Chapter 7: Moving beyond linearity

Yu-Tzung Chang and Hsuan-Wei Lee

Department of Political Science, National Taiwan University

2018.12.06.

Outline

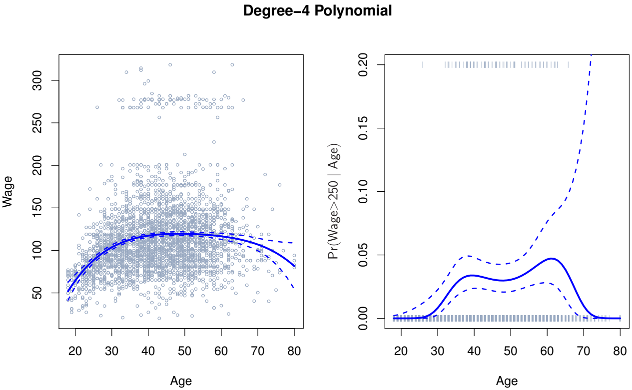
- Polynomial regression
- Step functions
- Basis functions
- Regression splines
- Smoothing splines
- Local regression
- Generalized additive splines

Moving beyond nonlinearity

- The real world is never linear! Or almost never!
- But often the linearity assumption is good enough.
- When it's not linear, we can use
  - polynomials
  - step functions
  - splines
  - local regression
  - generalized additive models
- The models above offer a lot of flexibility, without losing the ease and interpretability of linear models.

Polynomial regression

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_d x_i^d + \epsilon_i$$



Details of polynomial regression

- Create new variables $X_1 = X$, $X_2 = X^2$, etc, and then treat as multiple linear regression.
- Not really interested in the coefficients; more interested in the fitted function values at any value x_0:

$$\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0 + \hat{\beta}_2 x_0^2 + \hat{\beta}_3 x_0^3 + \hat{\beta}_4 x_0^4.$$

- Since $\hat{f}(x_0)$ is a linear function of the $\hat{\beta}_l$, we can get a simple expression for *pointwise-variances* $Var[\hat{f}(x_0)]$ at any value x_0. In the figure before we have computed the fit and pointwise standard errors on a grid of values for x_0. We show $\hat{f}(x_0) \pm 2 \cdot se[\hat{f}(x_0)]$.

Details of polynomial regression

- Logistic regression follows naturally. For example, in the previous figure we model

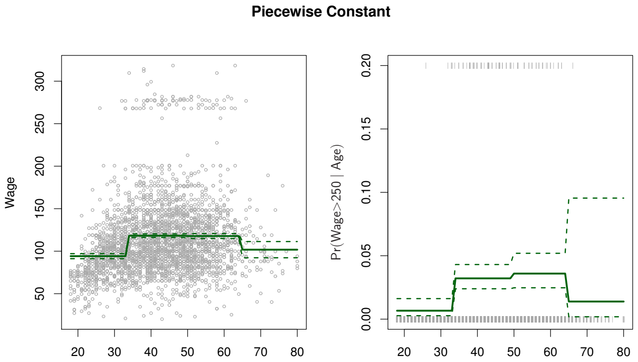
$$Pr(y_i > 250|x_i) = \frac{\exp(\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_d x_i^d)}{1 + \exp(\beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \cdots + \beta_d x_i^d)}.$$

- To get confidence intervals, compute upper and lower bounds *on the logit scale*, and then invert to get on probability scale.
- Can do separately on several variables—just stack the variables into one matrix, and separate out the pieces afterwards (see GAM later).
- Caveat: polynomials have notorious tail behavior—very bad for extrapolation.
- Can fit using $y \sim poly(x, degree = 3)$ in formula.

Step functions

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

- $C_1(X) = I(X < 35)$
- $C_2(X) = I(35 \leqslant X < 50)$
- $C_3(X) = I(50 \leqslant X < 65)$
- $C_4(X) = I(X \geqslant 65)$



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*:

$$I(\mathit{year} < 2005) \cdot \mathit{age}, \;\; I(\mathit{year} \geqslant 2005) \cdot \mathit{age}$$

  would allow for different linear functions in each age category.
- In R: use the *cut* function
  e.g. *cut(age, c(10,20,30,40,50,60,70))*.

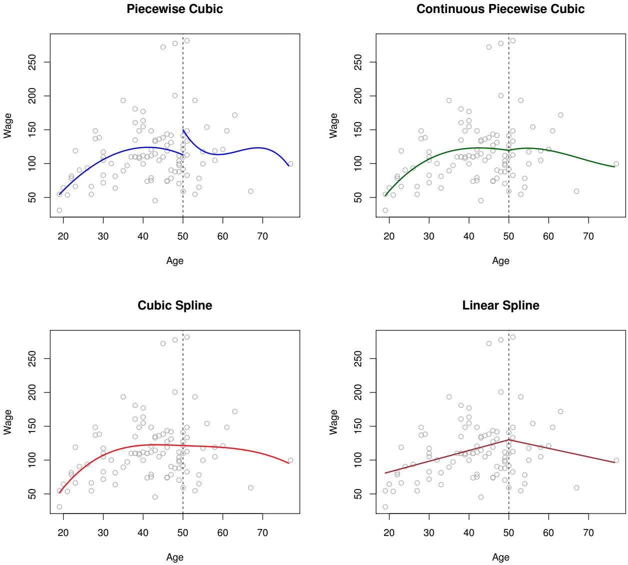
Piecewise polynomials

- Instead of a single polynomial in X over its whole domain, we can rather use different polynomials in regions defined by knots.

$$\alpha(x) = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \epsilon_i \text{ if } x_i < c; \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \epsilon_i \text{ if } x_i \geqslant c. \end{cases}$$

- Better to add constraints to the polynomials, e.g. continuity.
- *Splines* have the "maximum" amount of continuity.

Piecewise polynomials and splines



Linear splines

- A linear spline with knots at $\xi_k, k = 1, \ldots, K$ is a piecewise linear polynomial continuous at each knot.
- We can represent this model as

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) \cdots + \beta_{K+1} b_{K+1}(x_i) + \epsilon_i,$$

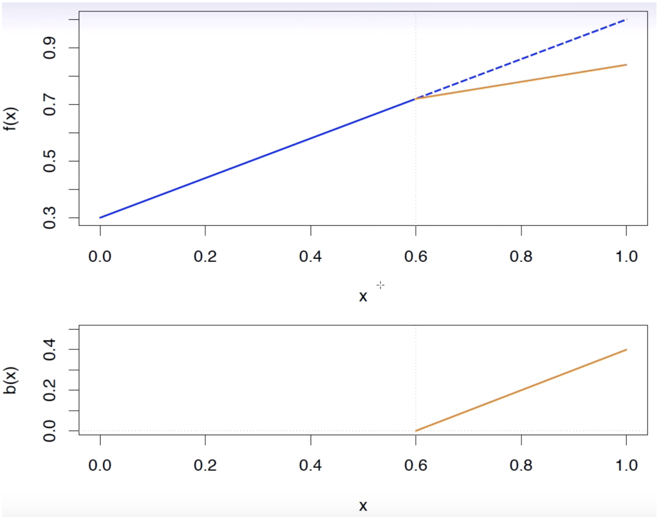
where the b_k are *basis functions*,

$$b_1(x_i) = x_i$$
$$b_{k+1}(x_i) = (x_i - \xi_k)_+, \; k = 1, 2, \ldots, K$$

- Here the $()_+$ means *positive part*; i.e.

$$(x_i - \xi_k)_+ = \begin{cases} x_i - \xi_k & \text{if } x_i > \xi_k; \\ 0 & \text{otherwise} \end{cases}$$

Linear splines: illustration



Cubic splines

- A cubic splines with knots at $\xi_k, k = 1, 2, \ldots, K$ is a piecewise cubic polynomial with continuous derivatives up to order 2 at each knot.
- Again we can represent this model truncated power basis functions

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) \cdots + \beta_{K+3} b_{K+3}(x_i) + \epsilon_i,$$

$$b_1(x_i) = x_i$$

$$b_2(x_i) = x_i^2$$

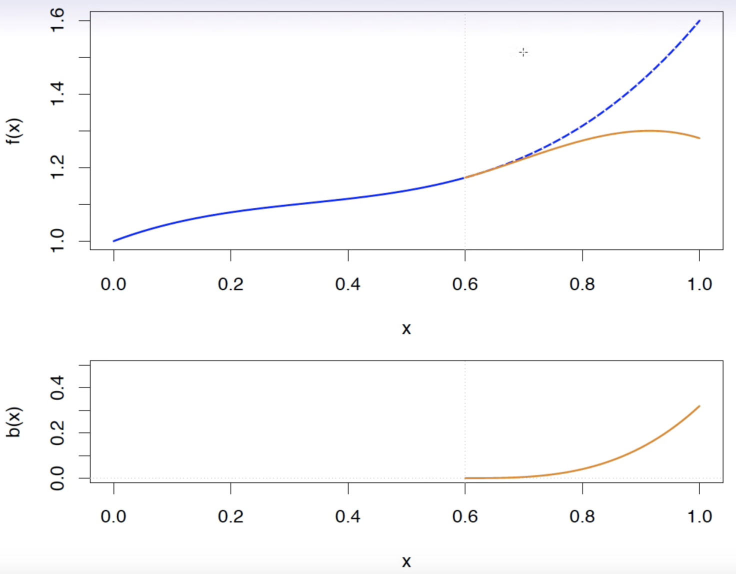
$$b_3(x_i) = x_i^3$$

$$b_{k+3}(x_i) = (x_i - \xi_k)_+^3, \;\; k = 1, 2, \ldots, K$$

where

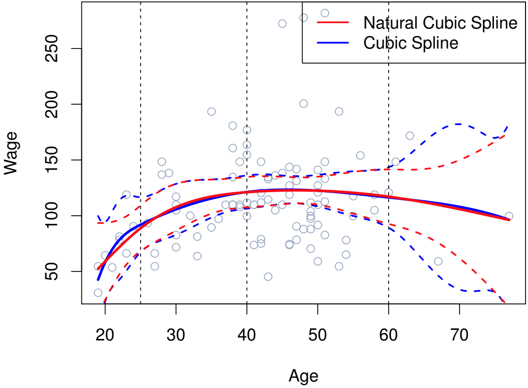
$$(x_i - \xi_k)_+^3 = \begin{cases} (x_i - \xi_k)^3 & \text{if } x_i > \xi_k; \\ 0 & \text{otherwise} \end{cases}$$

Cubic splines: illustration



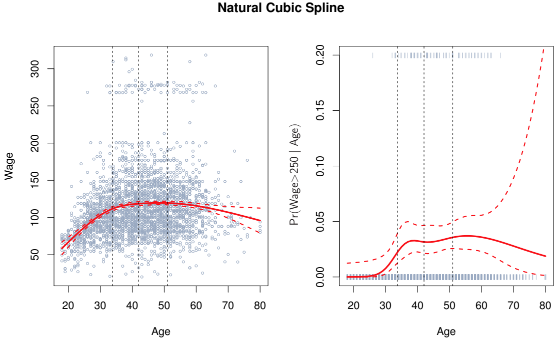
Natural cubic splines

A natural cubic splines extrapolates linearly beyond the boundary knots. This adds $4 = 2 \times 2$ extra constraints, and allows us to put more internal knots for the same degrees of freedom as a regular cubic spline.



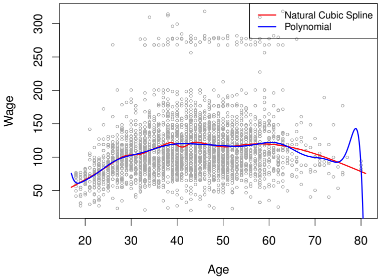
Fitting splines

Fitting splines in R is easy: *bs(x, ...)* for any degree of splines and *ns(x, ...)* for natural cubic splines, in the package *splines*.



Knot placement

- One strategy is to decide K, the number of knots, and then place them at appropriate quantiles of the observed X.
- A cubic spline with K knots has $K + 4$ parameters or degrees of freedom.
- A natural spline with K knots has K degrees of freedom.
- Comparison of a degree-14 polynomial and a natural cubic spline, each with 15 degrees of freedom.



Smoothing splines

This section is a little bit mathematical.
Consider this criterion for fitting a smooth function $g(x)$ to some data:

$$\text{minimize}_{g \in S} \sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt.$$

- The first term is RSS, and tries to make $g(x)$ match the data at each x_i.
- The second term is a *roughness penalty* and controls how wiggly $g(x)$ is. It is modulated by the *tuning parameter λ*.
  - $\lambda \geqslant 0$
  - The smaller λ, the more wiggly the function, eventually interpolating y_i when $\lambda = 0$.
  - As $\lambda \to \infty$, the function $g(x)$ becomes linear.

Smoothing splines (continued)

The solution is a natural cubic spline, with a knot at every unique value of x_i. The roughness penalty still controls the roughness via λ.

Some details

- Smoothing splines avoid the knot-selection issue, leaving a single λ to be chosen.
- The algorithmic details are too complex to describe here. In R, the function *smooth.spline()* will fit a smoothing spline.
- The vector of n fitted values can be written as $\hat{g}_\lambda = S_\lambda y$, where S_λ is a $n \times n$ matrix (determined by the x_i and λ).
- The *effective degrees of freedom* are given by

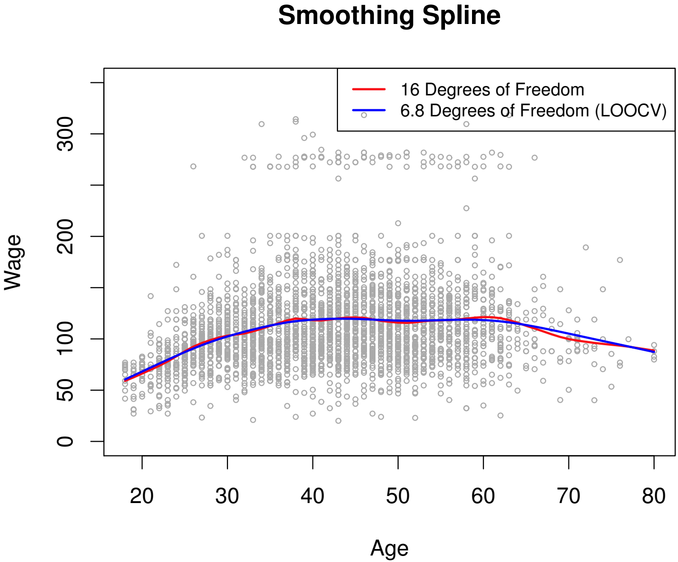
$$df_\lambda = \sum_{i=1}^{n} \{S_\lambda\}_{ii}.$$

Smoothing splines – choosing λ

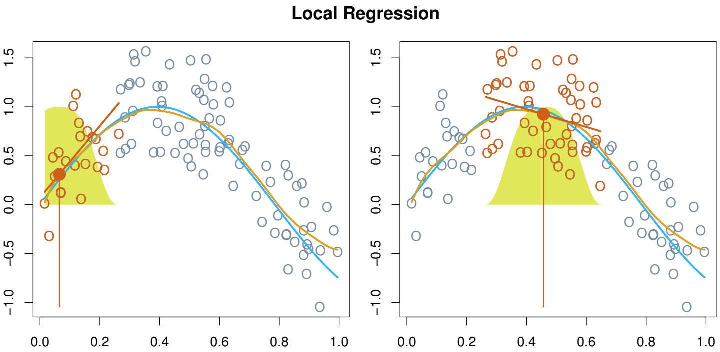
- We can specify *df* rather than λ.
  In R, e.g. *smooth.spline(age, wage, df = 10)*.
- The leave-one-out (LOO) cross-validated error is given by

$$RSS_{CV}(\lambda) = \sum_{i=1}^{n} (y_i - \hat{g}_\lambda^{(-i)}(x_i))^2 = \sum_{i=1}^{n} \left[\frac{y_i - \hat{g}_\lambda(x_i)}{1 - \{S_\lambda\}_{ii}} \right]^2 .$$

  In R, e.g. *smooth.spline(age, wage)*.


Smoothing Spline
16 Degrees of Freedom
6.8 Degrees of Freedom (LOOCV)
Wage
Age
0
50
100
200
300
20
30
40
50
60
70
80

Local regression

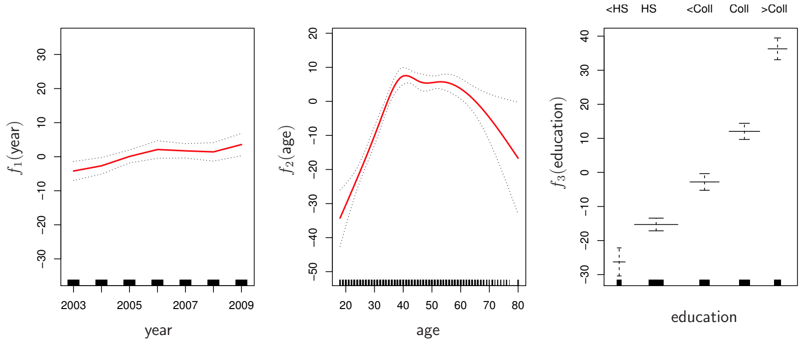


With a sliding weight function, we fit separate linear fits over the range of X by weighted least squares. When the *span* goes down, the flexibility of the model goes up. Use *loess()* function in R.

Generalized additive models

Allows for flexible nonlinearities in several variables, but remains the additive structure of linear models.

$$y_i = \beta_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_p(x_{ip}) + \epsilon_i.$$

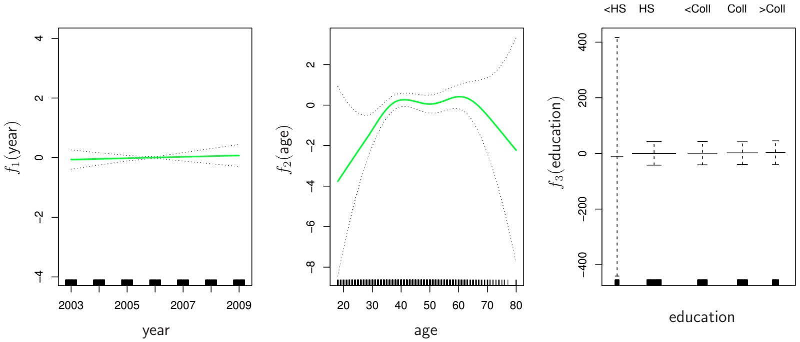


GAM details

- Can fit a GAM simply using e.g. natural splines:
  lm(wage ∼ ns(year,df=5) + ns(age, df=5) + education)
- Coefficients not that interesting, fitted functions are. The previous plot was produced using *plot.gam*.
- Can mix terms – some linear, some nonlinear – and use *anova()* to compute models.
- Can use smoothing splines or local regression as well:
  gam(wage ∼ s(year,df=5) + lo(age, span=5) +education)
- GAMs are additive, although low-order interactions can be included in a natural way using, e.g. bivariate smoothers or interactions of the form *ns(age, df=5):ns(year, df=5)*.

GAMs for classification

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + f_1(X_1) + f_2(X_2) + \cdots + f_p(X_p).$$



gam(I(wage > 250) ~ year + s(age, df = 5) + education, family = binomial)