p-value: < 2.2e-16
```

The Environment pane on the right lists the following objects:

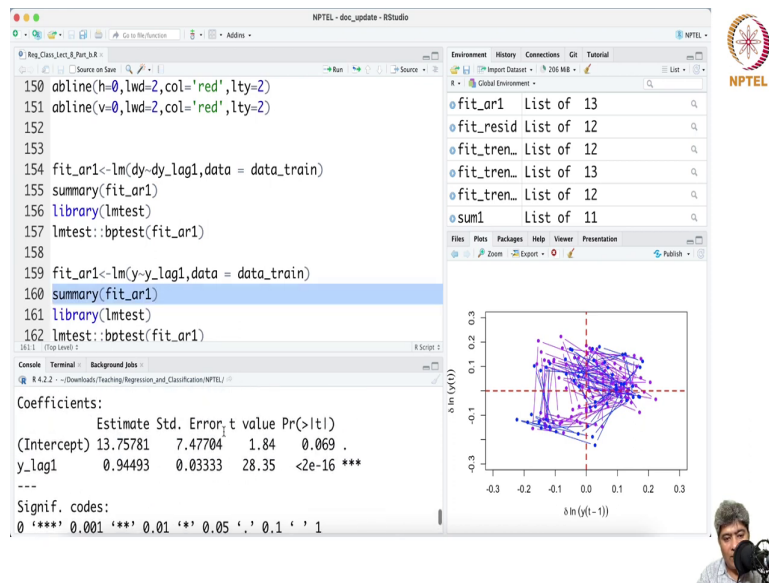
- fit_ar1 List of 13
- fit_resid List of 12
- fit_tren_ List of 12
- fit_tren_ List of 13
- fit_tren_ List of 12
- sum1 List of 11

The Plots pane shows a residual plot with the following axes:

- Y-axis: $\Delta \ln(y(t))$
- X-axis: $\Delta \ln(y(t-1))$

The plot displays a scatter of points with blue lines connecting them, centered around the origin. Two red dashed lines are drawn at $x=0$ and $y=0$. The NPTEL logo is visible in the top right corner, and a small video feed of the presenter is in the bottom right corner.

(Refer Slide Time: 18:18)



The screenshot displays the RStudio interface with the following R code in the editor:

```
150 abline(h=0,lwd=2,col='red',lty=2)
151 abline(v=0,lwd=2,col='red',lty=2)
152
153
154 fit_ar1<-lm(dy_dy_lag1,data = data_train)
155 summary(fit_ar1)
156 library(lmtest)
157 lmtest::bptest(fit_ar1)
158
159 fit_ar1<-lm(y-y_lag1,data = data_train)
160 summary(fit_ar1)
161 library(lmtest)
162 lmtest::bptest(fit_ar1)
```

The console output shows the following coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	13.75781	7.47704	1.84	0.069 .
y_lag1	0.94493	0.03333	28.35	<2e-16 ***

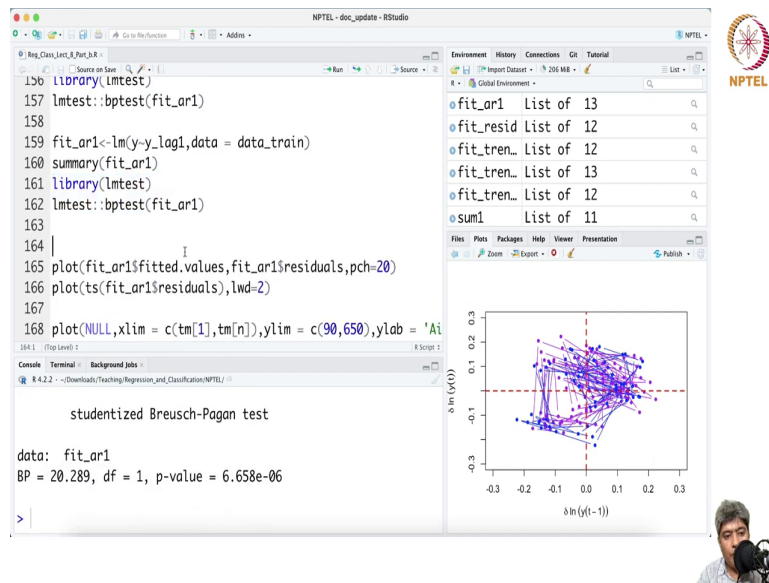
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The environment pane on the right lists several objects: fit_ar1 (List of 13), fit_resid (List of 12), fit_tren_ (List of 12), fit_tren_ (List of 13), fit_tren_ (List of 12), and sum1 (List of 11).

The plot shows the relationship between $\Delta \ln(y(t))$ on the y-axis and $\Delta \ln(y(t-1))$ on the x-axis. The plot includes a red dashed horizontal line at $y=0$ and a red dashed vertical line at $x=0$. The data points are scattered around the origin, showing a positive correlation.

So, at least immediately what we are seeing that it is very strong model with Adjusted R-square 0.89 and we run a Breusch-Pagan test.

(Refer Slide Time: 18:27)



The screenshot displays the RStudio interface. The script editor on the left contains the following R code:

```
157 lmtest::bptest(fit_ar1)
158
159 fit_ar1<-lm(y~y_lag1,data = data_train)
160 summary(fit_ar1)
161 library(lmtest)
162 lmtest::bptest(fit_ar1)
163
164 |
165 plot(fit_ar1$fitted.values,fit_ar1$residuals,pch=20)
166 plot(ts(fit_ar1$residuals),lwd=2)
167
168 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Ai
```

The Environment pane on the right shows several objects: fit_ar1 (List of 13), fit_resid (List of 12), fit_tren_ (List of 12), fit_tren_ (List of 13), fit_tren_ (List of 12), and sum1 (List of 11).

The console at the bottom left displays the results of the Breusch-Pagan test:

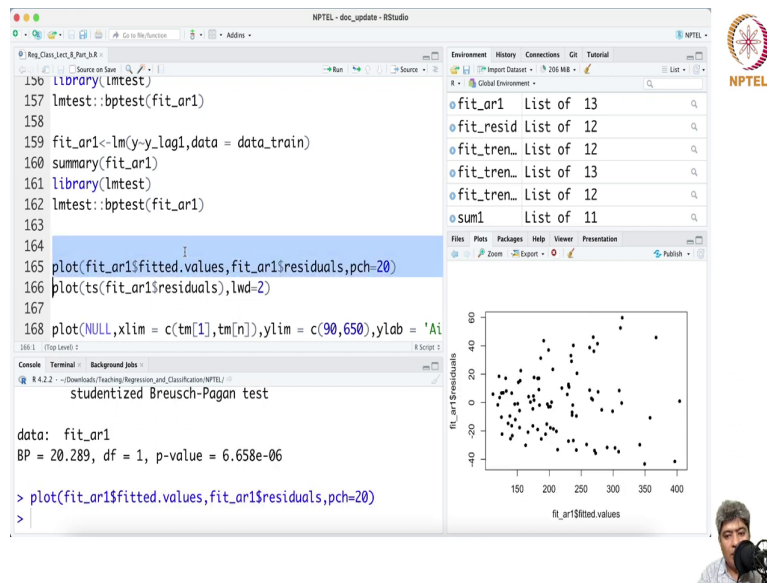
```
studentized Breusch-Pagan test
data: fit_ar1
BP = 20.289, df = 1, p-value = 6.658e-06
```

The plot window on the right shows a scatter plot of residuals. The x-axis is labeled $\delta \ln(y(t-1))$ and the y-axis is labeled $\delta \ln(y(t))$. The plot shows a dense cluster of points around the origin, with a vertical dashed line at $x=0$ and a horizontal dashed line at $y=0$.

The NPTEL logo is visible in the top right corner of the RStudio window.

The Breusch-Pagan test still rejects the you know homogeneity.

(Refer Slide Time: 18:36)



The screenshot displays the RStudio interface. The console shows the following code and output:

```
156 library(lmtest)
157 lmtest::bptest(fit_ar1)
158
159 fit_ar1<-lm(y~y_lag1,data = data_train)
160 summary(fit_ar1)
161 library(lmtest)
162 lmtest::bptest(fit_ar1)
163
164
165 plot(fit_ar1$fitted.values,fit_ar1$residuals,pch=20)
166 plot(ts(fit_ar1$residuals),lwd=2)
167
168 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Ai
```

The environment pane on the right lists several objects: fit_ar1 (List of 13), fit_resid (List of 12), fit_tren_ (List of 12), fit_tren_ (List of 13), fit_tren_ (List of 12), and sum1 (List of 11).

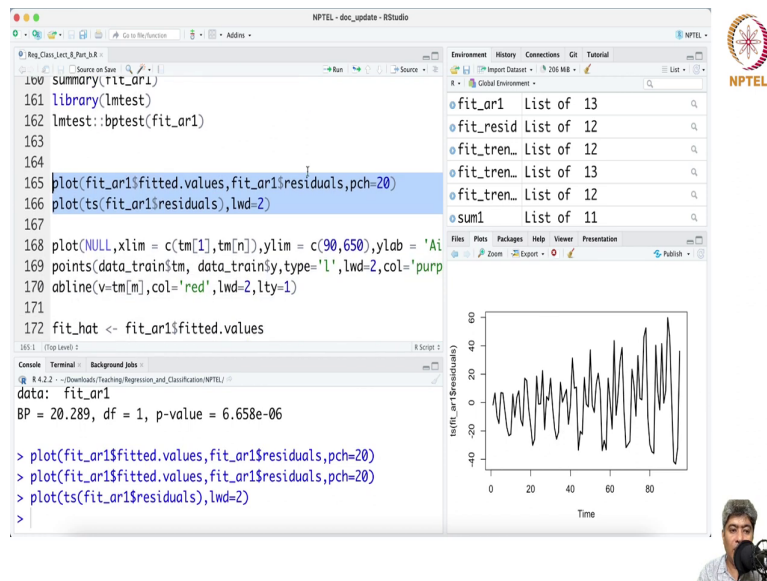
The console output includes a studentized Breusch-Pagan test result:

```
data: fit_ar1
BP = 20.289, df = 1, p-value = 6.658e-06
```

The plot shows a scatter of residuals against fitted values, exhibiting a clear fan shape, which is characteristic of heteroscedasticity. The x-axis is labeled 'fit_ar1\$fitted values' and ranges from 150 to 400. The y-axis is labeled 'fit_ar1\$residuals' and ranges from -40 to 60.

And if we fit this you can clearly see at the beginning it was much more dense, but as it increases the residual increases with that its slight a fan kind of you know fan kind of behavior. So, clearly its not homogeneous.

(Refer Slide Time: 18:55)



The screenshot displays the RStudio interface for an AR(1) model. The script editor on the left contains the following code:

```
summary(fit_ar1)
library(lmtest)
lmtest::bptest(fit_ar1)
plot(fit_ar1$fitted.values, fit_ar1$residuals, pch=20)
plot(ts(fit_ar1$residuals), lwd=2)
plot(NULL, xlim = c(tm[1], tm[n]), ylim = c(-90, 650), ylab = 'Ai')
points(data_train$tm, data_train$y, type='l', lwd=2, col='purple')
abline(v=tm[n], col='red', lwd=2, lty=1)
fit_hat <- fit_ar1$fitted.values
```

The console output shows the results of the Breusch-Pagan test:

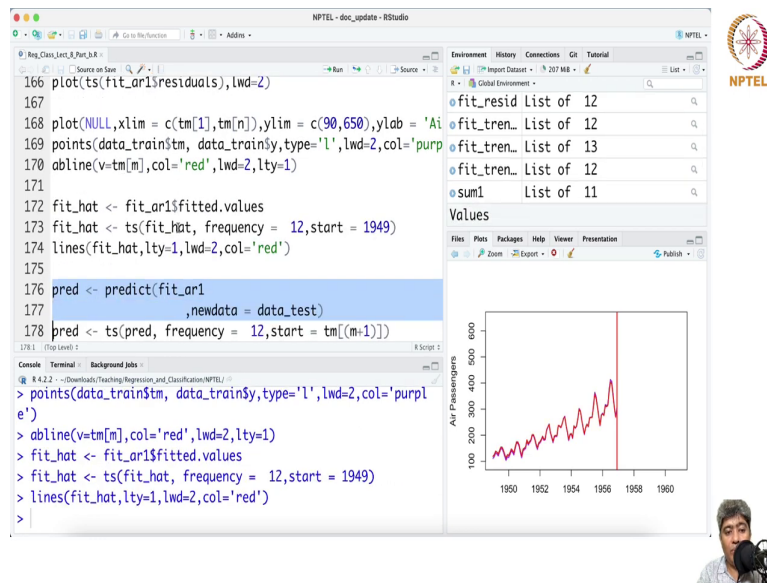
```
data: fit_ar1
BP = 20.289, df = 1, p-value = 6.658e-06
```

The Environment pane on the right lists the objects: fit_ar1 (List of 13), fit_resid (List of 12), fit_tren_ (List of 12), fit_tren_ (List of 13), fit_tren_ (List of 12), and sum1 (List of 11).

The plot window shows a time series plot of the residuals. The x-axis is labeled 'Time' and ranges from 0 to 80. The y-axis is labeled 'ts(fit_ar1\$residuals)' and ranges from -40 to 60. The plot shows a highly volatile time series with a clear upward trend, indicating a positive drift in the residuals over time.

And a line plot also sort of in initially it is the residuals these are the residuals much less, but the residuals much higher at the end points.

(Refer Slide Time: 19:12)



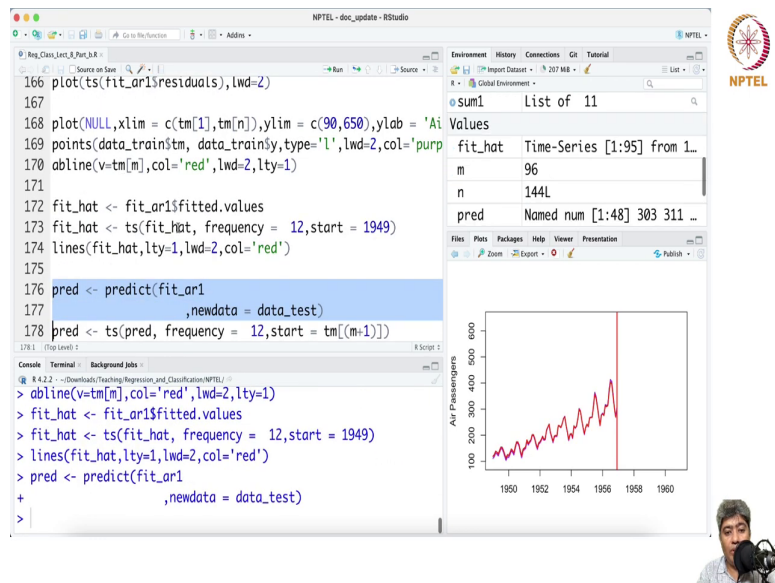
The screenshot displays an RStudio interface with the following components:

- Source Editor:** Contains R code for plotting residuals, training data points, fitted values, and predictions. The code includes:

```
166 plot(ts(fit_ar1$residuals),lwd=2)
167
168 plot(NULL,xlim = c(tm[1],tm[n]),ylim = c(90,650),ylab = 'Air
169 points(data_train$tm, data_train$y,type='l',lwd=2,col='purple'
170 abline(v=tm[m],col='red',lwd=2,ty=1)
171
172 fit_hat <- fit_ar1$fitted.values
173 fit_hat <- ts(fit_hat, frequency = 12,start = 1949)
174 lines(fit_hat,ty=1,lwd=2,col='red')
175
176 pred <- predict(fit_ar1
177               ,newdata = data_test)
178 pred <- ts(pred, frequency = 12,start = tm[(m+1)])
```
- Environment:** Lists objects such as `fit_resid`, `fit_tren`, and `sum1`.
- Console:** Shows the execution of the code from the source editor, including the `plot` and `lines` commands.
- Plots:** A time series plot titled "Air Passengers" showing data from 1950 to 1960. The y-axis ranges from 100 to 600. A vertical red line is drawn at approximately 1949, indicating the split between training and testing data.
- NPTEL Logo:** Located in the top right corner of the RStudio window.

So, if we just fit this guys, but still if you fit this guys that they are doing pretty good job in terms of prediction and in terms of out of the sample prediction.

(Refer Slide Time: 19:17)



The screenshot displays the RStudio environment with the following code in the script editor:

```
166 plot(ts(fit_ar1$residuals), lwd=2)
167
168 plot(NULL, xlim = c(tm[1], tm[n]), ylim = c(90, 650), ylab = 'Air
169 passengers', data_train$tm, data_train$y, type='l', lwd=2, col='purple',
170 abline(v=tm[m], col='red', lwd=2, lty=1)
171
172 fit_hat <- fit_ar1$fitted.values
173 fit_hat <- ts(fit_hat, frequency = 12, start = 1949)
174 lines(fit_hat, lty=1, lwd=2, col='red')
175
176 pred <- predict(fit_ar1
177                 , newdata = data_test)
178 pred <- ts(pred, frequency = 12, start = tm[(m+1)])
```



The console shows the execution of these commands:

```
> abline(v=tm[m], col='red', lwd=2, lty=1)
> fit_hat <- fit_ar1$fitted.values
> fit_hat <- ts(fit_hat, frequency = 12, start = 1949)
> lines(fit_hat, lty=1, lwd=2, col='red')
> pred <- predict(fit_ar1
+                 , newdata = data_test)
+ 
```

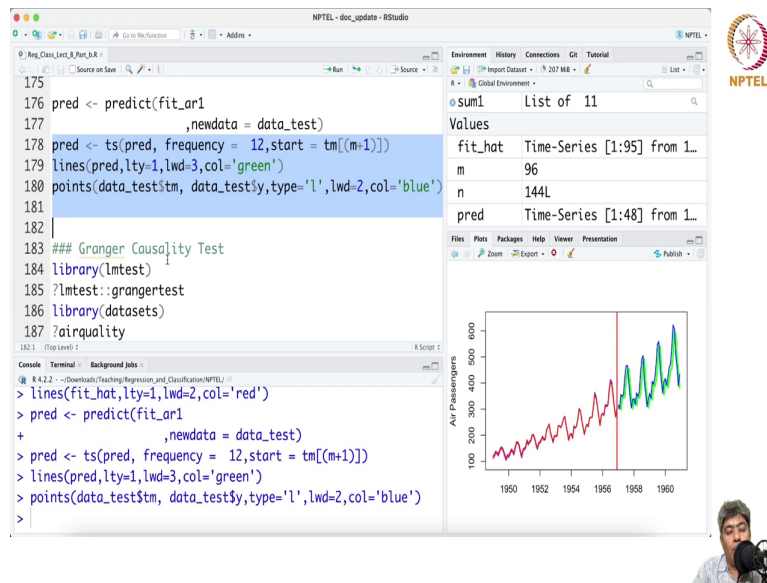
The Environment pane shows a list of objects:

Object	Type
sum1	List of 11
fit_hat	Time-Series [1:95] from 1...
m	96
n	144L
pred	Named num [1:48] 303 311 ...

The Plots pane displays a time series plot titled "Air Passengers". The y-axis is labeled "Air Passengers" and ranges from 100 to 600. The x-axis shows years from 1950 to 1960. A red line represents the fitted values, and a vertical red line is drawn at the year 1957 (corresponding to $t = m$).



(Refer Slide Time: 19:19)



The screenshot displays the RStudio interface. The main editor shows the following R code:

```
175 pred <- predict(fit_ar1
176                 ,newdata = data_test)
177
178 pred <- ts(pred, frequency = 12,start = tm[(m-1)])
179 lines(pred,lty=1,lwd=3,col='green')
180 points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
181
182
183 ## Granger Causality Test
184 library(lmtest)
185 ?lmtest::grangertest
186 library(datasets)
187 ?airquality
```

The console shows the execution of the code:

```
> lines(fit_hat,lty=1,lwd=2,col='red')
> pred <- predict(fit_ar1
+                 ,newdata = data_test)
> pred <- ts(pred, frequency = 12,start = tm[(m-1)])
> lines(pred,lty=1,lwd=3,col='green')
> points(data_test$tm, data_test$y,type='l',lwd=2,col='blue')
>
```

The environment pane shows the following variables:

Variable	Value
sum1	List of 11
fit_hat	Time-Series [1:95] from 1_
m	96
n	144L
pred	Time-Series [1:48] from 1_

The plot shows 'Air Passengers' on the y-axis (ranging from 100 to 600) and years on the x-axis (1950 to 1960). A vertical red line is drawn at approximately 1956.5. The training data is shown as a red line, the test data as a green line, and the predicted values as blue points.

So, we will stop here we will stop here and next video we will move to the Granger Causality test and what is Granger causality and how it is going to you know play a big role in the causal inference and how simple linear regression model that we discussed in this; discussing in this can be used to develop some Granger causal model and Granger can be used in in defining Granger causal inference.

Thank you very much, see you in the next video, see you soon, take care bye.