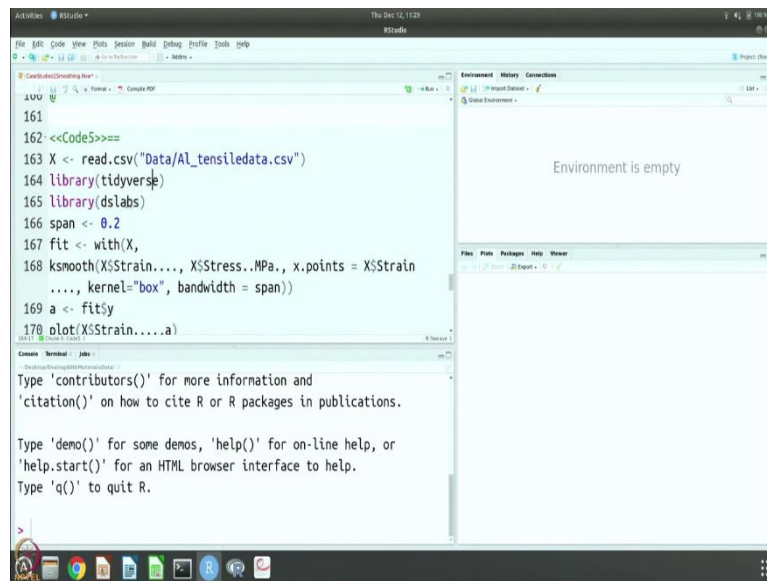


**Dealing with Materials Data: Collection, Analysis and Interpretation**  
**Professor M P Gururajan**  
**Professor Hina A Gokhale**  
**Department of Metallurgical Engineering and Materials Science**  
**Indian Institute of Technology, Bombay**  
**Lecture No. 94**  
**Case study 1: Data smoothing II**

We are continuing on Data Smoothing. We took the aluminium tensile strength data and we smoothen it ourselves and calculated the measures like modulus and the strength and so on. But it is also possible to smoothen the data using some of the existing libraries and Irizarry's book gives lots of these details.

So in this session, we are going to take two of those commands that is given, the libraries that are given in Irizarry's textbook. We are going to use them and we are going to use those function curves to do the smoothing of the data aluminium (0:52) and I will show you the difference between the two and how do they differ in terms of how they smoothen the data out.

(Refer Slide Time: 1:03)



```
161
162 <<Code5>>==
163 X <- read.csv("Data/Al_tensiledata.csv")
164 library(tidyverse)
165 library(dslabs)
166 span <- 0.2
167 fit <- with(X,
168 ksmooth(X$Strain..., X$Stress.MPa., x.points = X$Strain
..., kernel="box", bandwidth = span))
169 a <- fit$y
170 plot(X$Strain....a)
```

Environment is empty

File Plot Packages Help Viewer

Console Terminal Jobs

Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.

Type 'q()' to quit R.

Activities RStudio Thu Dec 12, 11:30

```

165 library(dslabs)
166 span <- 0.2
167 fit <- with(X,
168 ksmooth(X$Strain..., X$Stress..MPa., x.points = X$Strain
..., kernel="box", bandwidth = span))
169 a <- fit$y
170 plot(X$Strain..., a)
171 X %>% mutate(smooth = fit$y) %>%
172 ggplot(aes(X$Strain..., X$Stress..MPa.)) +
173 geom_point(size = 1.0, alpha = .7, color = "black") +
174 geom_line(aes(X$Strain..., smooth), color="red")

```

Environment History Connections

Data

- fit List of 2
- X 1061 obs. of 2 variables

Values

- a num [1:1061] 15.5 15.5 15.5 15.5 ...
- span 0.2

Activities RStudio Thu Dec 12, 11:30

Plot Zoom

Environment History Connections

Data

- List of 2
- 1061 obs. of 2 variables
- num [1:1061] 15.5 15.5 15.5 15.5 ...
- 0.2

Activities RStudio Thu Dec 12, 11:30

Plot Zoom

Environment History Connections

Data

- List of 2
- 1061 obs. of 2 variables
- num [1:1061] 15.5 15.5 15.5 15.5 ...
- 0.2

```

191 LinRegion <-
192 data.frame("Strain"=SmootherData$Strain[c(200:300)],
193            "Stress"=SmootherData$Stress[c(200:300)])
194 fit <- lm(LinRegion$Stress ~ LinRegion$Strain)
195 fit
196 summary(fit)
197 plot(LinRegion)
198 abline(fit$coefficients)
199 plot(fit$residuals)
200 qqnorm(fit$residuals)
201 @

```

F-statistic: 1.164e+05 on 1 and 99 DF, p-value: < 2.2e-16

```

> plot(LinRegion)
> abline(fit$coefficients)
> plot(fit$residuals)
> qqnorm(fit$residuals)
> plot(fit$residuals)
>

```

Environment: History Connections  
 LinRegion 101 obs. of 2 variables  
 SmootherData 1061 obs. of 2 variables  
 X 1061 obs. of 2 variables

Values  
 a num [1:1061] 1.56 1.56 1.56 1.56 ...  
 span 0.1

```

191 LinRegion <-
192 data.frame("Strain"=SmootherData$Strain[c(200:300)],
193            "Stress"=SmootherData$Stress[c(200:300)])
194 fit <- lm(LinRegion$Stress ~ LinRegion$Strain)
195 fit
196 summary(fit)
197 plot(LinRegion)
198 abline(fit$coefficients)
199 plot(fit$residuals)
200 qqnorm(fit$residuals)
201 @

```

```

> plot(LinRegion)
> abline(fit$coefficients)
> plot(fit$residuals)
> qqnorm(fit$residuals)
> plot(fit$residuals)
> plot(LinRegion)
> abline(fit$coefficients)
>

```

Environment: History Connections  
 LinRegion 101 obs. of 2 variables  
 SmootherData 1061 obs. of 2 variables  
 X 1061 obs. of 2 variables

Values  
 a num [1:1061] 1.56 1.56 1.56 1.56 ...  
 span 0.1

```

191 LinRegion <-
192 data.frame("Strain"=SmootherData$Strain[c(200:300)],
193            "Stress"=SmootherData$Stress[c(200:300)])
194 fit <- lm(LinRegion$Stress ~ LinRegion$Strain)
195 fit
196 summary(fit)
197 plot(LinRegion)
198 abline(fit$coefficients)
199 plot(fit$residuals)
200 qqnorm(fit$residuals)
201 @

```

```

              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -8.6312      0.1805  -47.83  <2e-16 ***
LinRegion$Strain 683.9573    2.0022  341.15  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

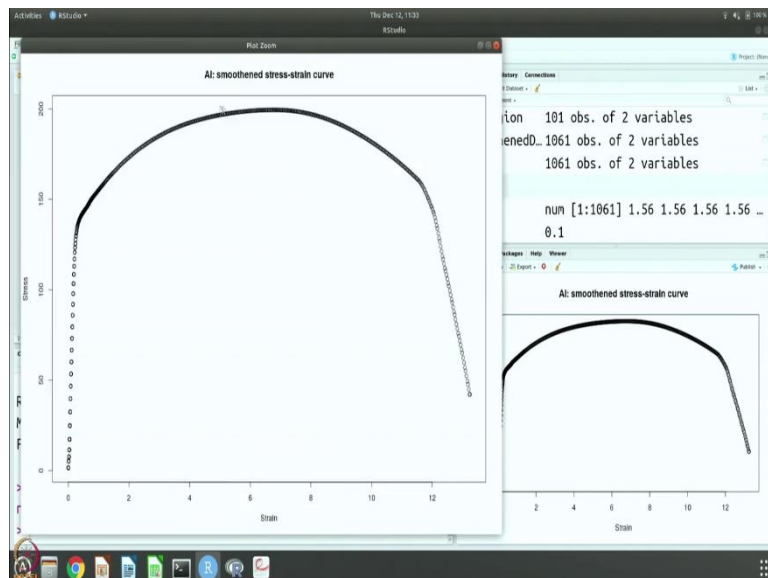
Residual standard error: 0.5594 on 99 degrees of freedom
Multiple R-squared:  0.9992,    Adjusted R-squared:  0.9991

```

Environment: History Connections  
 LinRegion 101 obs. of 2 variables  
 SmootherData 1061 obs. of 2 variables  
 X 1061 obs. of 2 variables

Values  
 a num [1:1061] 1.56 1.56 1.56 1.56 ...  
 span 0.1

Stress (MPa)	Strain (%)
-0.15	0
0.01	0
0.03	0
0.27	0
0.62	0
0.85	0
1.29	0
1.54	0
1.6	0
1.33	0
1.21	0
1.17	0
1.38	0
1.66	0
1.91	0



So the first one, let us read the data and use the library ‘tidyverse’ and ‘ds labs’ and you have to give a measure called span and this is equivalent to the, this is like the bin that we gave earlier. And in this case, what is done is that you take the data and you try to fit them, you split the data into many different smaller portions and you try to fit these given data and using that and you do a box average and you get the smoothed version here.

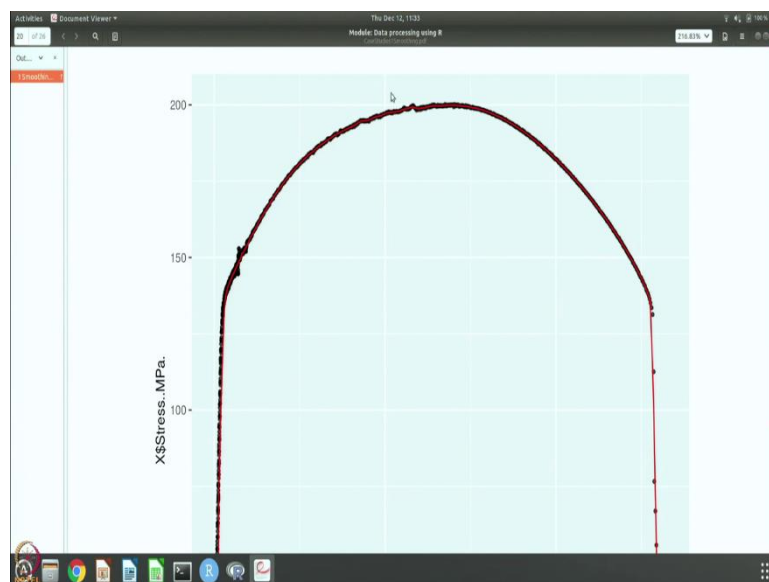
So let us run this and, so here is the data that is given and it is better if we zoom in. So here is the data that is plotted and here is our smoothed curve that is plotted, but if you see in these regions, this smoothing actually also follows these noise, so if you just look at the fitted curve, let us, so you will see that there are these small bumps that you see even in the fitted curve, so it is not strictly for example here, it is not really that smooth.

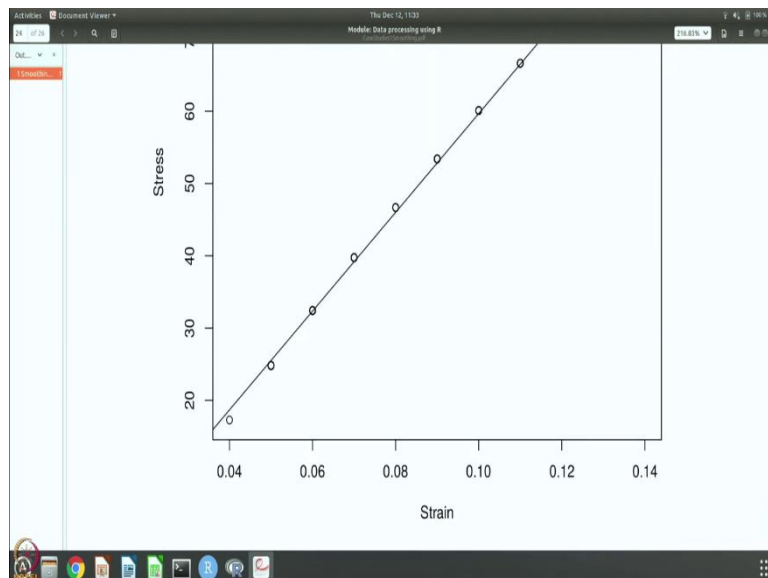
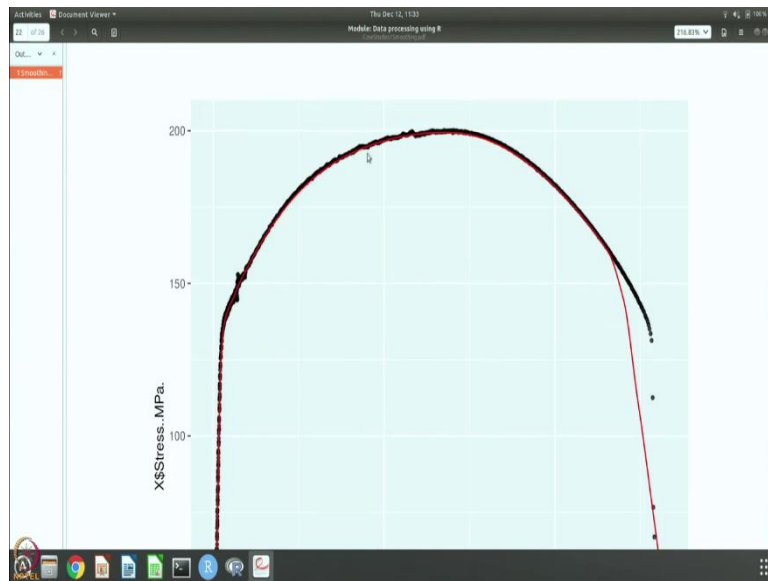
So one of the things that you can change is by changing these parameters, you can try to get a better fit. There is also this other way which is 'loess' which is used to do the fitting which will give you much more smoother fit and this is also from Irizarry's book and this is also there in 'tidyverse' and 'dslabs' library. And so let us do this fitting.

So we have got the QQ norm to show it is a straight line and okay, so this is the residuals, it is really not that normal and of course... So this is the linear portion that we are fitting and these are the data points and that is the fit and summary fit will tell you the coefficients that you get. So here also you get 683, like we pointed out earlier, so the stress is in MPa, the strain is in %, so 683 would be 683 into 100 MPa, so that will be 68305.73 MPa, that will be 68.3 GPa and that is the Young's modulus for aluminium that is known.

So we are getting the right number and if you want to get the error in that quantity, so of course, this is 200, so it is 683, 68.3 plus or minus point 2, so that is the error that we are getting in our estimate of the modulus. Of course, we should also plot the other one, Smoothened data in this case. And you can see that now all these different bumps have sort of disappeared and this is a much better Smoothened data and that is what you also see here.

(Refer Slide Time: 5:14)





So this is the one where you see the wiggles but here is one where you see that all the wiggles are even doubt and you get there proper curve and so these fitting also give you the same modulus namely 68.3 and in the case of brose data if you do, you will get about 100 GPa as the Young's modulus of the brass samples on which the tests are conducted. So these are different ways of doing smoothing.

And like we have mentioned it is a good idea to go to Irizarry's book and learn more about smoothing, its useful for analysis of data, it also leads you naturally to mission learning algorithms and we have also been indicating how manual intervention is needed the way we are doing the analysis but the machine's already do without any manual interventions some of these calculations and give you the parameters.

So how do they optimise these magic numbers like how many bins over which you have to average and things like that or is there a way that you can write a program which will do it automatically and from the different fits that it gets or different parameters evaluate, it will decide what is the optimal parameter using which it has to calculate this quantities and report it to you. So that is slightly more involved exercise, but it is an interesting and useful exercise for you to do. Thank you.