7.6 Local Regression and Generalized Additive Models

Local Regression

- Local regression is another approach to fitting flexible non-linear functions
- We compute the fit at some target point x_0, using only the nearby training observations.

Algorithm: Local Regression at x_0

1. Gather the k training points where x_i are closest to x_0
2. Assign a weight $K_{i0} = K(x_i, x_0)$ to each point in this neighborhood.
   - the point farthest from x_0 has weight 0
   - the closest point has the highest weight
   - all but the k points in the neighborhood get weight 0
3. Fit a weighted least squares regression by minimizing

$$\sum_{i=1}^{n} K_{i0}(y_i - \beta_0 - \beta_1 x_i)^2$$

4. The fitted value at x_0 is given by $\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$

Notes

- The weights will differ for each value of x_0, so we need to refit the model for every new point.
- Local regression sometimes referred to as a "memory-based procedure", because we need all the training data each time we wish to compute a prediction.

Need to Choose

- How to define the weighting function K
- What type of regression model to fit in step 3
- The *span*, s, which is the proportion of observations used to compute the local regression ($s = k/n$)
  - The span serves as a sort of tuning parameter and controls the wiggliness of the fit
  - Can set s using cross-validation or specify directly

Local Regression

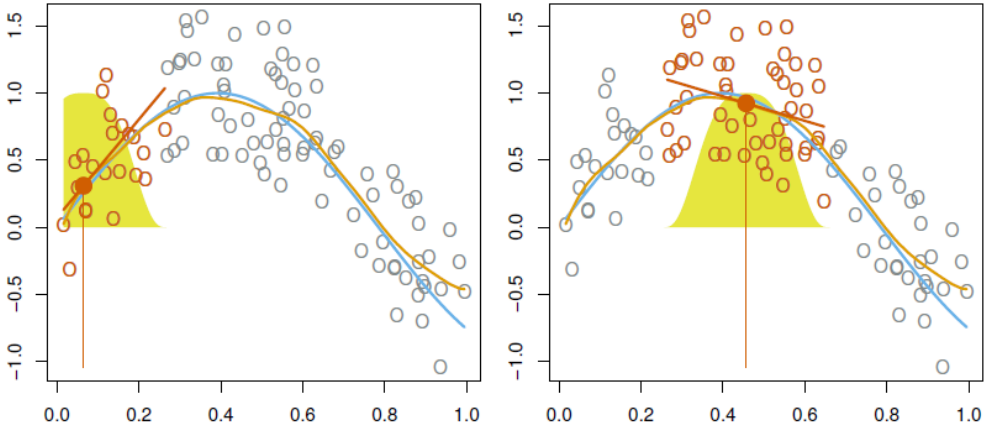


Figure Details

- Local regression illustrated on simulated data.
- The blue curve represents $f(x)$ from which the data were generated
- The light orange curve corresponds to the local regression estimate $\hat{f}(x)$
- The orange points are local to the target point x0, represented by the orange vertical line
- The yellow bell-shape indicates weights assigned to each point, decreasing to zero with distance from the target point
- The fit $\hat{f}(x_0)$ at x_0 is obtained by fitting a weighted linear regression (orange line segment), and using the fitted value at x_0 (orange solid dot) as the estimate.

Local Regression

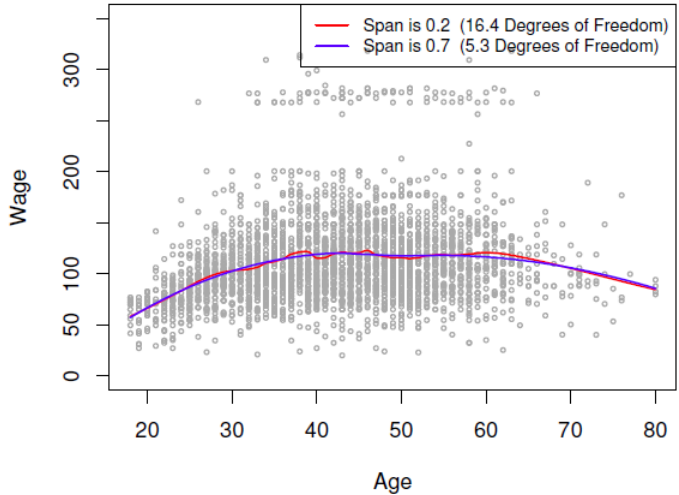


FIGURE 7.10. *Local linear fits to the* `Wage` *data. The span specifies the fraction of the data used to compute the fit at each target point.*

Generalizations

- May fit models that are global in some variables, but local in another (such as time)
  - Called *varying coefficient models*
- Can specify 2-dimensional neighborhoods and fit bivariate models.
- Can theoretically specify p-dimensional neighborhoods, but these tend to perform poorly if $p > 3$ or 4

GAMs

- Allow for flexible nonlinearities in several variables, but retains additive structure of linear models.
- Because we have already thought about regression sing the matrix formulation, this is a trivial extension of the previous sections.

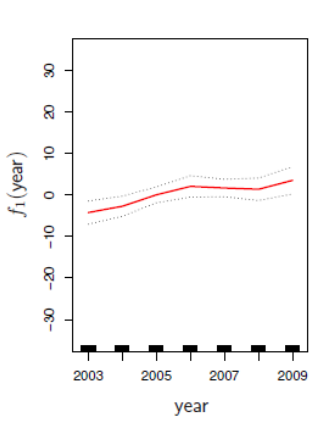
$$y_i = \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) + \epsilon_i$$

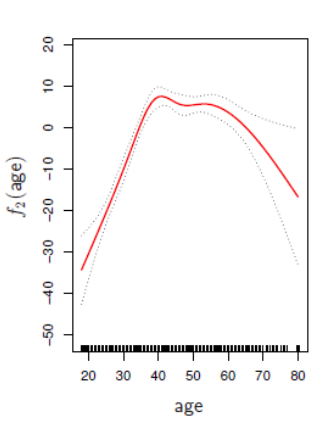
GAMs in R

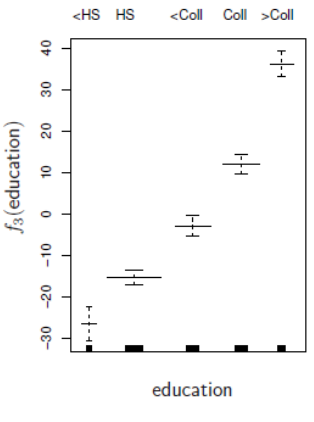
- Can fit GAMs using, for example, natural splines

```
lm(wage ~ ns(year, df=5) + ns(age, df=5) + education)
```

- Coefficients generally less interesting than fitted functions/predictions (using `plot.gam`)
- Can also use polynomial terms, smoothing splines, local regression, etc.
- Can also include low-dimensional interactions.
- Extension to logistic regression, etc., exactly the same as before.


$f_1(\text{year})$
30
20
10
0
−10
−20
−30
2003
2005
2007
2009
year


$f_2(\text{age})$
20
10
0
−10
−20
−30
−40
−50
20
30
40
50
60
70
80
age


$f_3(\text{education})$
<HS
HS
<Coll
Coll
>Coll
40
30
20
10
0
−10
−20
−30
education

GAMs: Pros and Cons

Pros:

- Allow for non-linear transformations on each input variable
- Non-linear fits may be more accurate
- Additive model meas we can examine the individual effect of each X_i on the response
- The smoothness of a function f_i can be summarized via degrees of freedom

Con:

- Additive restriction may cause missed interactions (but we can add them!)