



# Smoothing

Jeffrey Leek, Assistant Professor of Biostatistics  
Johns Hopkins Bloomberg School of Public Health

# Key ideas

- Sometimes there are non-linear trends in data
- We can use "smoothing" to try to capture these
- Still a risk of overfitting
- Often hard to interpret

# CD4 Data

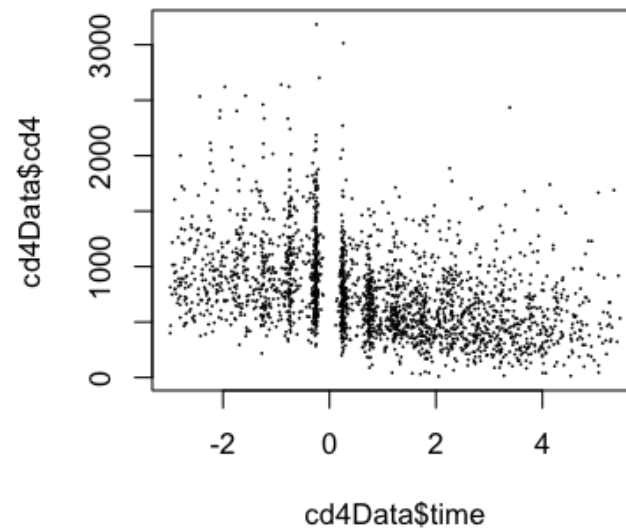
```
download.file("https://spark-public.s3.amazonaws.com/dataanalysis/cd4.data",
              destfile="./data/cd4.data",method="curl")
cd4Data <- read.table("./data/cd4.data",
                     col.names=c("time", "cd4", "age", "packs", "drugs", "sex",
                                "cesd", "id"))
cd4Data <- cd4Data[order(cd4Data$time),]
head(cd4Data)
```

	time	cd4	age	packs	drugs	sex	cesd	id
1279	-2.990	814	6.17	3	1	5	-3	30183
2190	-2.990	400	-6.02	0	0	3	-4	41406
1167	-2.984	467	13.94	0	1	1	0	30046
1427	-2.957	749	-4.54	0	1	-1	-7	30498
2032	-2.951	1218	5.57	3	1	5	3	41032
1813	-2.949	1015	-9.15	2	1	0	-7	40375

<http://www.cbc.b.umd.edu/~hcorrada/PracticalML/>

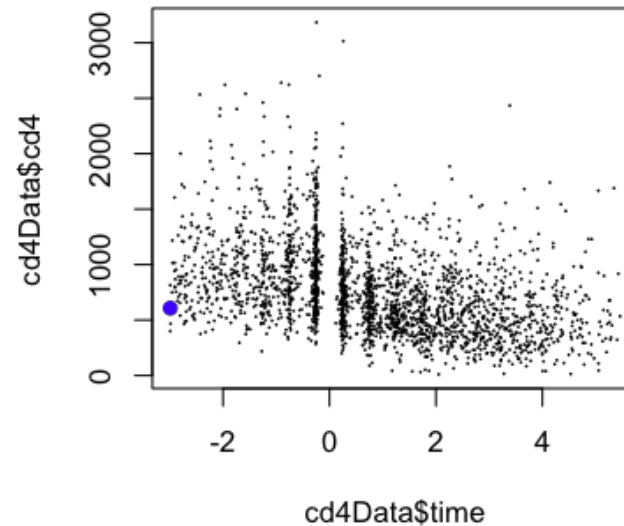
# CD4 over time

```
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1)
```



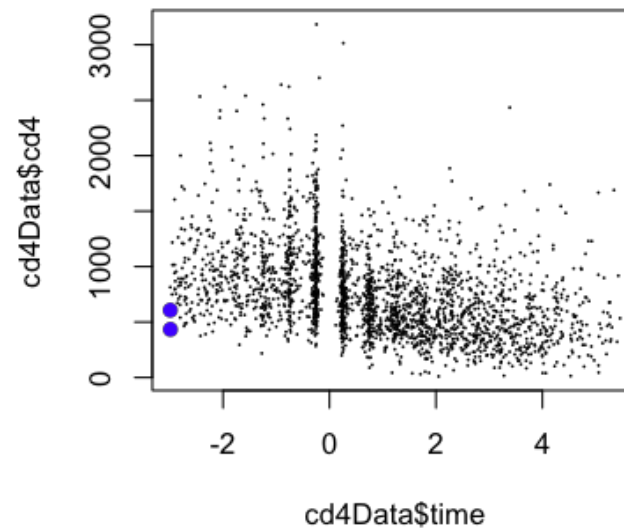
# Average first 2 points

```
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1)  
points(mean(cd4Data$time[1:2]), mean(cd4Data$cd4[1:2]), col="blue", pch=19)
```



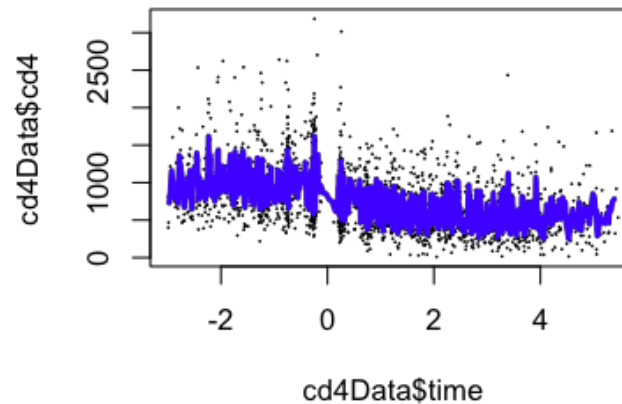
# Average second and third points

```
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1)
points(mean(cd4Data$time[1:2]), mean(cd4Data$cd4[1:2]), col="blue", pch=19)
points(mean(cd4Data$time[2:3]), mean(cd4Data$cd4[2:3]), col="blue", pch=19)
```



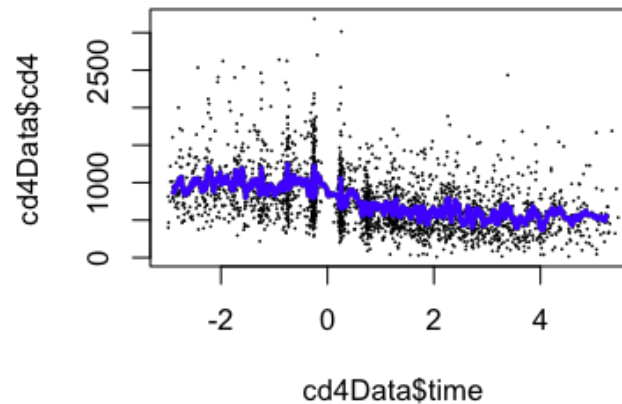
# A moving average

```
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1)
aveTime <- aveCd4 <- rep(NA, length(3:(dim(cd4Data)[1]-2)))
for(i in 3:(dim(cd4Data)[1]-2)){
  aveTime[i] <- mean(cd4Data$time[(i-2):(i+2)])
  aveCd4[i] <- mean(cd4Data$cd4[(i-2):(i+2)])
}
lines(aveTime, aveCd4, col="blue", lwd=3)
```



# Average more points

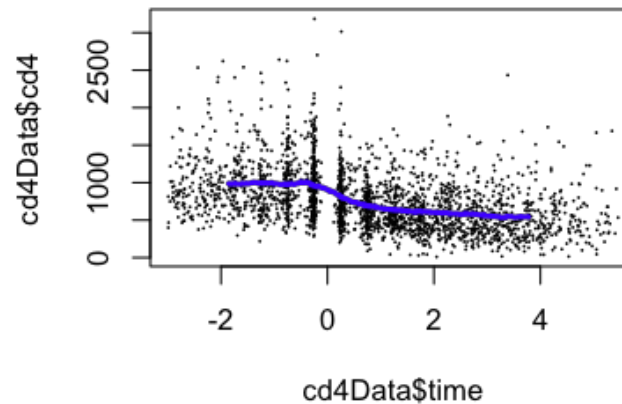
```
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1)
aveTime <- aveCd4 <- rep(NA, length(11:(dim(cd4Data)[1]-10)))
for(i in 11:(dim(cd4Data)[1]-2)){
  aveTime[i] <- mean(cd4Data$time[(i-10):(i+10)])
  aveCd4[i] <- mean(cd4Data$cd4[(i-10):(i+10)])
}
lines(aveTime, aveCd4, col="blue", lwd=3)
```





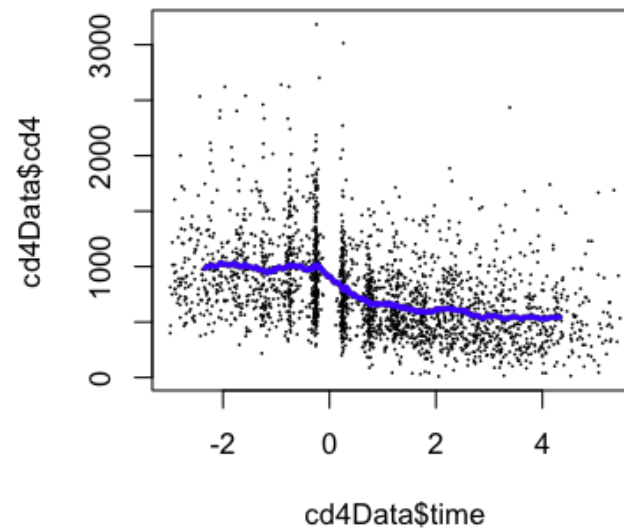
# Average many more

```
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1)
aveTime <- aveCd4 <- rep(NA, length(201:(dim(cd4Data)[1]-200)))
for(i in 201:(dim(cd4Data)[1]-200)){
  aveTime[i] <- mean(cd4Data$time[(i-200):(i+200)])
  aveCd4[i] <- mean(cd4Data$cd4[(i-200):(i+200)])
}
lines(aveTime, aveCd4, col="blue", lwd=3)
```



# A faster way

```
filtTime <- as.vector(filter(cd4Data$time,filter=rep(1,200))/200)  
filtCd4 <- as.vector(filter(cd4Data$cd4,filter=rep(1,200))/200)  
plot(cd4Data$time,cd4Data$cd4,pch=19,cex=0.1); lines(filtTime,filtCd4,col="blue",lwd=3)
```



# Averaging = weighted sums

```
filtCd4 <- as.vector(filter(cd4Data$cd4, filter=rep(1,4))/4)  
filtCd4[2]
```

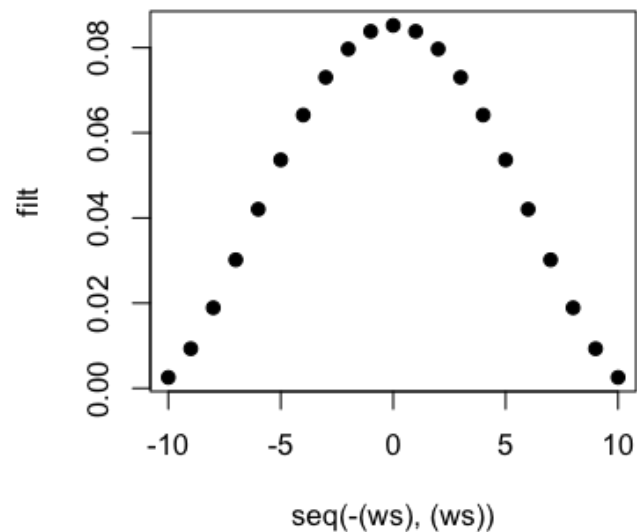
```
[1] 607.5
```

```
sum(cd4Data$cd4[1:4] * rep(1/4,4))
```

```
[1] 607.5
```

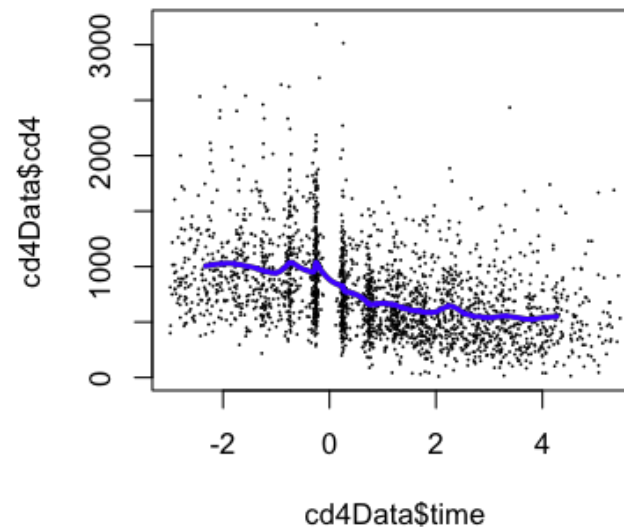
# Other weights -> should sum to one

```
ws = 10; tukey = function(x) pmax(1 - x^2, 0)^2  
filt= tukey(seq(-ws,ws)/(ws+1));filt=filt/sum(filt)  
plot(seq(-(ws), (ws)),filt,pch=19)
```



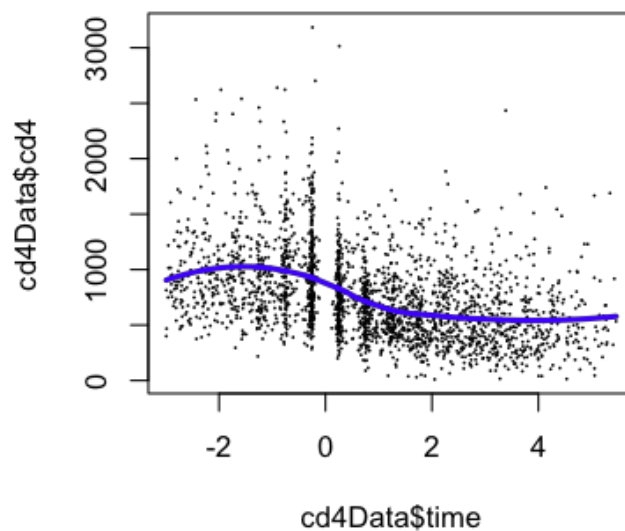
# Other weights -> should sum to one

```
ws = 100; tukey = function(x) pmax(1 - x^2, 0)^2  
filt= tukey(seq(-ws,ws)/(ws+1));filt=filt/sum(filt)  
filtTime <- as.vector(filter(cd4Data$time,filter=filt))  
filtCd4 <- as.vector(filter(cd4Data$cd4,filter=filt))  
plot(cd4Data$time,cd4Data$cd4,pch=19,cex=0.1); lines(filtTime,filtCd4,col="blue",lwd=3)
```



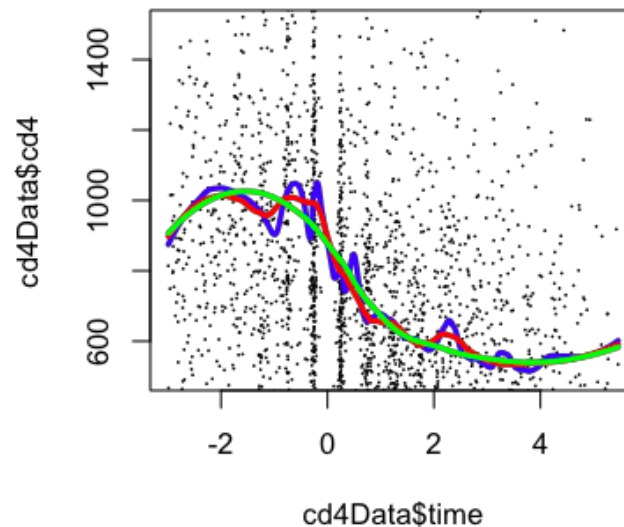
# Lowess (loess)

```
lw1 <- loess(cd4 ~ time, data=cd4Data)  
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1)  
lines(cd4Data$time, lw1$fitted, col="blue", lwd=3)
```



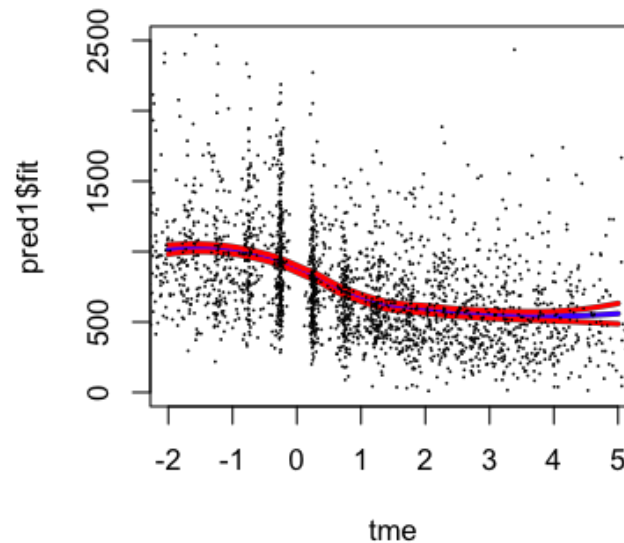
# Span

```
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1, ylim=c(500, 1500))  
lines(cd4Data$time, loess(cd4 ~ time, data=cd4Data, span=0.1)$fitted, col="blue", lwd=3)  
lines(cd4Data$time, loess(cd4 ~ time, data=cd4Data, span=0.25)$fitted, col="red", lwd=3)  
lines(cd4Data$time, loess(cd4 ~ time, data=cd4Data, span=0.76)$fitted, col="green", lwd=3)
```



# Predicting with loess

```
tme <- seq(-2,5,length=100); pred1 = predict(lw1,newdata=data.frame(time=tme),se=TRUE)
plot(tme,pred1$fit,col="blue",lwd=3,type="l",ylim=c(0,2500))
lines(tme,pred1$fit + 1.96*pred1$se.fit,col="red",lwd=3)
lines(tme,pred1$fit - 1.96*pred1$se.fit,col="red",lwd=3)
points(cd4Data$time,cd4Data$cd4,pch=19,cex=0.1)
```





# Splines

$$Y_i = b_0 + \sum_{k=1}^K b_k s_k(x_i) + e_i$$

$Y_i$  - outcome for  $i$ th observation

$b_0$  - Intercept term

$b_k$  - Coefficient for  $k$ th spline function

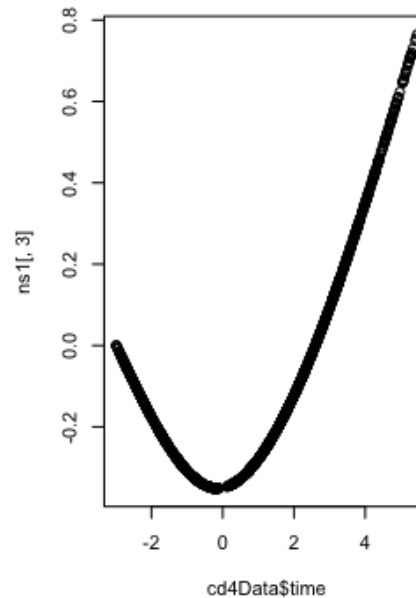
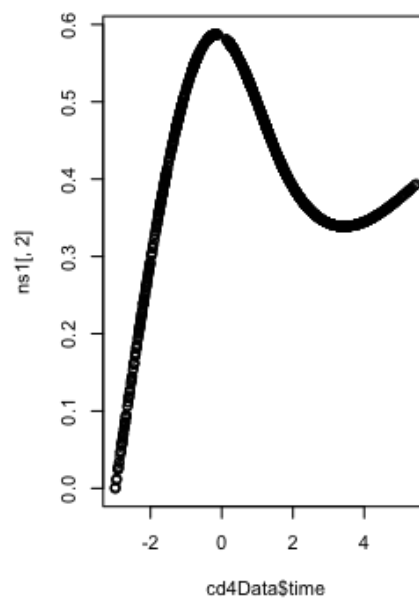
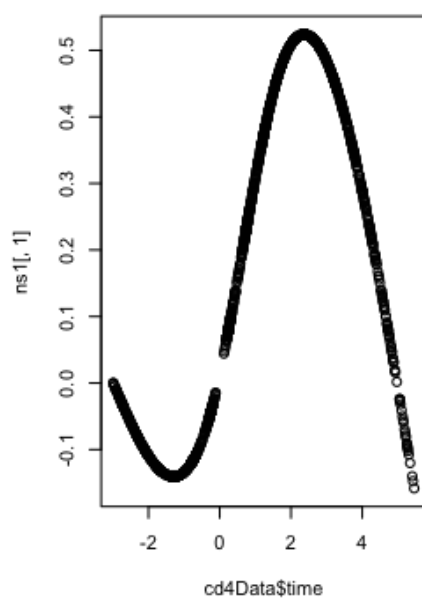
$s_k$  -  $k$ th spline function

$x_i$  - covariate for  $i$ th observation

$e_i$  - everything we didn't measure/model

# Splines in R

```
library(splines)
ns1 <- ns(cd4Data$time,df=3)
par(mfrow=c(1,3))
plot(cd4Data$time,ns1[,1]); plot(cd4Data$time,ns1[,2]); plot(cd4Data$time,ns1[,3])
```



# Regression with splines

```
lm1 <- lm(cd4Data$cd4 ~ ns1)
summary(lm1)
```

Call:

```
lm(formula = cd4Data$cd4 ~ ns1)
```

Residuals:

Min	1Q	Median	3Q	Max
-780.0	-242.4	-61.3	169.5	2263.7

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	982.0	33.9	29.01	< 2e-16 ***
ns11	-611.3	32.6	-18.78	< 2e-16 ***
ns12	-373.7	79.4	-4.71	2.6e-06 ***
ns13	-374.8	41.2	-9.09	< 2e-16 ***

---

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

Residual standard error: 357 on 2372 degrees of freedom

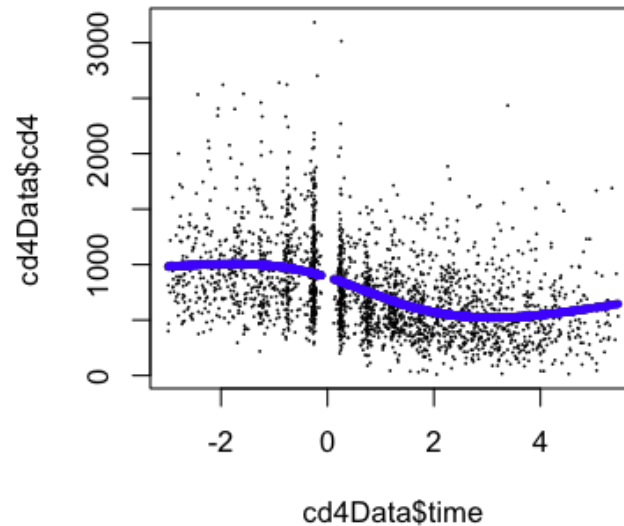
Multiple R-squared: 0.203, Adjusted R-squared: 0.202

F-statistic: 202 on 3 and 2372 DF, p-value: <2e-16

19/21

# Fitted values

```
plot(cd4Data$time, cd4Data$cd4, pch=19, cex=0.1)  
points(cd4Data$time, lm1$fitted, col="blue", pch=19, cex=0.5)
```



# Notes and further resources

## Notes:

- Cross-validation with splines/smoothing is a good idea
- Do not predict outside the range of observed data

## Further resources:

- [Hector Corrada Bravo's Lecture Notes](#)
- [Rafa Irizarry's Lecture Notes on smoothing](#), [On splines](#)
- [Elements of Statistical Learning](#)
- [Advanced Data Analysis from An Elementary Point of View](#)