Day 7: Polynomial Models

Lucas Leemann

Essex Summer School

Introduction to Statistical Learning

1. Motivation and Overview
2. Polynomials
3. Step Functions
4. Splines
5. Local Regression
6. Generalized Additive Model

So Far

- Yesterday: Constrained linear models to reduce complexity (reducing variance)
- Today: More complex relationships between outcome and predictor: polynomials, splines, GAM.

Linearity and social reality

- Social world is almost never linear.
- Often the linearity assumption is good enough.
- When linearity doesn't hold we can use
  - polynomials,
  - step functions,
  - splines,
  - local regression, and
  - generalized additive models
- These models offer a lot of flexibility, without losing too much interpretability.

Today

- **Polynomials:** Include higher order X's to increase flexibility:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_1^3 + ... + \beta_j X_1^j + \varepsilon$$

- **Splines:** A more flexible form of polynomial regression, where:

$$Y = \begin{cases} \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_1^3 + ... + \beta_j X_1^j + \varepsilon & \text{if } X_1 < c, \\ \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_1^3 + ... + \beta_j X_1^j + \varepsilon & \text{if } X_1 > c. \end{cases}$$

- **GAM:** Generalized additive model, we have various functional forms for the individual variables:

$$Y = \beta_0 + f(X_1) + f(X_2) + f(X_3) + ... + f(X_j) + \varepsilon$$

Polynomials

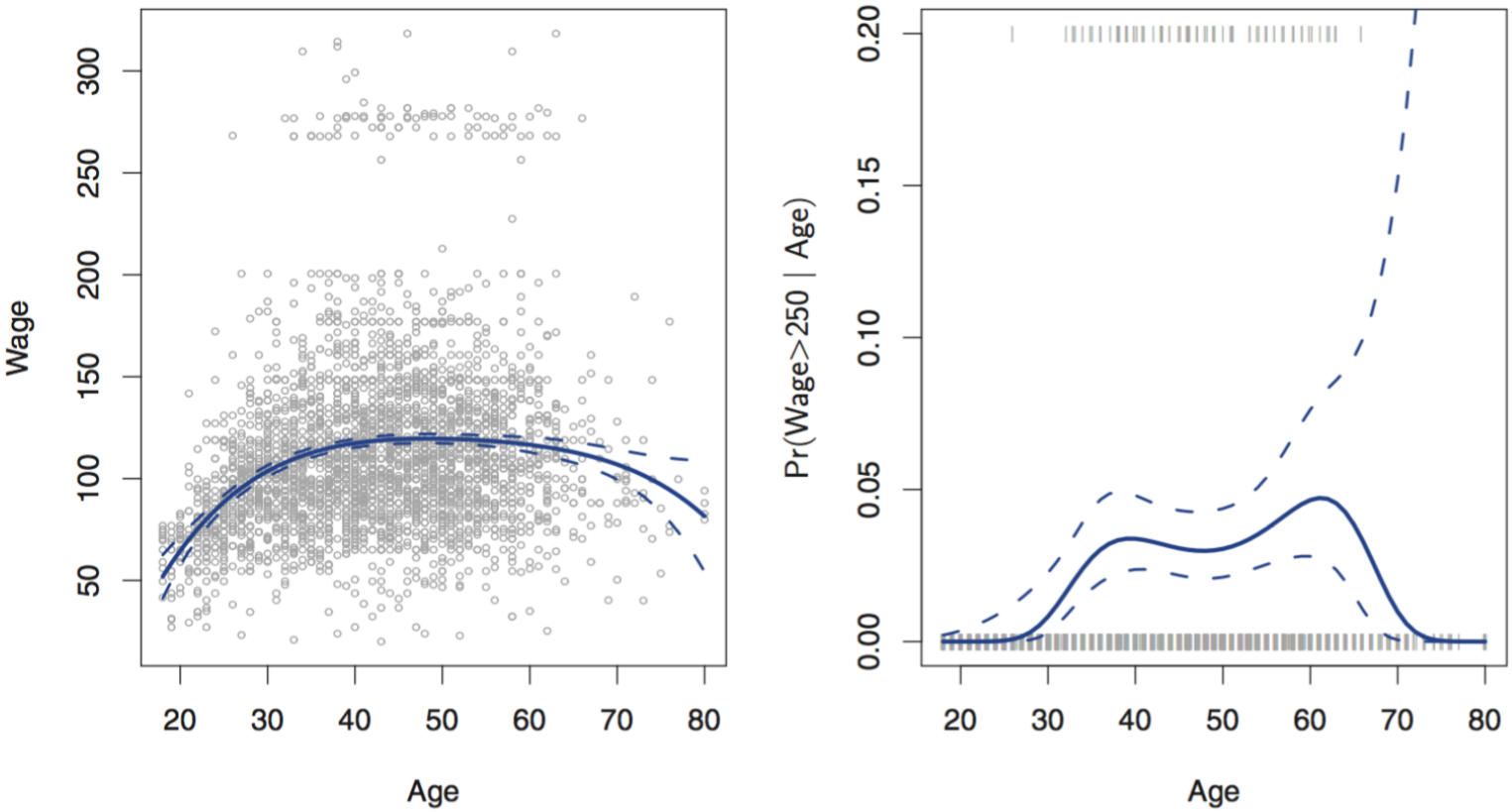
Polynomials

- Adding higher ordered X's allows to increase the flexibility of a model.
- While in general:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_1^3 + ... + \beta_j X_1^j + \varepsilon$$

  we usually do not go beyond the forth degree.
- We are hoping to reduce *bias* at the cost of *variance*.
- Can also be applied to classification (e.g. logistic regression).

Polynomial and Data Fit (d=4)



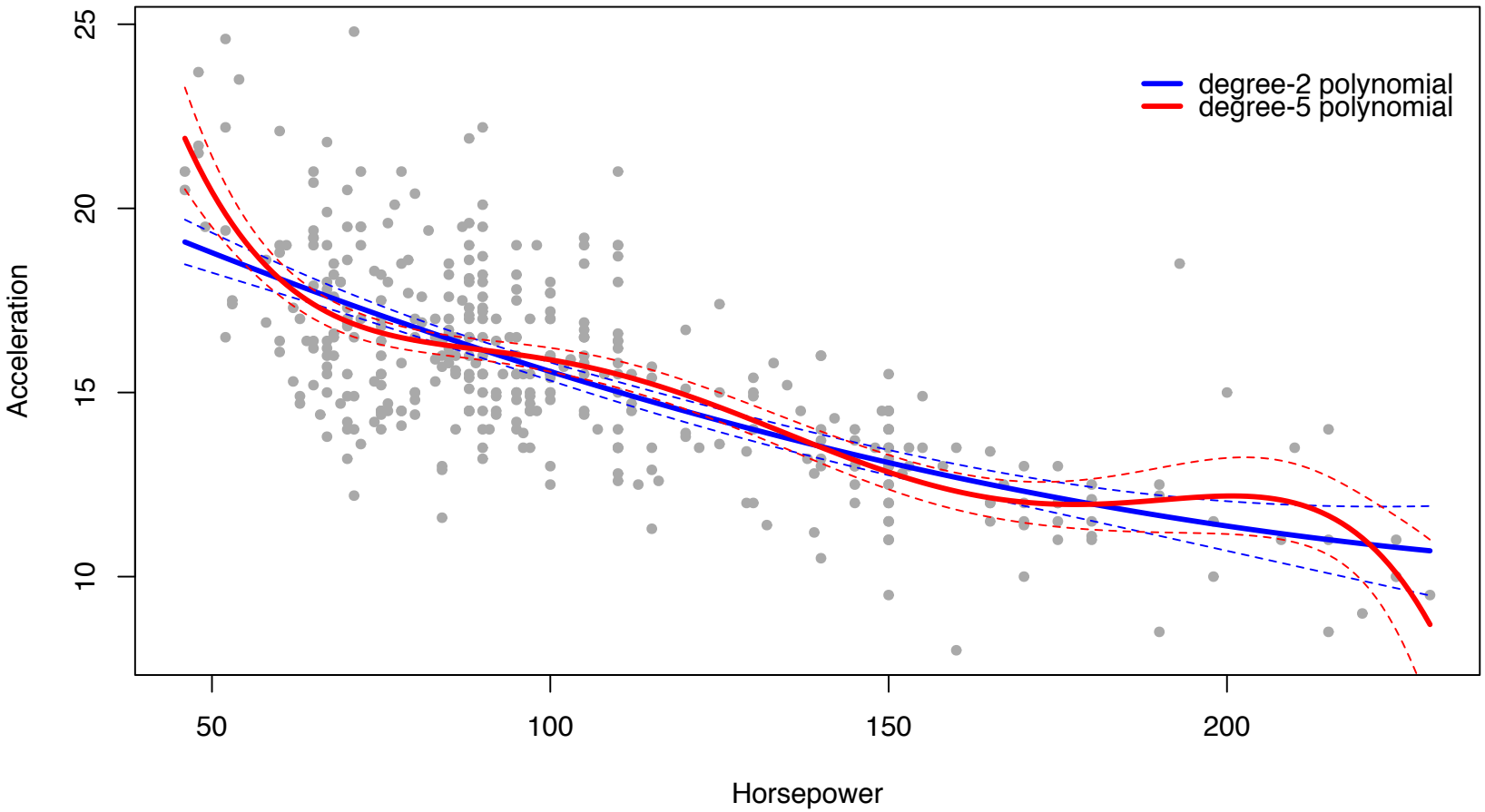
(Hastie et al, 2008: 267)

Example: Polynomial and `Auto` Dataset

Acceleration $\approx f$(horsepower)

```
> fit.1=lm(acceleration~horsepower,data=Auto)
> fit.2=lm(acceleration~poly(horsepower,2),data=Auto)
> fit.3=lm(acceleration~poly(horsepower,3),data=Auto)
> fit.4=lm(acceleration~poly(horsepower,4),data=Auto)
> fit.5=lm(acceleration~poly(horsepower,5),data=Auto)
> anova(fit.1,fit.2,fit.3,fit.4,fit.5)
Analysis of Variance Table

Model 1: acceleration ~ horsepower
Model 2: acceleration ~ poly(horsepower, 2)
Model 3: acceleration ~ poly(horsepower, 3)
Model 4: acceleration ~ poly(horsepower, 4)
Model 5: acceleration ~ poly(horsepower, 5)
  Res.Df    RSS Df Sum of Sq       F    Pr(>F)
1    390 1562.4
2    389 1533.4  1    29.022  7.7671  0.005583 **
3    388 1529.2  1     4.270  1.1428  0.285727
4    387 1511.8  1    17.387  4.6531  0.031614 *
5    386 1442.3  1    69.454 18.5877 2.063e-05 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

Example: Polynomial and `Auto` Dataset 2



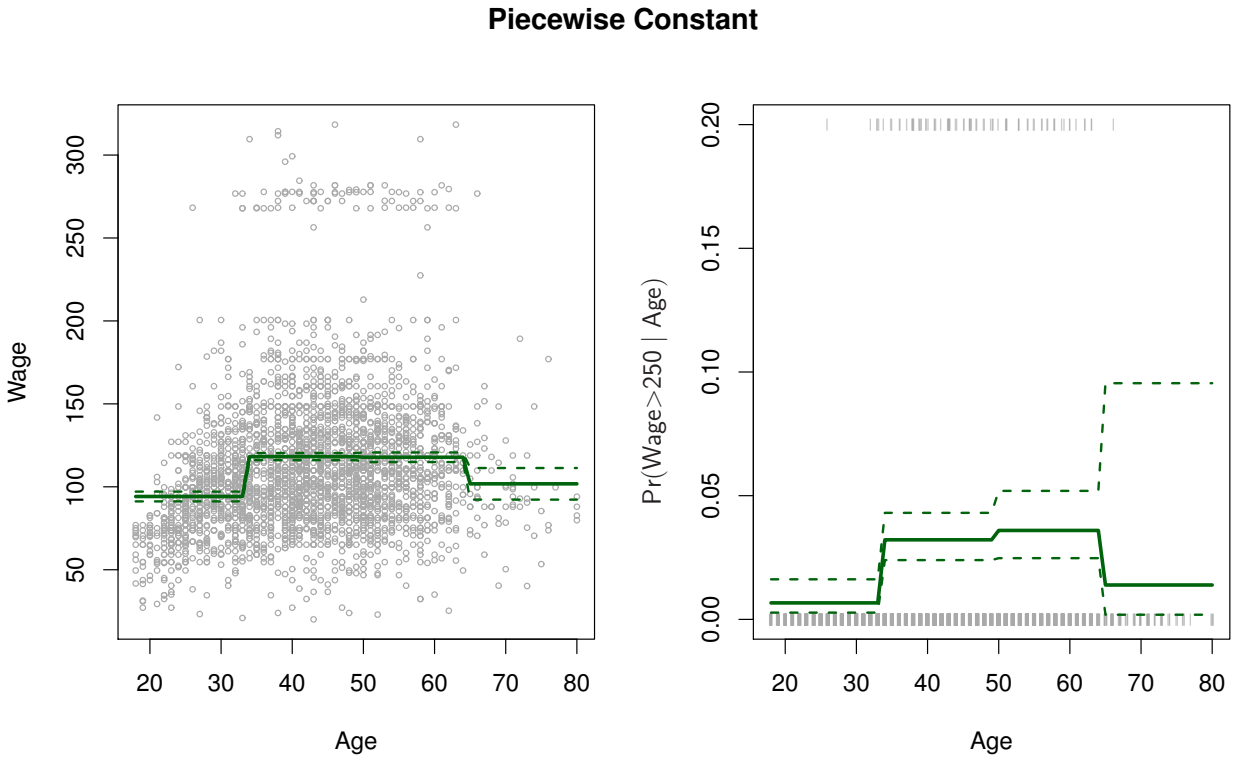
(code for figure)

```
par(mfrow=c(1,1))
plot(Auto$horsepower,Auto$acceleration, ylab="Acceleration",
               xlab="Horsepower", pch=19, col="darkgrey", cex=0.7)
# fit.2
hsplims=range(horsepower)
hsp.grid=seq(from=hsplims[1],to=hsplims[2])
preds=predict(fit.2,newdata=list(horsepower=hsp.grid),se=TRUE)
se.bands=cbind(preds$fit+2*preds$se.fit,preds$fit-2*preds$se.fit)
lines(hsp.grid,preds$fit,col="blue", lwd=3)
lines(hsp.grid,se.bands[,1],col="blue", lwd=1, lty=2)
lines(hsp.grid,se.bands[,2],col="blue", lwd=1, lty=2)

# fit.5
hsplims=range(horsepower)
hsp.grid=seq(from=hsplims[1],to=hsplims[2])
preds=predict(fit.5,newdata=list(horsepower=hsp.grid),se=TRUE)
se.bands=cbind(preds$fit+2*preds$se.fit,preds$fit-2*preds$se.fit)
lines(hsp.grid,preds$fit,col="red", lwd=3)
lines(hsp.grid,se.bands[,1],col="red", lwd=1, lty=2)
lines(hsp.grid,se.bands[,2],col="red", lwd=1, lty=2)

legend(185,24,c("degree-2 polynomial","degree-5 polynomial"), lty=1, col=c("blue", "red"), lwd=3, bty="n")
```

Step Functions

Another way of creating transformations of a variable – cut the variable into distinct regions.

$$C_1(X) = I(X < 35), C_2(X) = I(35 \leq X < 50), \ldots, C_3(X) = I(X \geq 65)$$



Step functions continued

- Easy to work with. Creates a series of dummy variables representing each group.
- Useful way of creating interactions that are easy to interpret. For example, interaction effect of *Year* and *Age*:

$$I(\mathit{Year} < 2005) \cdot \mathit{Age}, \;\; I(\mathit{Year} \geq 2005) \cdot \mathit{Age}$$

  would allow for different linear functions in each age category.
- In R: $I(year < 2005)$ or $cut(age, c(18, 25, 40, 65, 90))$.
- Choice of cutpoints or knots can be problematic. For creating nonlinearities, smoother alternatives such as splines are available.

Splines

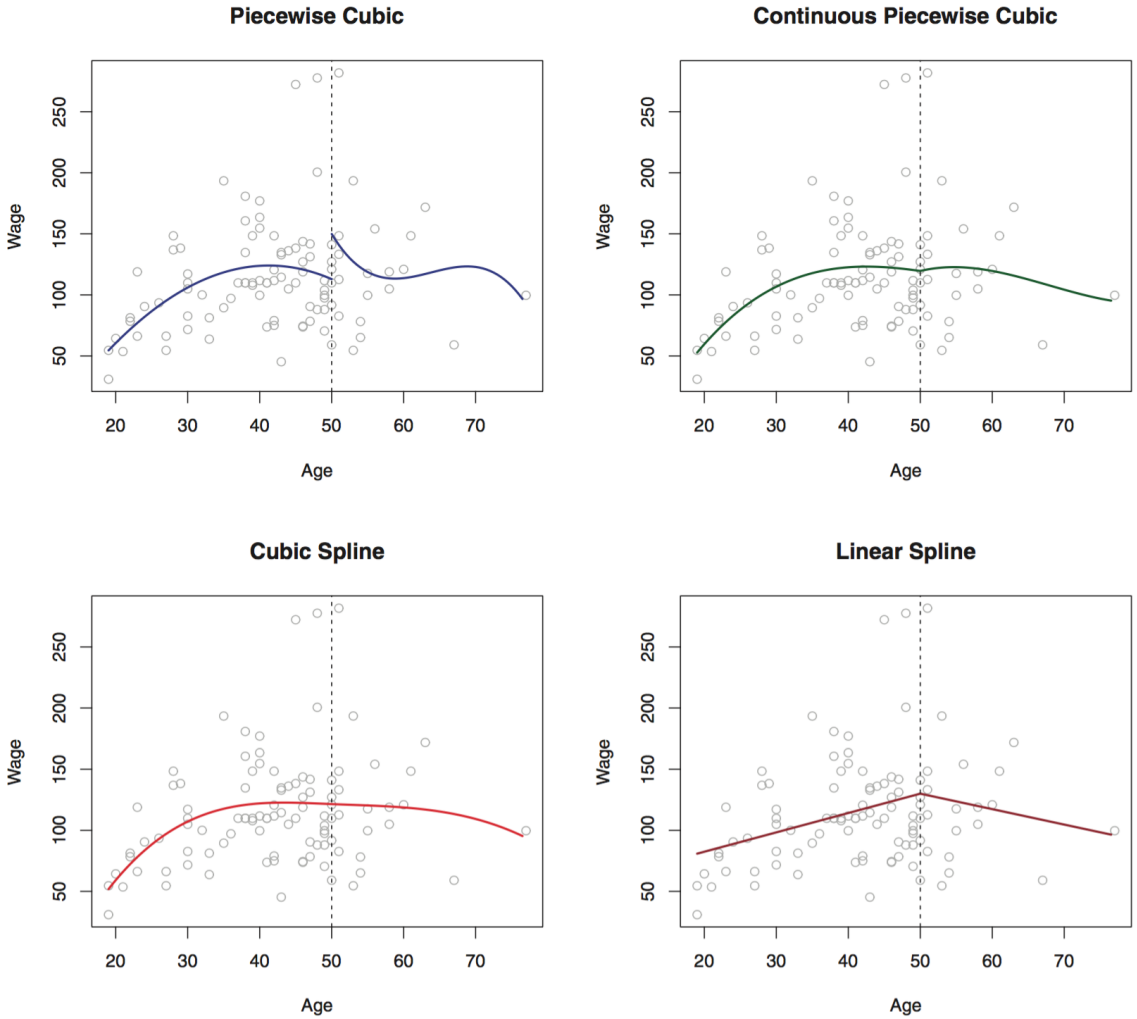
Splines 1 (piecewise polynomials)

- We want to avoid a global function and rather split the range of X in different areas.
- In each area we fit a separat polynomial leading to a very flexible function.
- This is known as piecewise polynomials:

$$Y = \begin{cases} \beta_0 + \beta_1^a X_1 + \beta_2^a X_1^2 + \beta_3^a X_1^3 + ... + \beta_j^a X_1^j + \varepsilon & \text{if } X_1 < c, \\ \beta_0 + \beta_1^b X_1 + \beta_2^b X_1^2 + \beta_3^b X_1^3 + ... + \beta_j^b X_1^j + \varepsilon & \text{if } X_1 > c. \end{cases}$$

- We call c a knot and the more knots, the more parameters to estimate.

Splines 2 (piecewise polynomials)



(Hastie et al, 2008: 272)

Splines 3 (constrained splines)

- We want to have a continuous function and no *breaks* at the knot(s).
- One way to achieve that is to include constraints, s.t. that the function has to be continuous (ur) or also that the first derivative(s) has to be continuous (lr) to create a smooth function.
- degree-d spline: Is a spline with that a degree-d polynomial between two knots and is continuous up to the $d-1$ derivative (squared spline is continuous at first derivative).

Splines 4 (degree-d spline)

- How do we get these splines to be continuous up to the $d-1$th derivative?
- It turns out that we can just create a function with $K+4$ parameters (cubic spline), s.t.

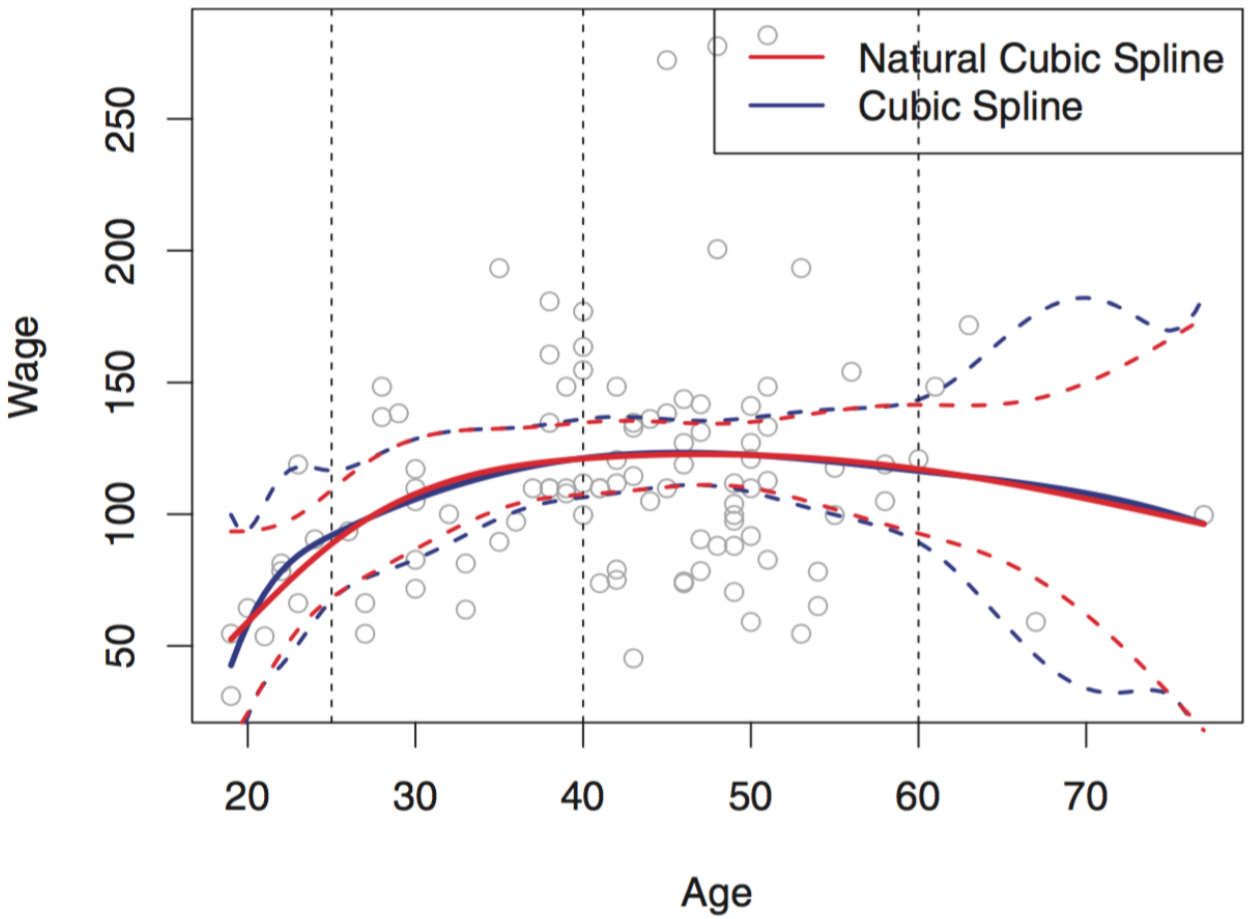
$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 h(x, \xi_1) ... + \beta_{K+3} h(x, \xi_K) + \varepsilon$$

- whereas

$$h(x, \xi_k) = (x - \xi_k)_+^3 = \begin{cases} (x - \xi_k)^3 \text{ if } x > \xi_k, \\ 0 \text{ if } x < \xi_k. \end{cases}$$

- This will guarantee that the resulting spline function is continous.
- High variance at boundaries of X, but can add constraint, s.t. first & last part are linear (a.k.a. *natural cubic spline*)

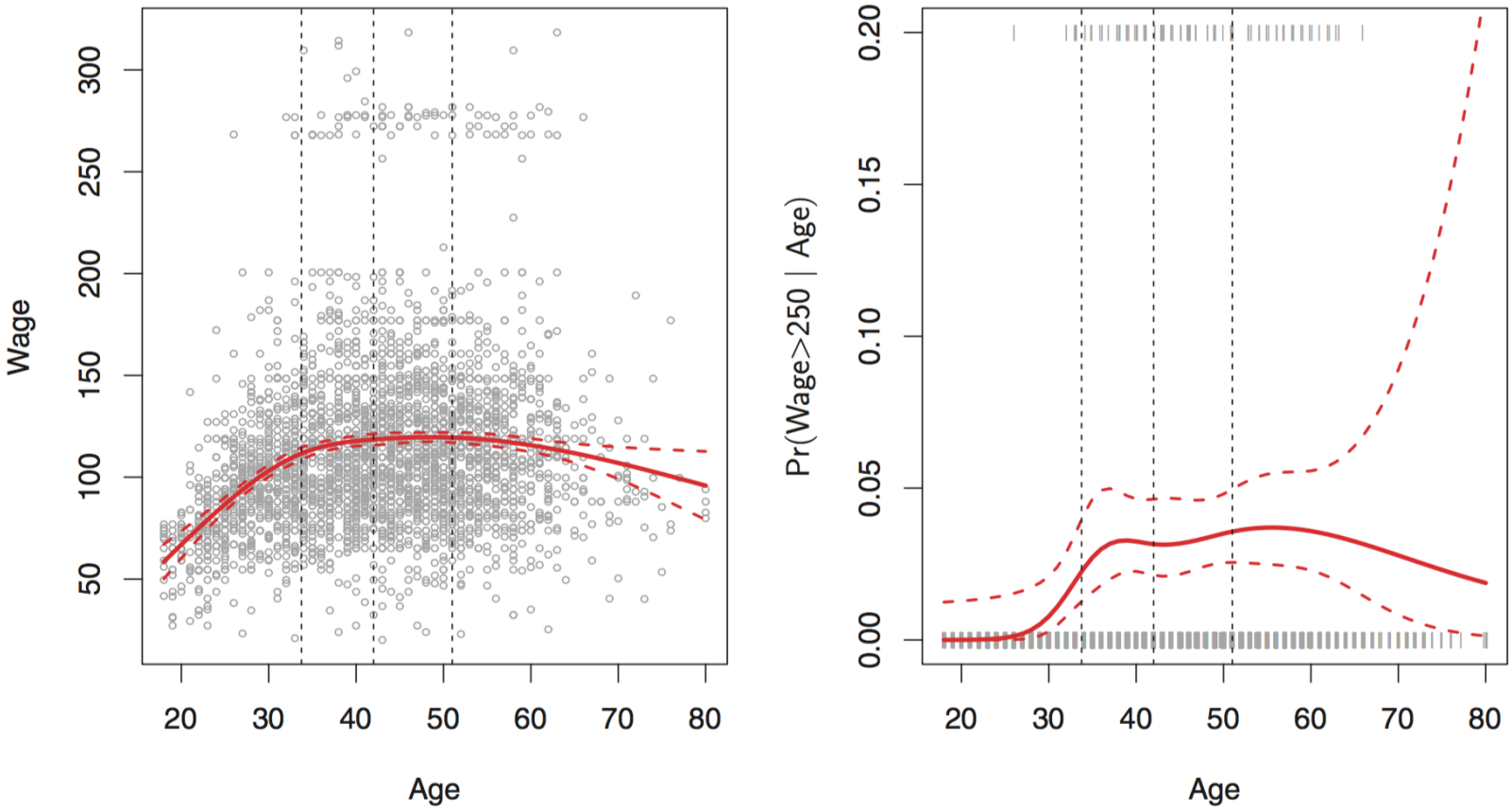
Splines 5 (natural cubic splines)



(Hastie et al, 2008: 274)

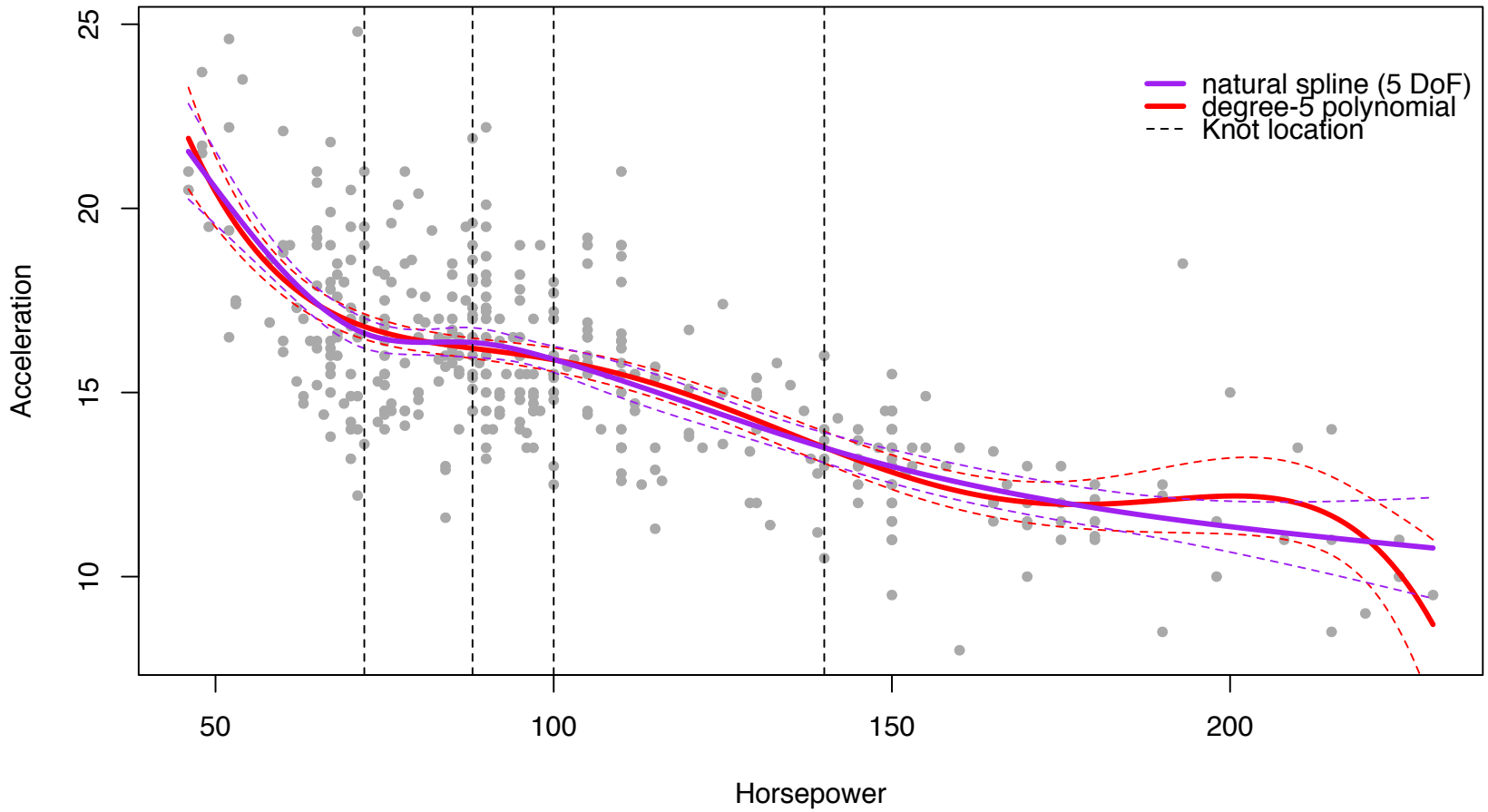
Splines 6 (Where to set the knots?)

We can set the knots at equal intervals (quantiles), e.g. three knots at 25th, 50th, and 75th percentile:



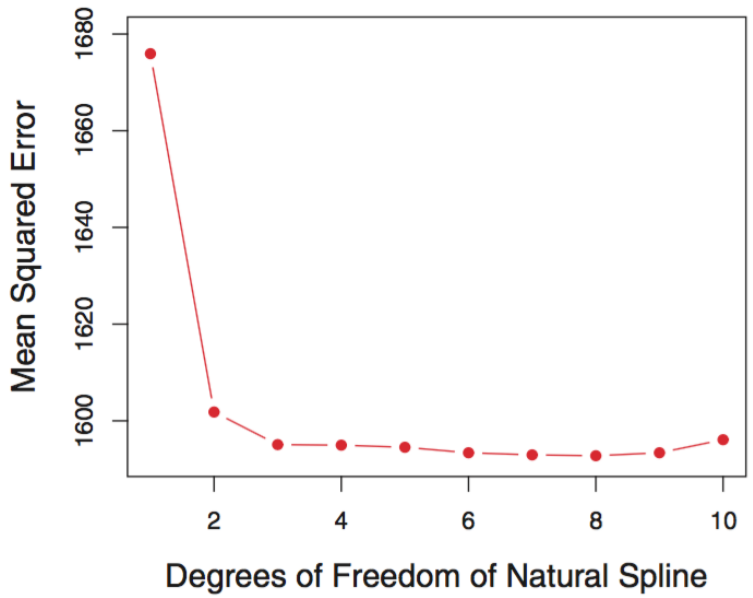
(Hastie et al, 2008: 275)

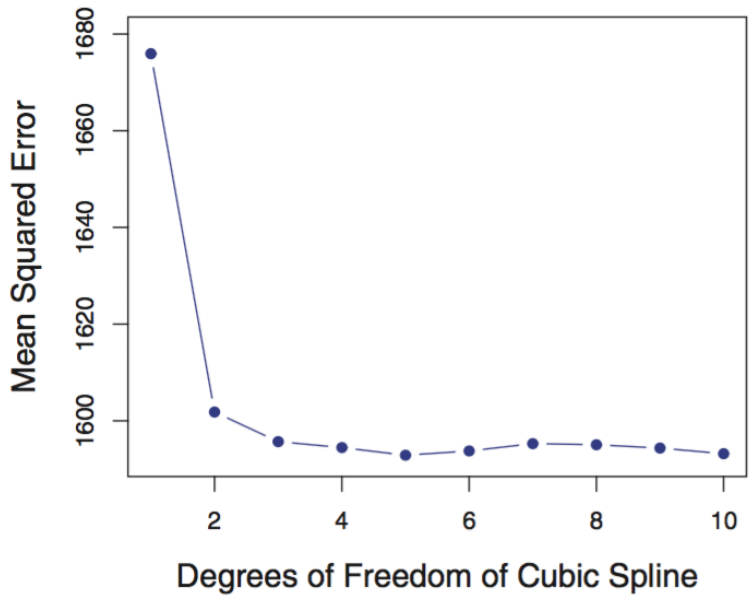
Splines 7 (Example with `Auto` dataset)



Splines 8 (wrapping up)

- There are many more possibilities (smoothing splines, see 7.5)
- How do we choose the number of knots? Cross-validation can help!





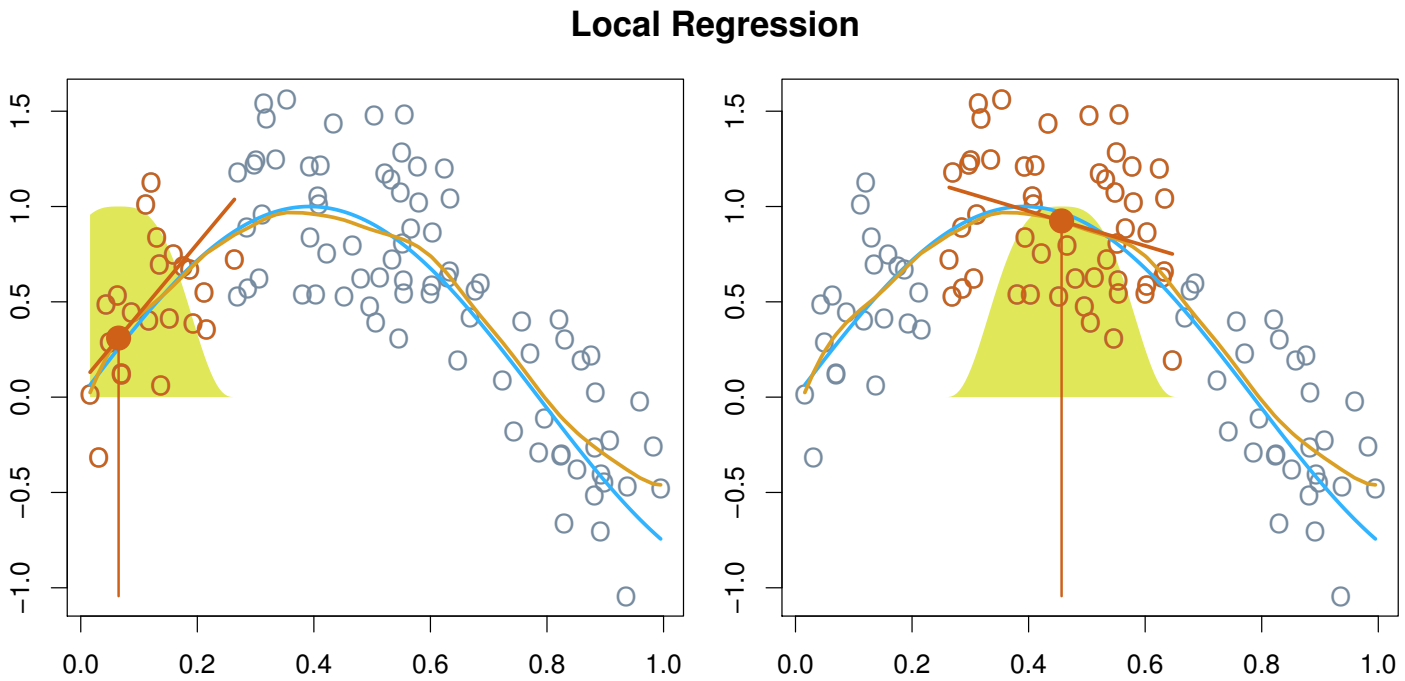
(Hastie et al, 2008: 275)

Polynomial vs Splines

- Natural splines often outperform polynomials.
- NS equally flexible as P, but more stable.
- Advantage comes through the setting of knots.[1]
- CV can help explore the right number of DoF.

[1] With the R command `ns(x, df=d)` you set the number of knots at $d - 1$.

Local regression



- With a sliding weight function, we fit separate linear fits over the range of X by weighted least squares.
- If $p \neq 1$ we need many training observations since there are many neighborhoods.

Generalized Additive Model

Generalized Additive Models (GAM)

- Solutions so far for one X, but we often have several predictors.
- We will extend the classic regression,

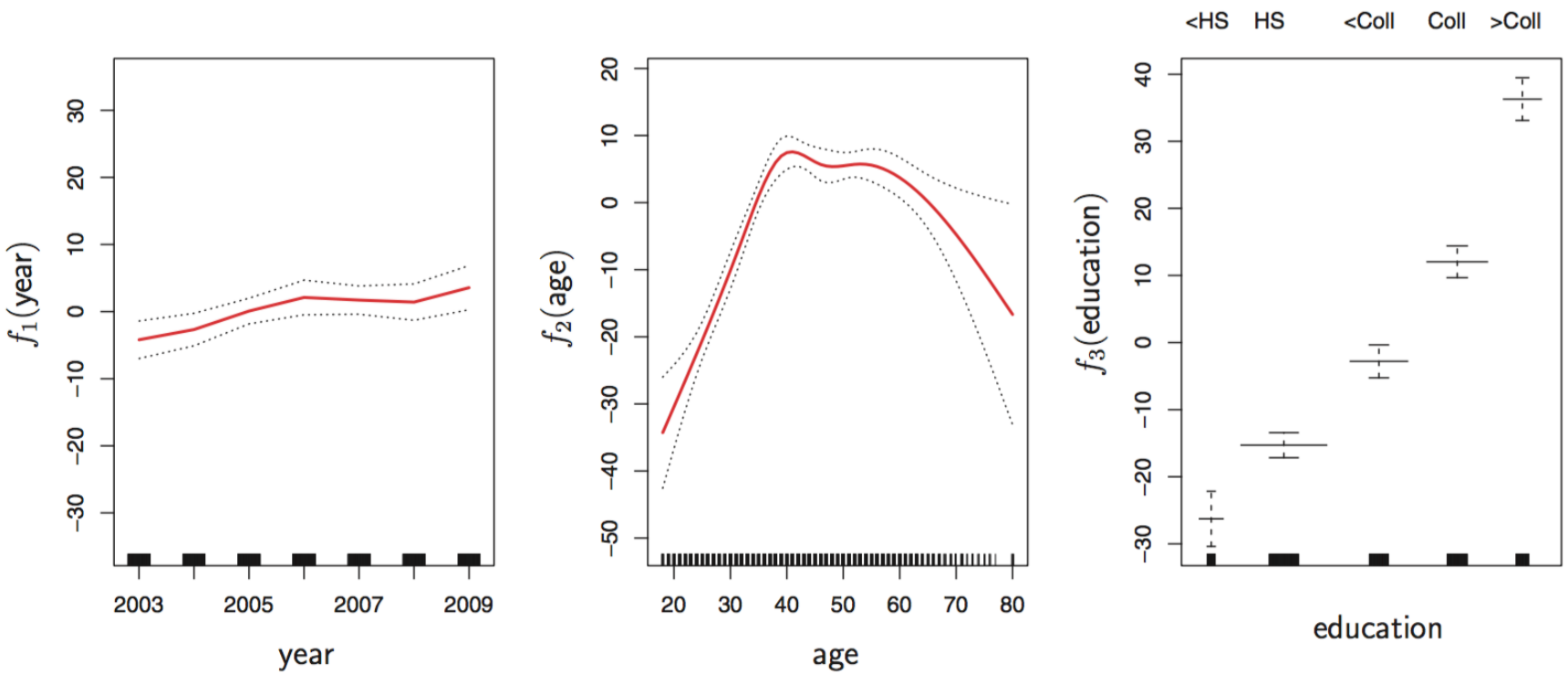
$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_j X_{ji} + \varepsilon_i$$

to be like this:

$$Y_i = \beta_0 + f_1(X_{1i}) + f_2(X_{2i}) + \cdots + f_J(X_{Ji}) + \varepsilon_i$$

- we will rely on splines or polynomials for $f_j(\cdot)$.

GAM 2



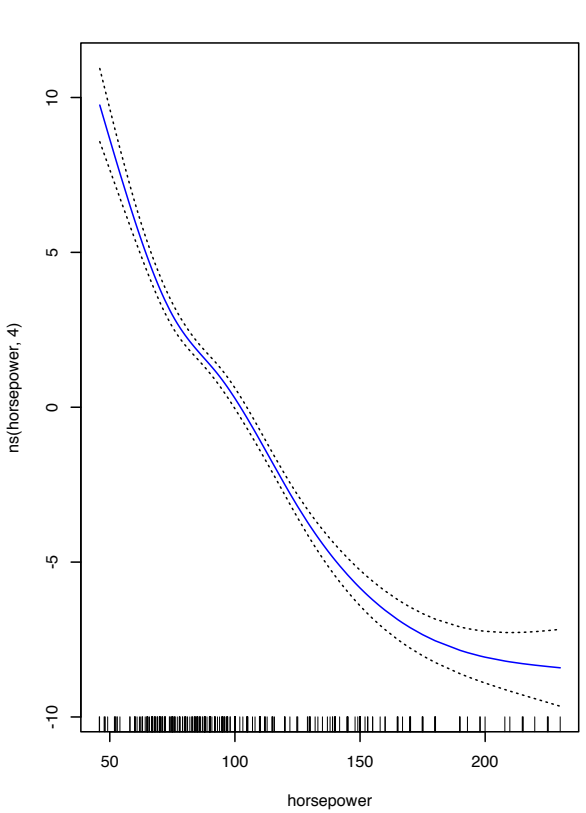
(Hastie et al, 2008: 284)

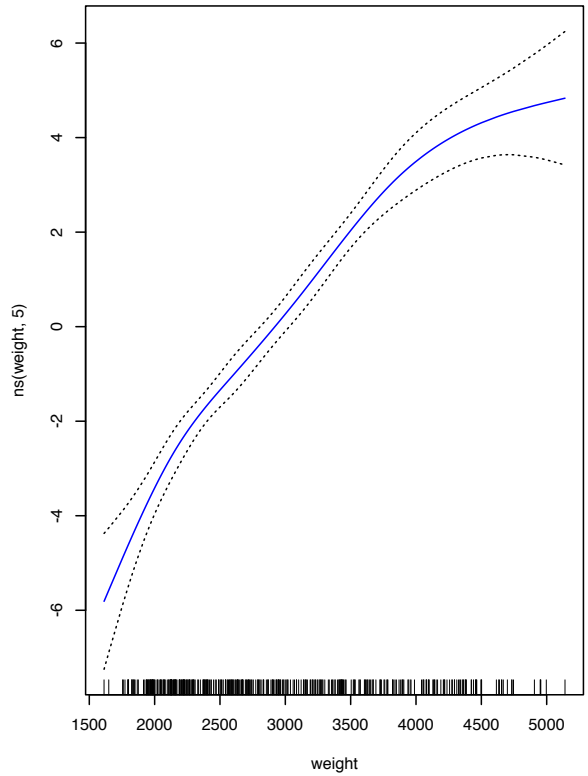
Note: $f_1()$ and $f_2()$ are natural splines with 4(5) DoF, while $f_3()$ is a step function (dummies).

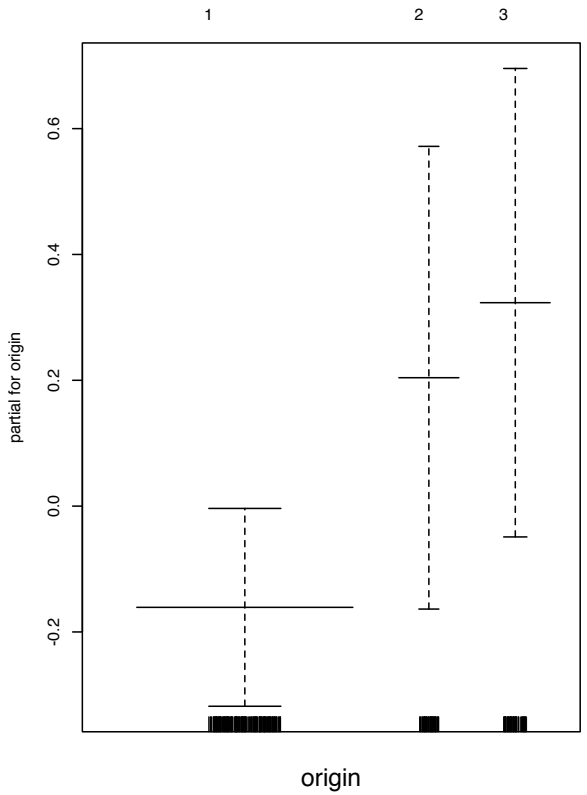
(Dis-)Advantages of GAM

+ GAMs allow to model non-linear relationships with several variables. Non-linear fit can be much better than linear approximation.
+ Because GAMs are additive, we can *control* for other factors and include all information.
- We cannot model interactions, but only have additivity.

Example: `Auto` Dataset







Lab

- Polynomial models and CV
- Natural splines and CV
- Putting it all together: GAM