Hodgepodge

Regression models

Brian Caffo, Jeff Leek, Roger Peng
Johns Hopkins Bloomberg School of Public Health

How to fit functions using linear models

- Consider a model $Y_i = f(X_i) + \epsilon$.
- How can we fit such a model using linear models (called scatterplot smoothing)
- Consider the model

gamma_k gehoert ins Summenzeichen

$$Y_i = \beta_0 + \beta_1 X_i + \sum_{k=1}^{d} (x_i - \xi_k)_+ \gamma_k + \epsilon_i$$

where $(a)_+ = a$ if $a > 0$ and 0 otherwise and $\xi_1 \leq \ldots \leq \xi_d$ are known knot points.

- Prove to yourelf that the mean function

$$\beta_0 + \beta_1 X_i + \sum_{k=1}^{d} (x_i - \xi_k)_+ \gamma_k$$

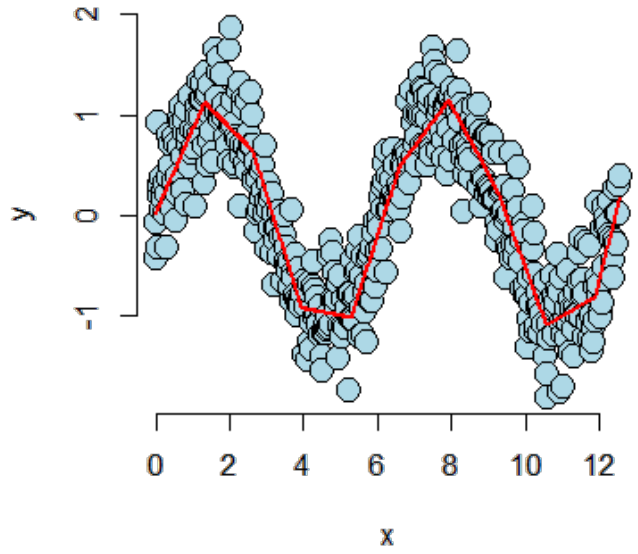
is continuous at the knot points.

zB mit einem (d=1) Knotenpunkt in gixgax_1 = 5: beta0 + beta1 * X_i + (x_i - 5) * gamma

fuer x_i <= 5: Summe wird 0, also nur beta0 + beta1 * x_i
fuer x_i > 5: … = beta0 - 5*gamma + (beta1 + gamma) * x_i
-> also wieder eine Gerade, aber andere Intercept und Steigung.
Dasselbe falls d>1: dann gibt es einfach mehrere Geraden.

Simulated example

```
n <- 500; x <- seq(0, 4 * pi, length = n); y <- sin(x) + rnorm(n, sd = .3)
knots <- seq(0, 8 * pi, length = 20);
splineTerms <- sapply(knots, function(knot) (x > knot) * (x - knot)) entspricht der Summe mit der
                                                                     (a)+ Funktion von Folie 2
xMat <- cbind(1, x, splineTerms)
yhat <- predict(lm(y ~ xMat - 1)) ohne Intercept: das ist schon in der Matrix
plot(x, y, frame = FALSE, pch = 21, bg = "lightblue", cex = 2)
lines(x, yhat, col = "red", lwd = 2)
```



Adding squared terms

- Adding squared terms makes it continuously differentiable at the knot points.
- Adding cubic terms makes it twice continuously differentiable at the knot points; etcetera.

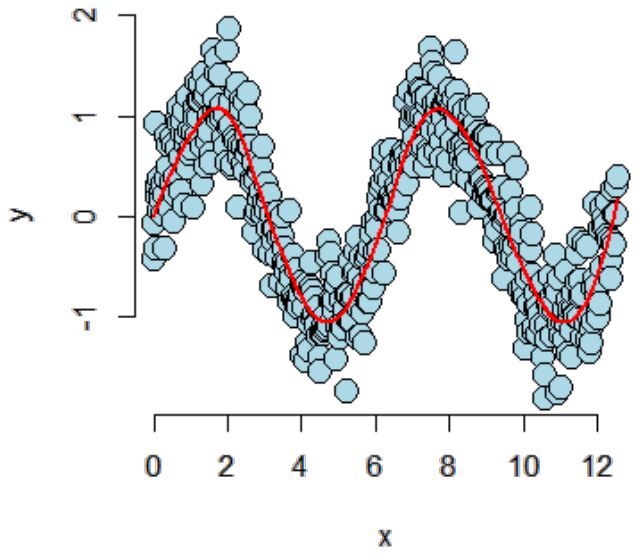
$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \sum_{k=1}^{d} (x_i - \xi_k)_+^2 \gamma_k + \epsilon_i$$

(a)_+^2 ist a^2 if a>0, sonst 0

dadurch keine Ecken mehr in der Kurve

nun wird die Kurve glatt:

```
splineTerms <- sapply(knots, function(knot) (x > knot) * (x - knot)^2)
xMat <- cbind(1, x, x^2, splineTerms)
yhat <- predict(lm(y ~ xMat - 1))
plot(x, y, frame = FALSE, pch = 21, bg = "lightblue", cex = 2)
lines(x, yhat, col = "red", lwd = 2)
```



Notes

- The collection of regressors is called a basis.
  - People have spent **a lot** of time thinking about bases for this kind of problem. So, consider this as just a teaser.
- Single knot point terms can fit hockey stick like processes.
- These bases can be used in GLMs as well.
- An issue with these approaches is the large number of parameters introduced.

```
-> mindestens d
   Parameter
```

  - Requires some method of so called regularization.

Harmonics using linear models

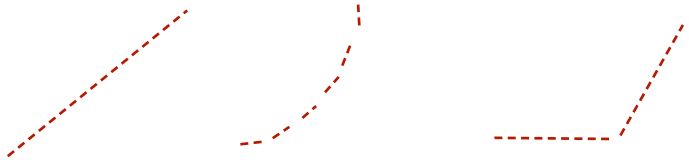
```
##Chord finder, playing the white keys on a piano from octave c4 - c5
notes4 <- c(261.63, 293.66, 329.63, 349.23, 392.00, 440.00, 493.88, 523.25) <- BASIS ?
t <- seq(0, 2, by = .001); n <- length(t) Alle Toene werden gleichzeitig fuer 2 Sek gespielt.
c4 <- sin(2 * pi * notes4[1] * t); e4 <- sin(2 * pi * notes4[3] * t); Frequenzen fuer c4, e4, g4
g4 <- sin(2 * pi * notes4[5] * t)
chord <- c4 + e4 + g4 + rnorm(n, 0, 0.3)  3 Toene zusammen sind ein Chord. + some noice
x <- sapply(notes4, function(freq) sin(2 * pi * freq * t))
fit <- lm(chord ~ x - 1)
```

c4 c5

Alle Toene werden gleichzeitig fuer 2 Sek gespielt.

Ziel: Herausfinden, welcher Akkord gespielt wird

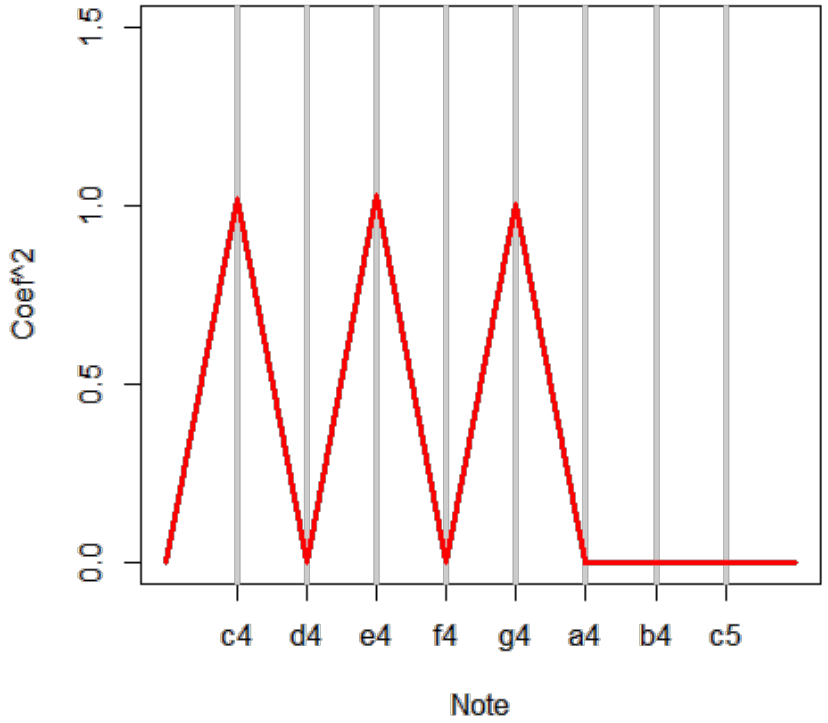
Basis, zB:



Linie Quadratisch Spline

Beruehmte Basis: Sinus und Kosinus

Und tatsaechlich, es schlaegt nur bei den “gespielten” Noten aus (chord): c4, e4, g4



```
##(How you would really do it)
a <- fft(chord); plot(Re(a)^2, type = "l")
```

fast fourier transform