Swiss Institute of Bioinformatics

Moving Beyond Linearity

*Rachel Marcone (Jeitziner) and **Mauro Delorenzi***

Slides credit also to Linda Dib, Frédéric Schütz, Isabelle Dupanloup, …

Moving Beyond Linearity

- Linear models are relatively simple to describe, and have advantages over other approaches in terms of interpretation and inference
- However, the linearity assumption is often an approximation
- Extending linear models to describe the relationship between a response Y and a single predictor X in a flexible way
  - Polynomial regression
  - Step functions
  - Splines
  - Local regression

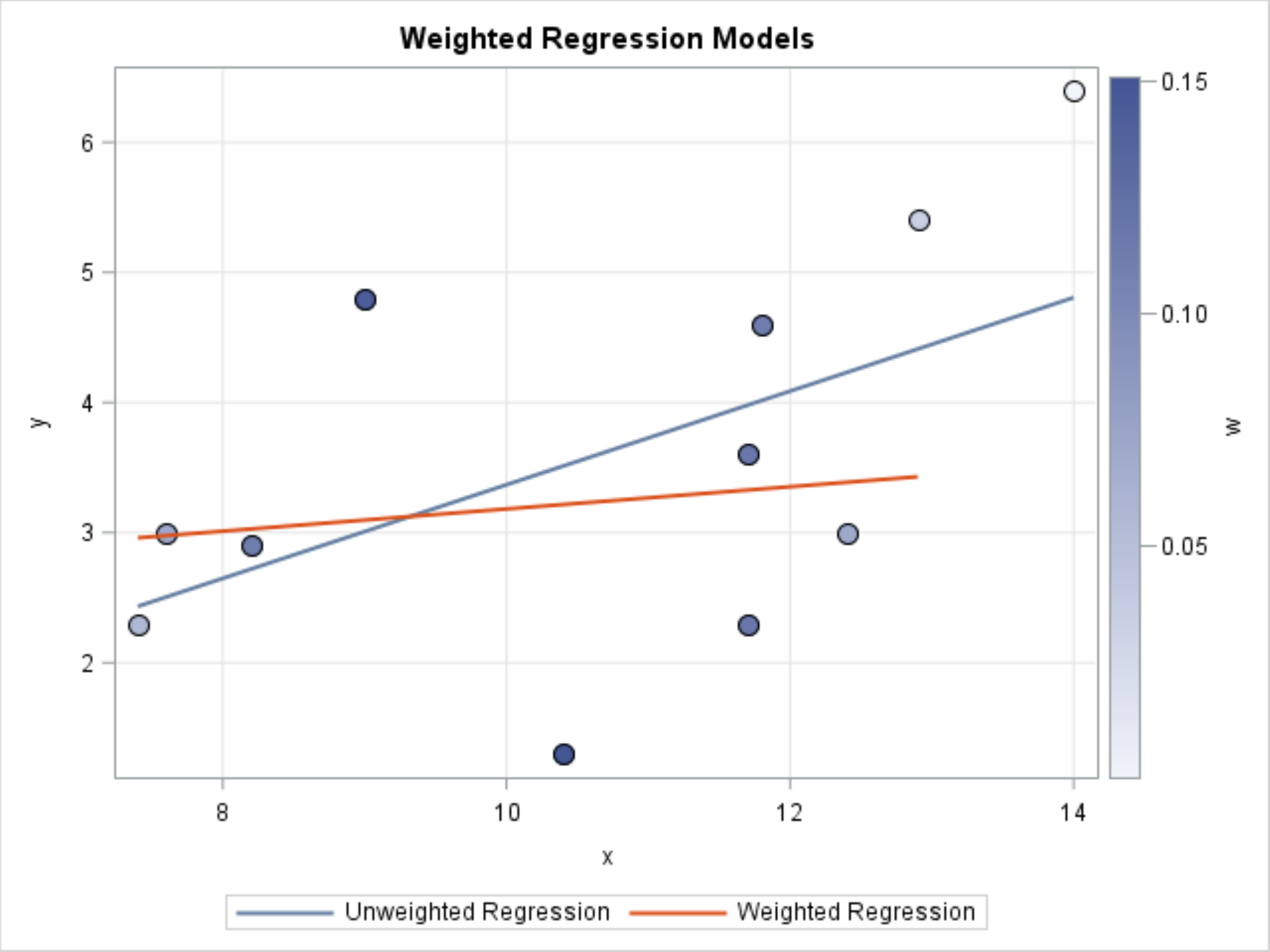
Local Regression

- Most common method: **LOESS** (**locally estimated scatterplot smoothing**)

- LOESS: non-parametric method for fitting a smooth curve between two variables

- Main steps in the loess algorithm:
  1. Choose a smoothing parameter s ∈ (0,1] that represents the proportion of observations to use for local regression
  2. Find the k nearest neighbors to x_i
  3. Assign weights to the nearest neighbors
  4. Perform local weighted (polynomial (often quadratic)) regression

Local Regression

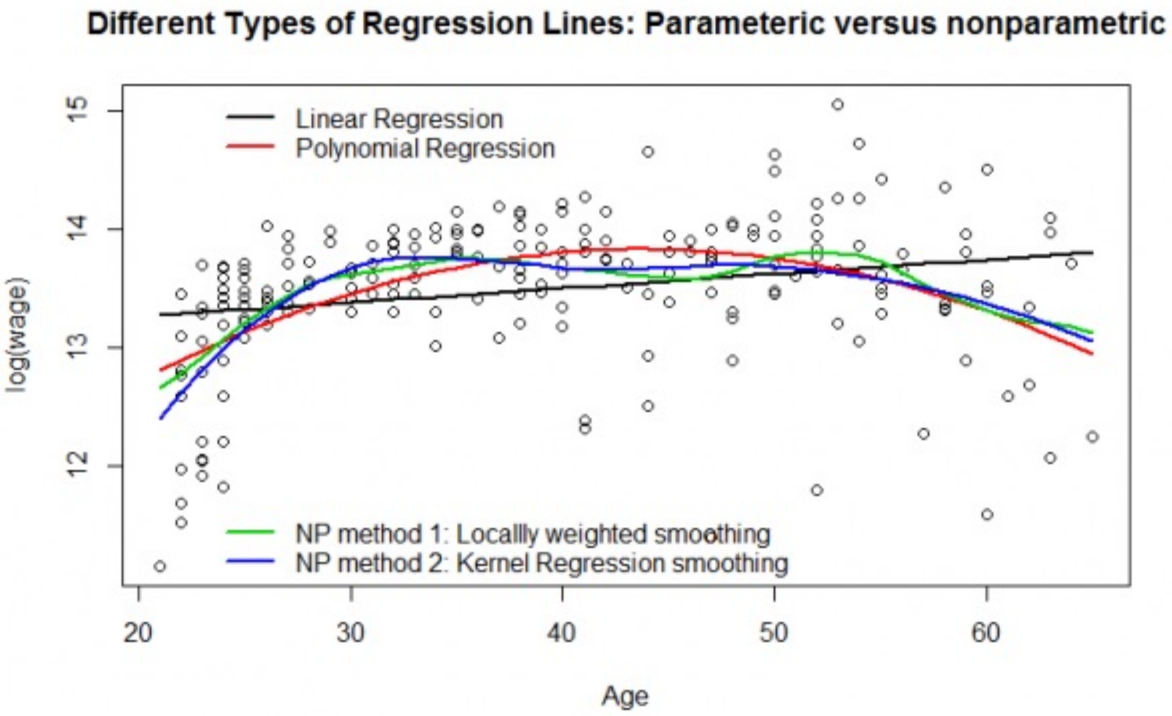


Local Regression

- Advantages
  - The process of fitting a model to the data do not begin with the specification of a function
  - Very flexible

- Disadvantages
  - Requires fairly large, densely sampled data sets in order to produce good models
  - Does not produce a regression function that is easily represented by a mathematical formula
  - Computationally intensive

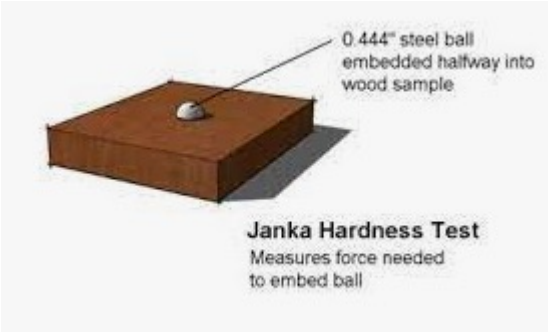
Parametric vs non parametric

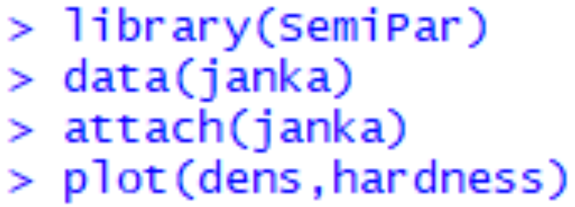


Polynomial Regression

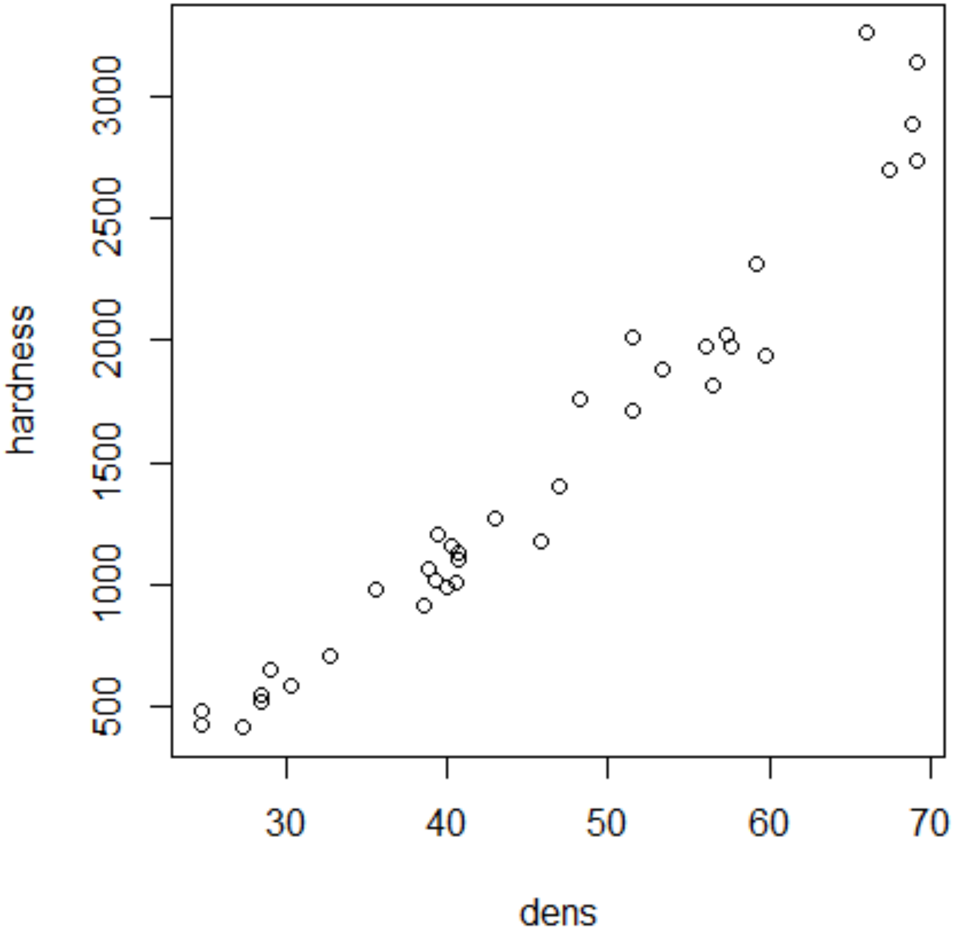
- The standard way to extend linear models by adding extra predictors, obtained by raising the original predictor to a power
- For example
  - A quadratic regression uses two variables: X and X^2
  - A cubic regression uses three variables: X, X^2, and X^3
- Generally speaking, unusual to use powers beyond 3 or 4
  - for large powers, the polynomial can become overly flexible and take on some strange shapes, especially near the boundaries of X

Janka dataset

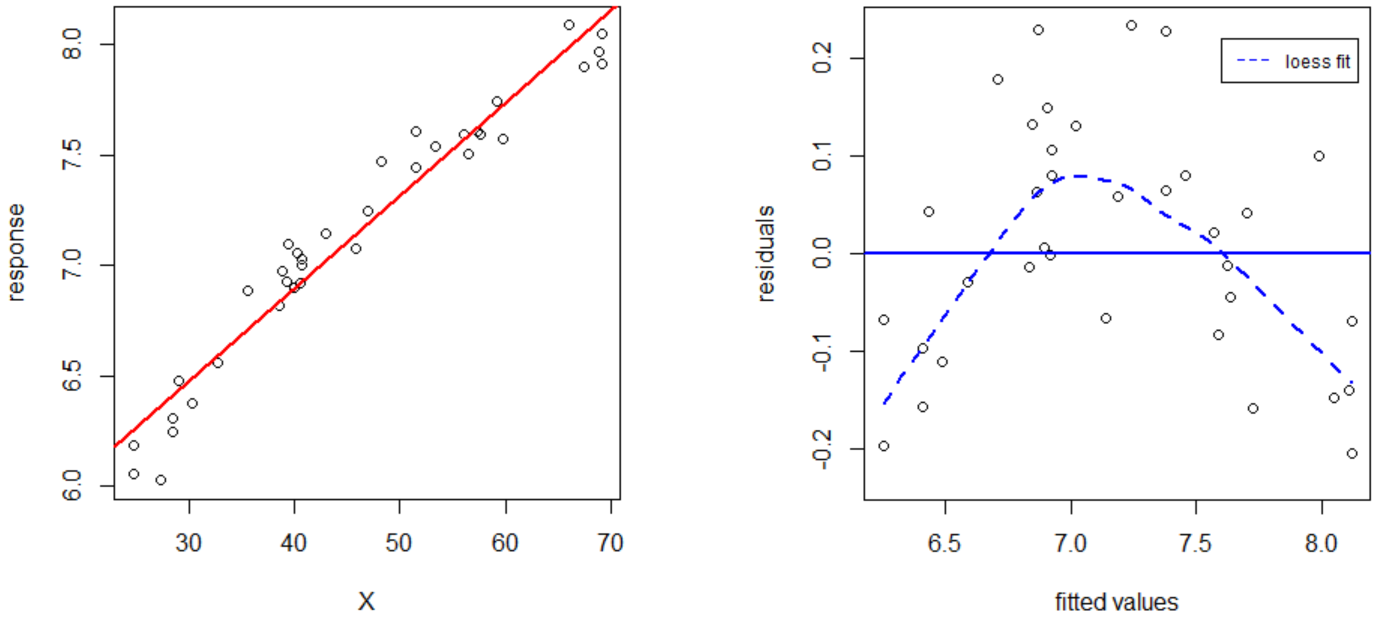




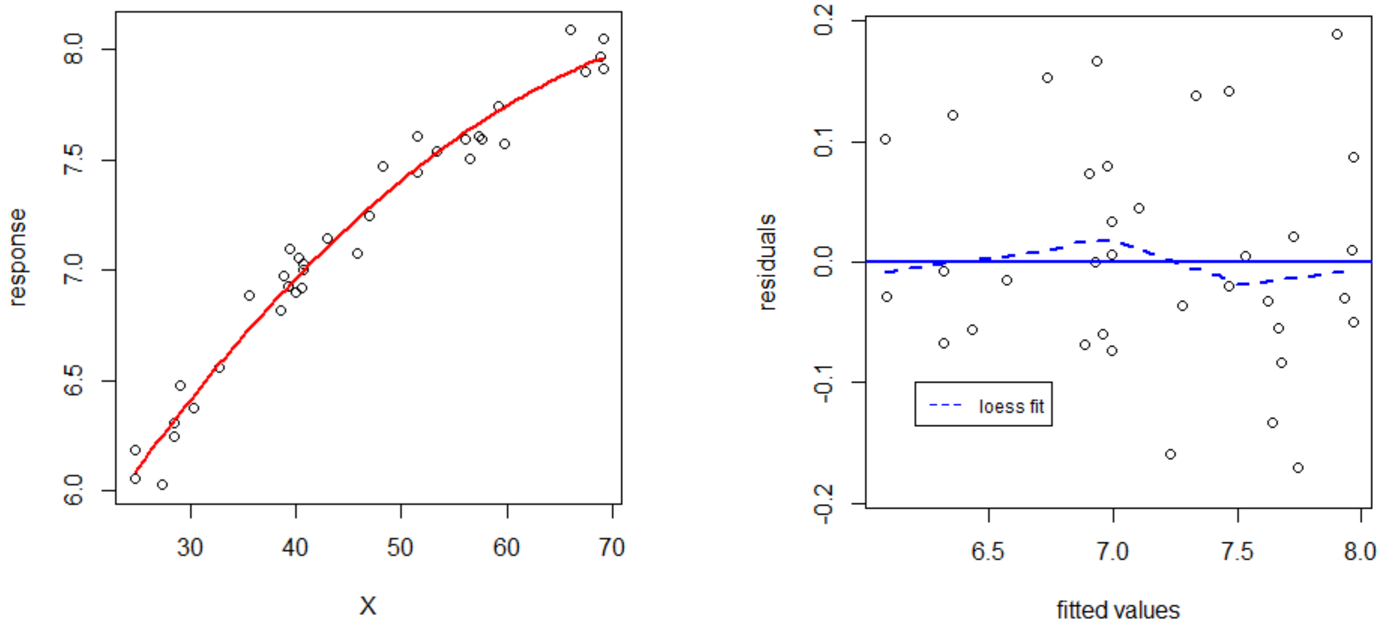
```
> library(SemiPar)
> data(janka)
> attach(janka)
> plot(dens,hardness)
```

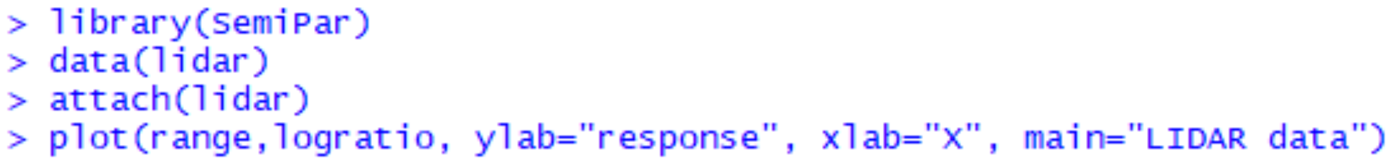


Linear



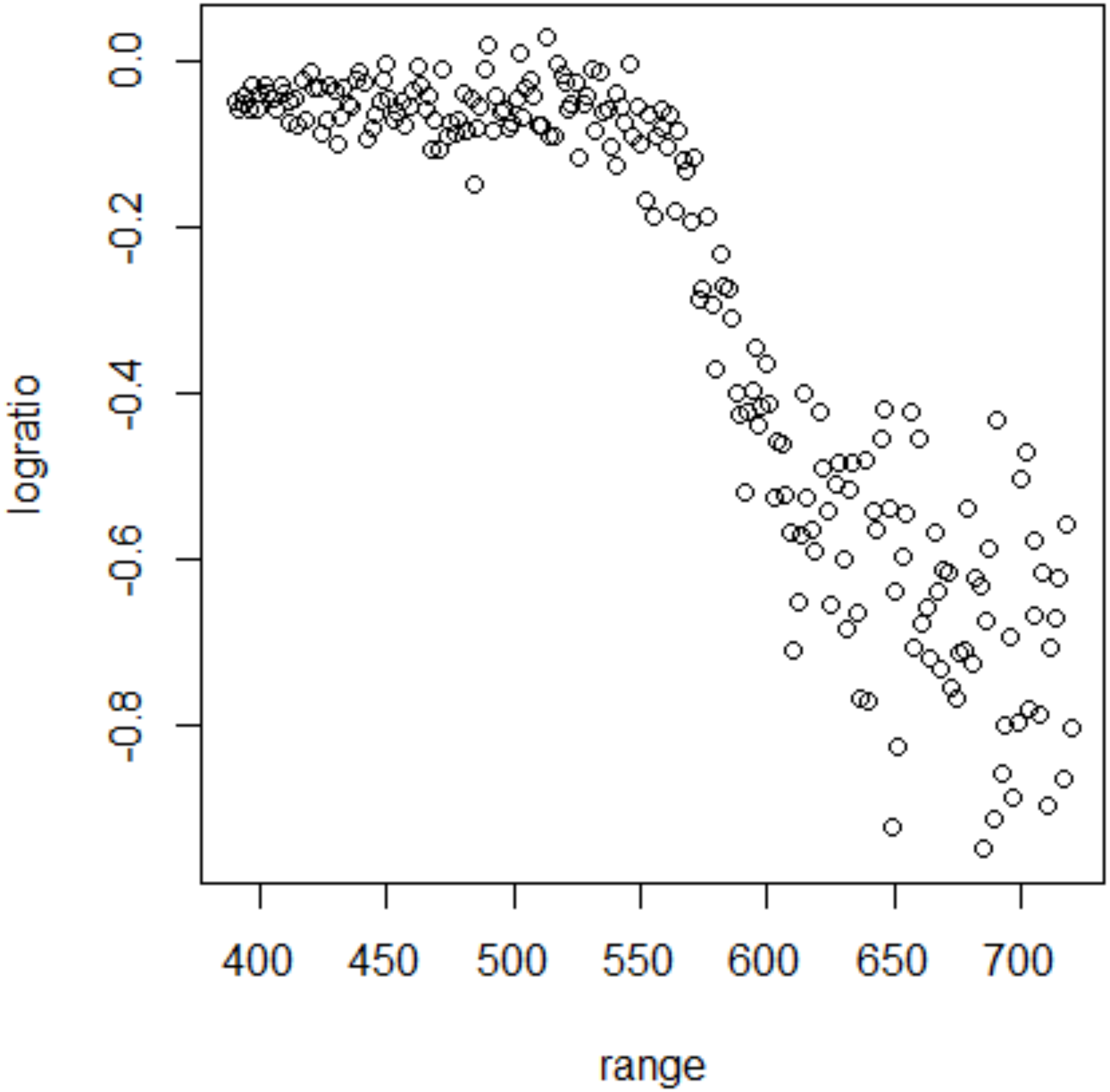
Quadratic



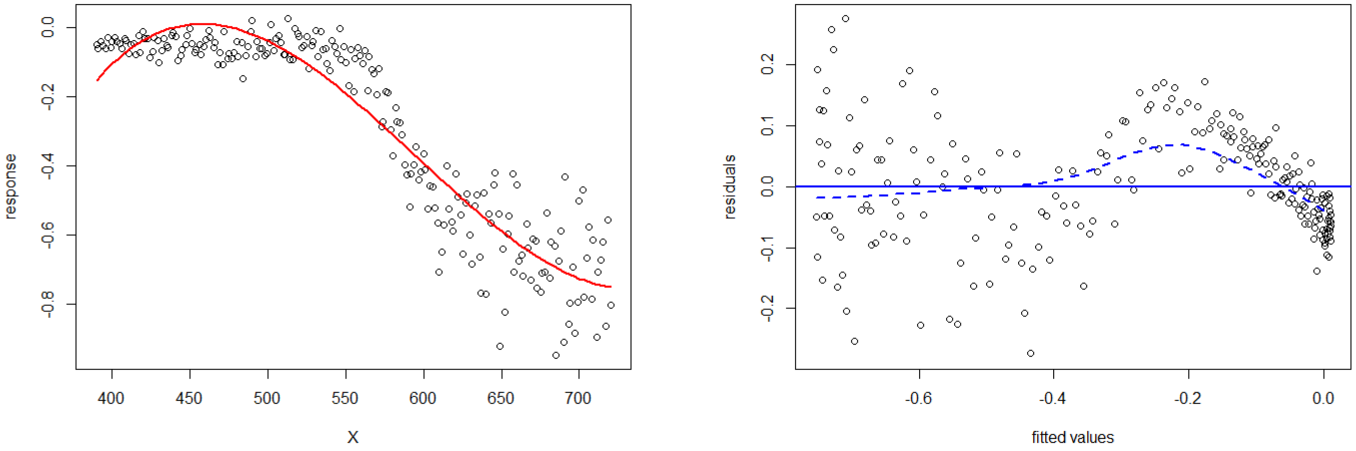


```
> library(SemiPar)
> data(lidar)
> attach(lidar)
> plot(range,logratio, ylab="response", xlab="X", main="LIDAR data")
```

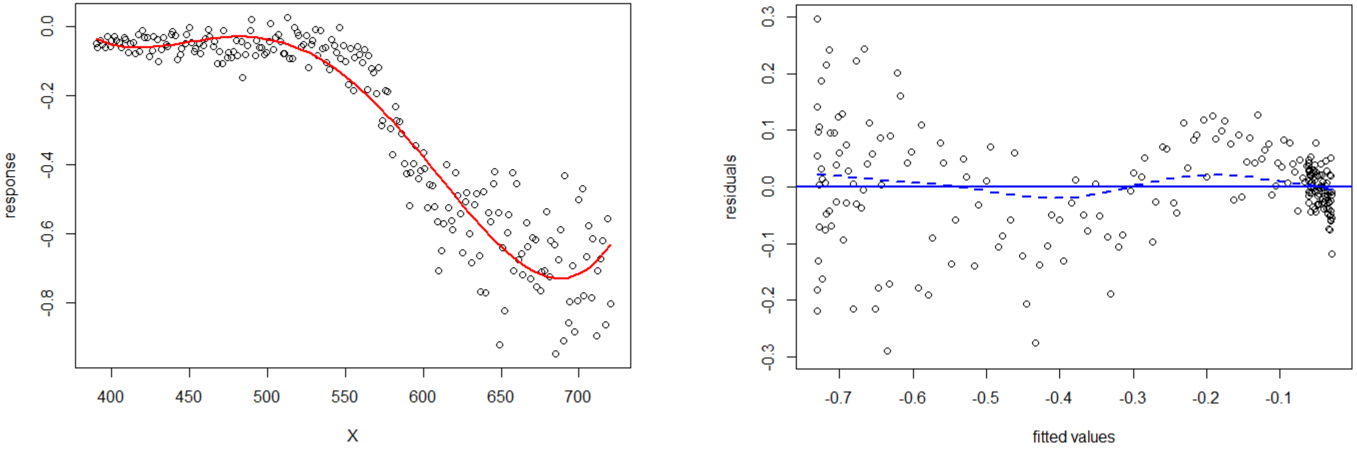
Lidar dataset



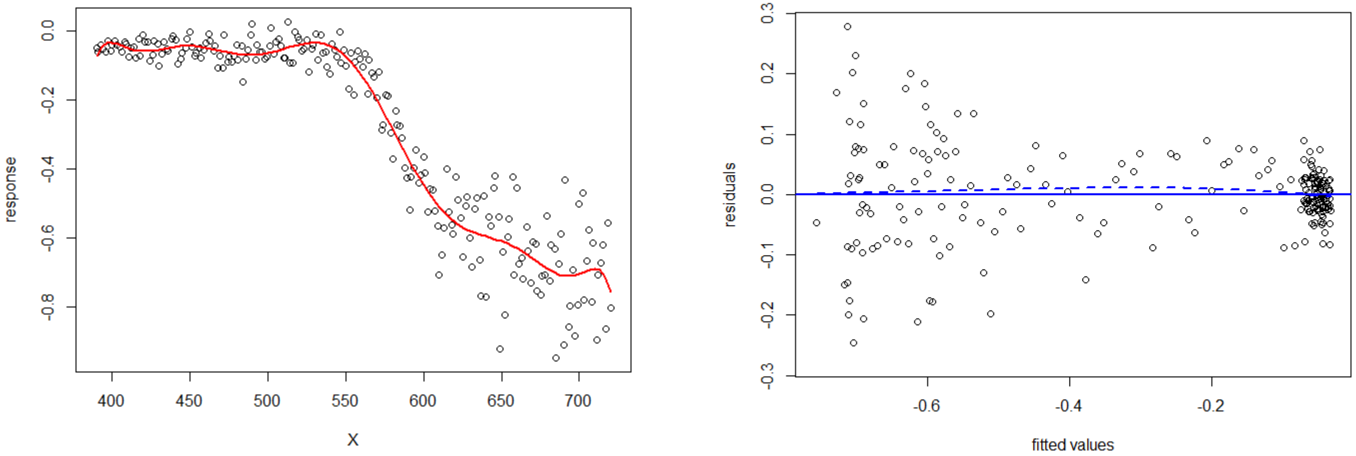
Polynomial degree 3



Polynomial degree 4



Polynomial degree 10



Pointwise Confidence Intervals

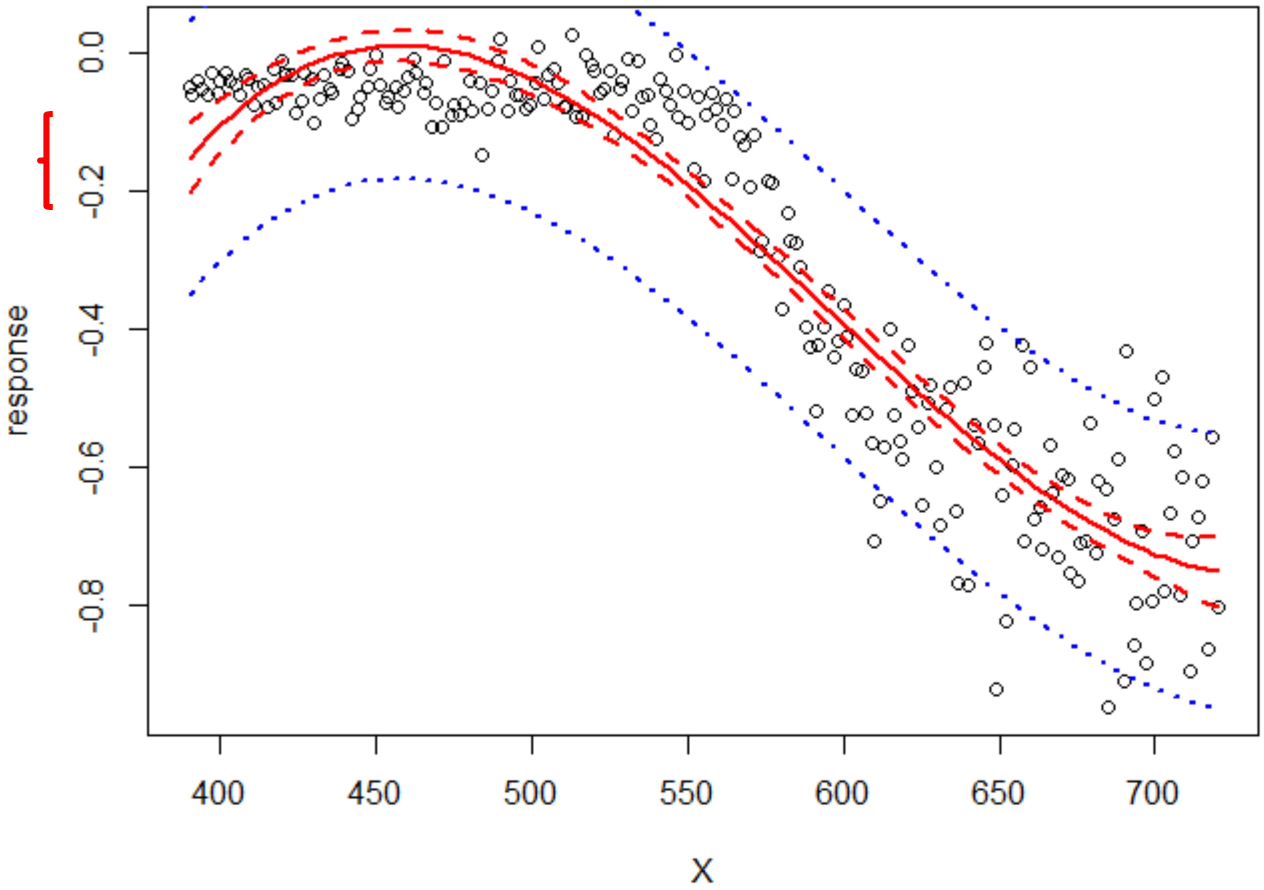
$$\hat{f}(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0 + \hat{\beta}_2 x_0^2 + \hat{\beta}_3 x_0^3 + \hat{\beta}_4 x_0^4$$

- What is the variance of the fit?
- Least squares returns variance estimates for each of the fitted coefficients, as well as the covariances between pairs of coefficient estimates. We can use these to compute the estimated variance of fitted values

- The estimated **pointwise** standard error is the square-root of the variance
- This computation is repeated at each reference point x_0, and we plot the fitted curve, as well as twice the standard error on either side of the fitted curve

We plot twice the standard error because, for normally distributed error terms, this quantity corresponds to an approximate 95 % confidence interval

Pointwise Confidence Intervals



Confidence interval

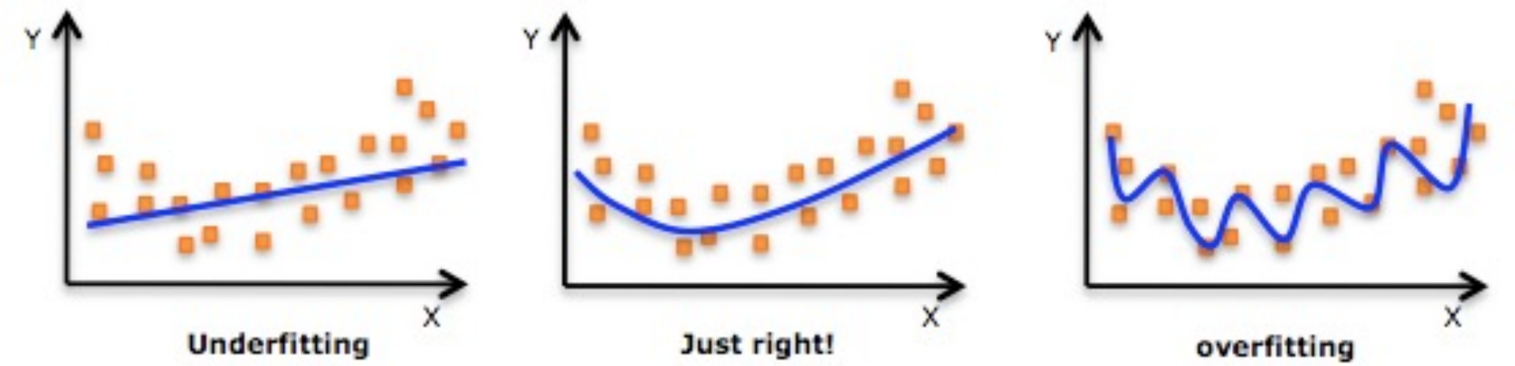
Prediction interval

Narrow bands: describe the uncertainty about the regression line

Wide bands: describe where most (95% by default) predictions would fall, assuming normality and constant variance.

Issues with Polynomial Regression

- Overfitting



- Inherently non-local

Step Functions

- Using polynomials of the predictor imposes a global structure on the non-linear function of X
- We can instead use step functions in order to avoid imposing such a global structure
  - We create cut-points (aka *knots*) $c_1, c_2, \ldots, c_K$ in the range of X, and then construct K+1 new variables

$$\begin{aligned}
C_0(X) &= I(X < c_1), \\
C_1(X) &= I(c_1 \leq X < c_2), \\
C_2(X) &= I(c_2 \leq X < c_3), \\
&\vdots \\
C_{K-1}(X) &= I(c_{K-1} \leq X < c_K), \\
C_K(X) &= I(c_K \leq X),
\end{aligned}$$

Where *I* is an indicator function that returns a 1 if the condition is true, and returns a 0 otherwise

  - Notice that for any value of X, $C_0(X)+C_1(X)+\ldots+C_K(X) = 1$, since X must be in exactly one of the K + 1 intervals

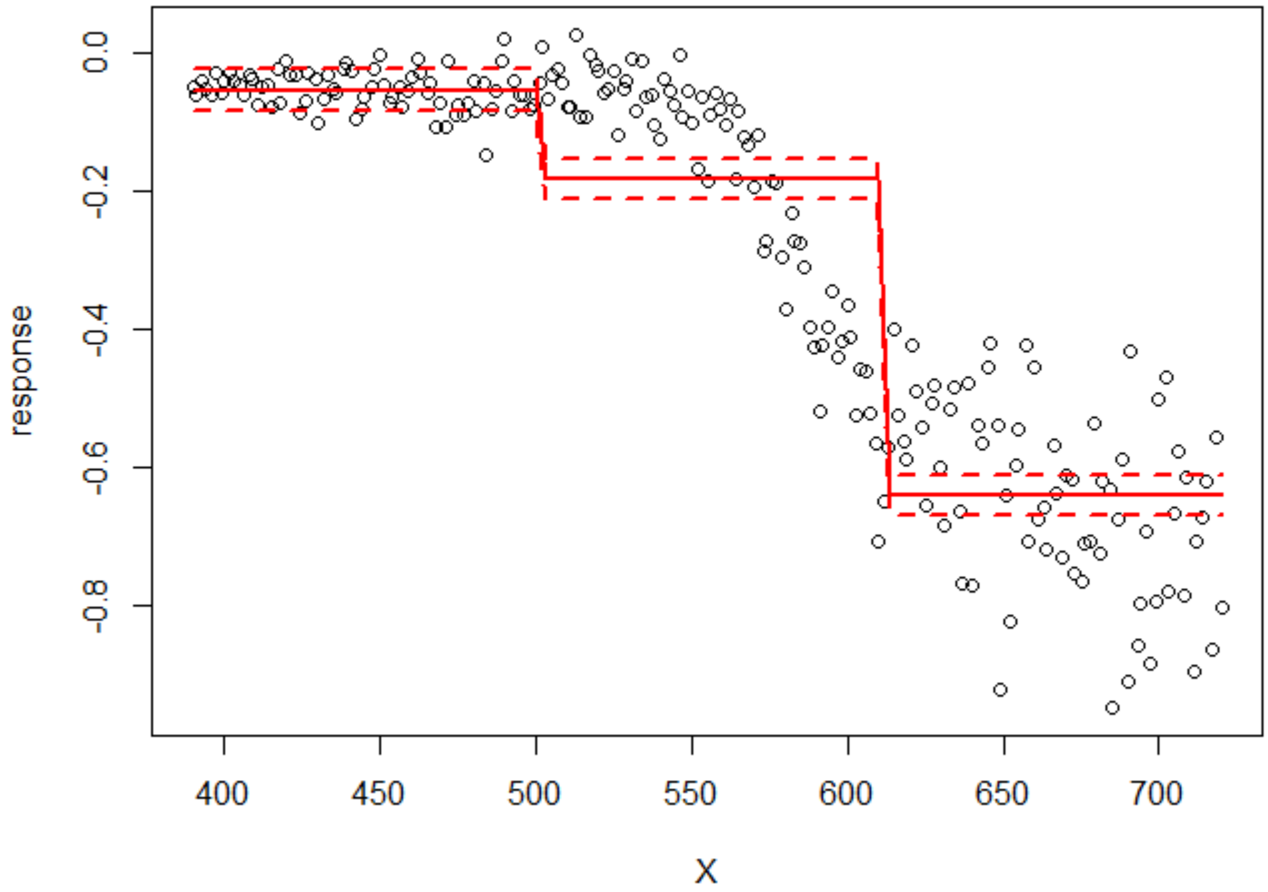
Step Functions

- We then use least squares to fit a linear model using $C_1(X)$, $C_2(X), \ldots, C_K(X)$ as predictors

$$y_i = \beta_0 + \beta_1 C_1(x_i) + \beta_2 C_2(x_i) + \ldots + \beta_K C_K(x_i) + \varepsilon_i$$

- For a given value of X, at most one of $C_1,\ldots,C_K$ can be non-zero
- Note that when $X < c_1$, all of the predictors are zero, so β_0 can be interpreted as the mean value of Y for $X < c_1$
- By comparison, the above regression model predicts a response of $\beta_0+\beta_j$ for $c_j \leq X < c_{j+1}$, so β_j represents the average increase in the response for X in $c_j \leq X < c_{j+1}$ relative to $X < c_1$

Step Functions



Clearly, unless there are natural breakpoints in the predictors, piecewise-constant functions can miss the trends

Basis Functions

- Polynomial and piecewise-constant regression models are in fact special cases of a basis function approach

- The idea is to have at hand a family of functions or transformations that can be applied to a variable X, e.g.

$$b_1(X),\ b_2(X),\ \ldots,\ b_K(X)$$

- Instead of fitting a linear model in X, we fit a linear model with $b_1(X), b_2(X), \ldots, b_K(X)$ as predictors, i.e.

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \beta_3 b_3(x_i) + \ldots + \beta_K b_K(x_i) + \varepsilon_i$$

Basis Functions

$$b_1(X), b_2(X), \ldots, b_K(X)$$

$$y_i = \beta_0 + \beta_1 b_1(x_i) + \beta_2 b_2(x_i) + \beta_3 b_3(x_i) + \ldots + \beta_K b_K(x_i) + \varepsilon_i$$

- For polynomial regression, the basis functions are

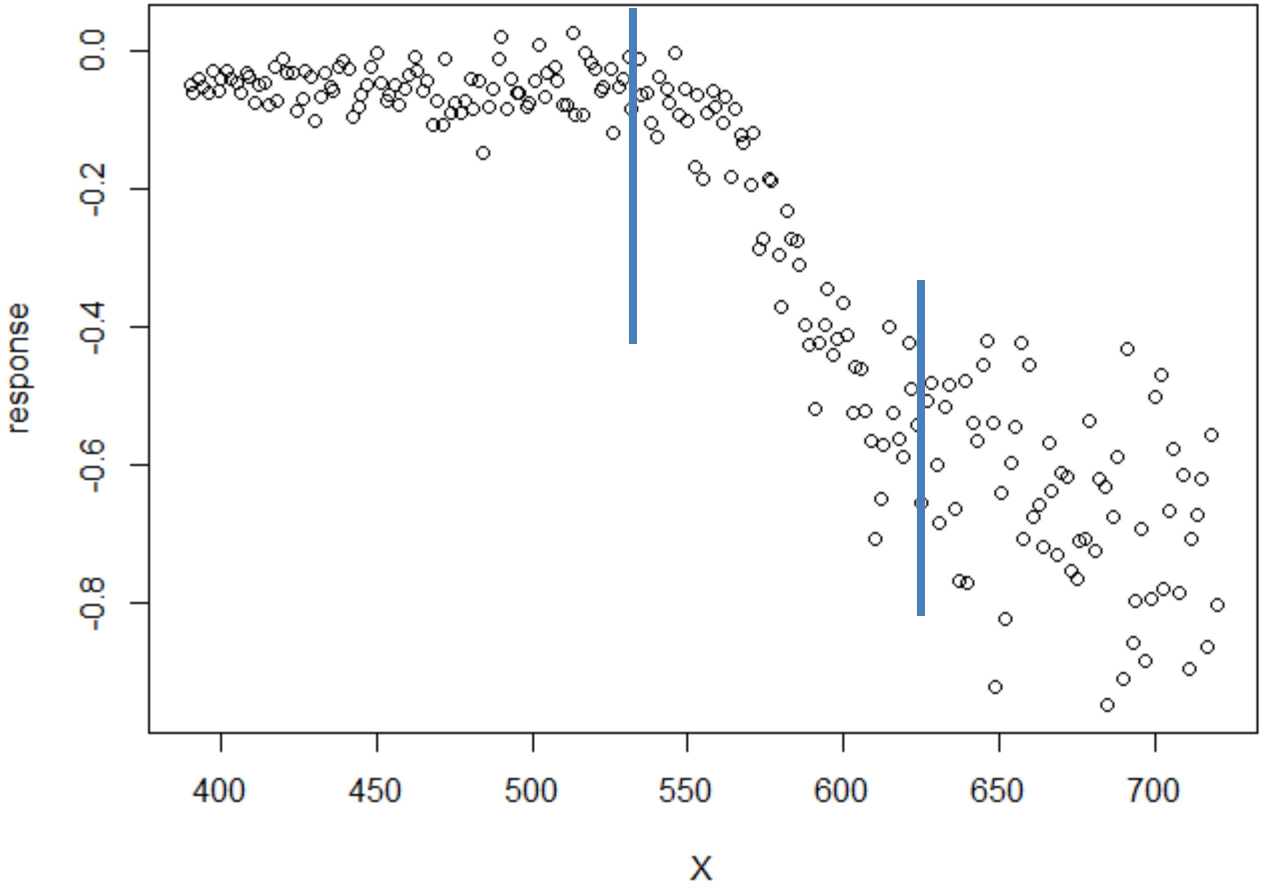
$$b_j(x_i) = (x_i)^j$$

- For piecewise constant functions they are

$$b_j(x_i) = I(c_j \leq x_i < c_{j+1})$$

- Basis functions come in various forms and shapes …

Regression Splines



Regression Splines

- More flexible than polynomial and step functions, in fact an extension of the two
- Instead of fitting a high-degree polynomial over the entire range of X, piecewise polynomial regression involves fitting **separate low-degree polynomials** over different regions of X
    - Dividing the range of X into K distinct regions
    - Within each region, a polynomial function is fit to the data
    - The polynomials are typically constraint to join **smoothly** at the cut-points or behave linearly beyond boundaries
- Provided that the interval is divided into enough regions, this can produce an extremely flexible fit
- “Splines”: series of local polynomial functions

Regression Splines

- For example, a piecewise cubic polynomial works by fitting a cubic regression model of the form

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \varepsilon_i$$

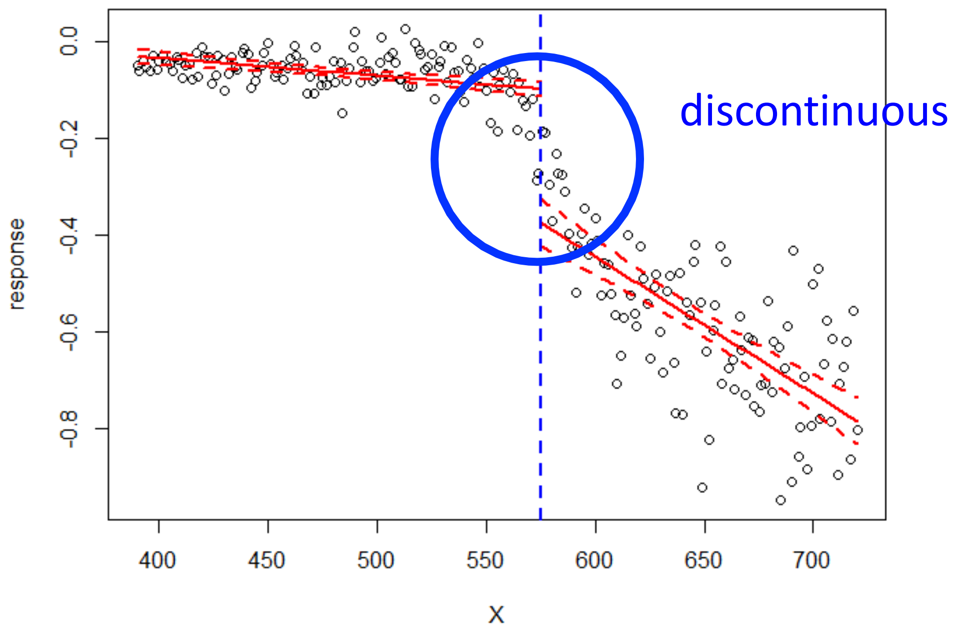
- Where the coefficients β_0, β_1, β_2, and β_3 differ in different parts of the range of X
- The points at which the coefficients change are called **knots**
- For example, a piecewise cubic polynomial with a single knot at a point c takes the form

$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \epsilon_i & \text{if } x_i < c \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \epsilon_i & \text{if } x_i \geq c \end{cases}$$

Regression Splines

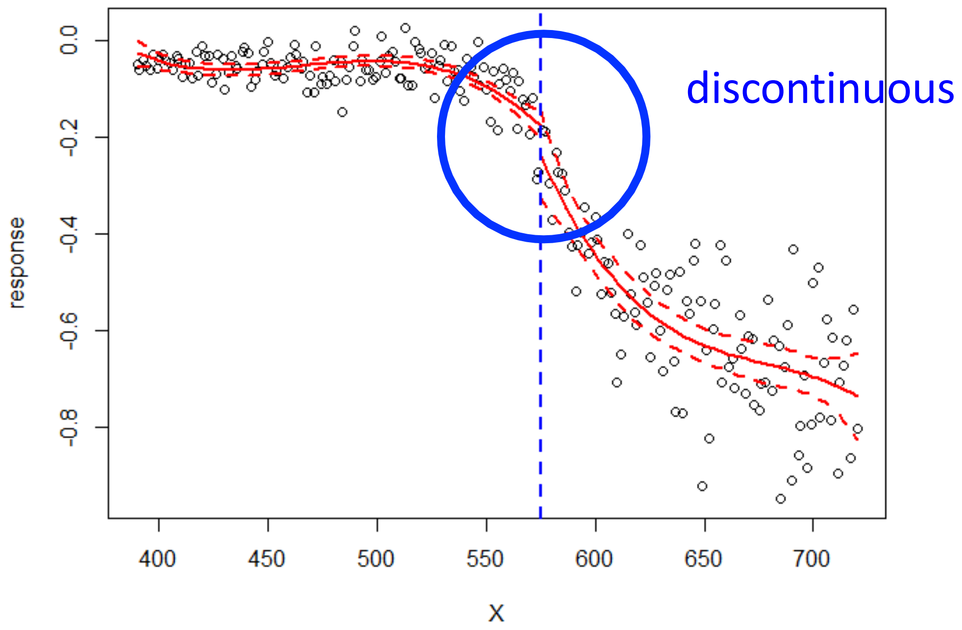
- Using more knots leads to a more flexible piecewise polynomial
- In general, if we place K different knots throughout the range of X, then we will end up fitting K + 1 different polynomials
  - Since each cubic polynomial has four parameters, for a piecewise cubic polynomial with a single knot we are using a total of eight degrees of freedom
  - Constraints reduce the degrees of freedom, for example down to only one additional DF for each added knot
- Note that we do not need to use a cubic polynomial
  - For example, we can instead fit piecewise linear functions

Piecewise linear fit
with 1 knot



discontinuous

Piecewise cubic fit
with 1 knot

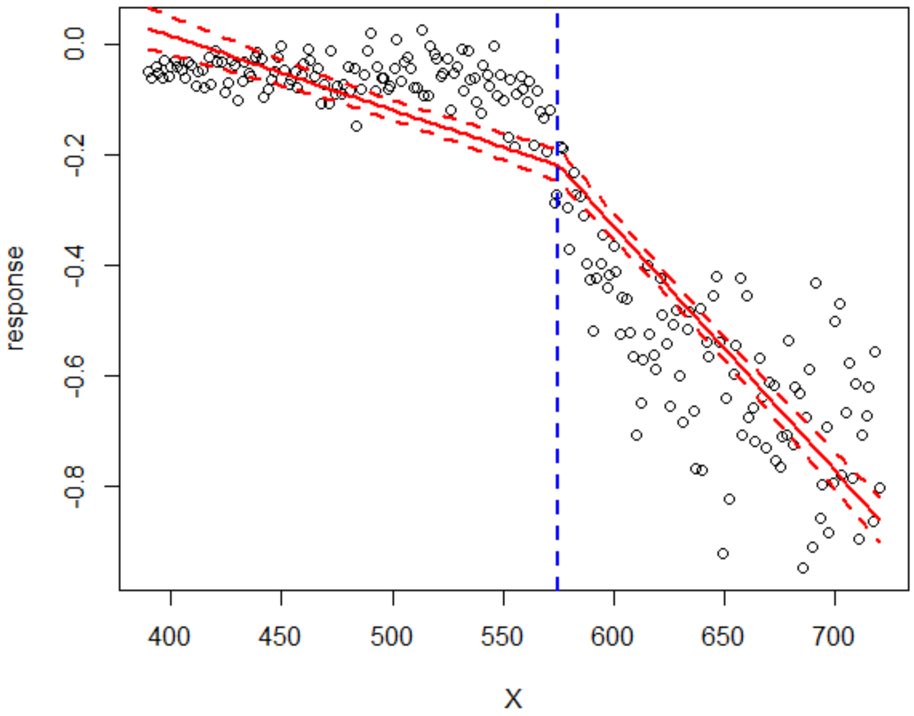


discontinuous

Constraints & Splines

- How to address discontinuity?
  - By adding additional constraint on the model that the fitted curve must be continuous at the knots

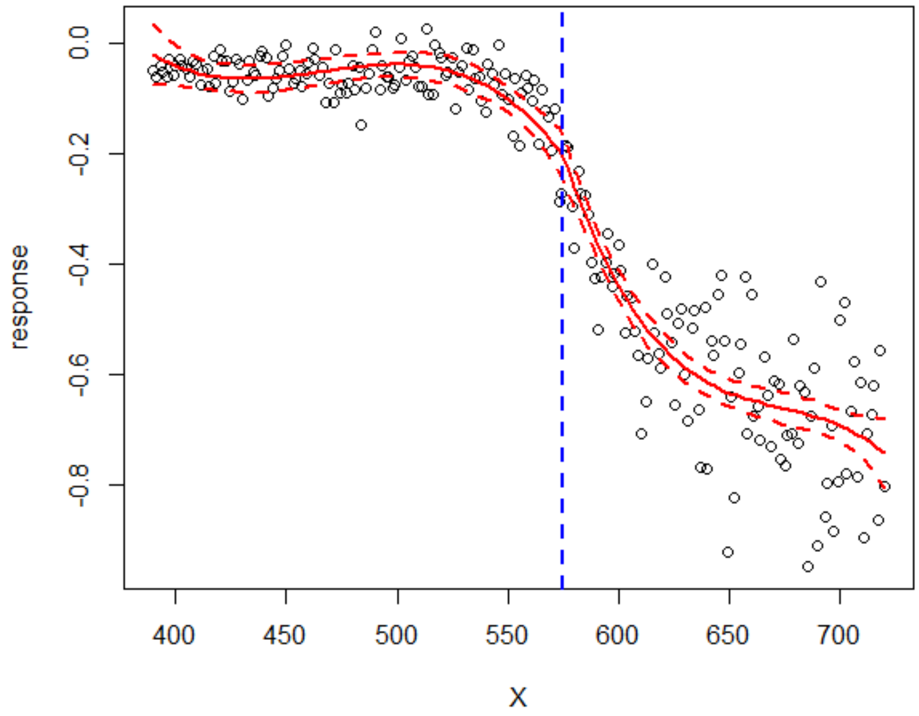
Piecewise **<u>linear</u>** fit with 1 knot & continuity constraint



Constraints & Splines

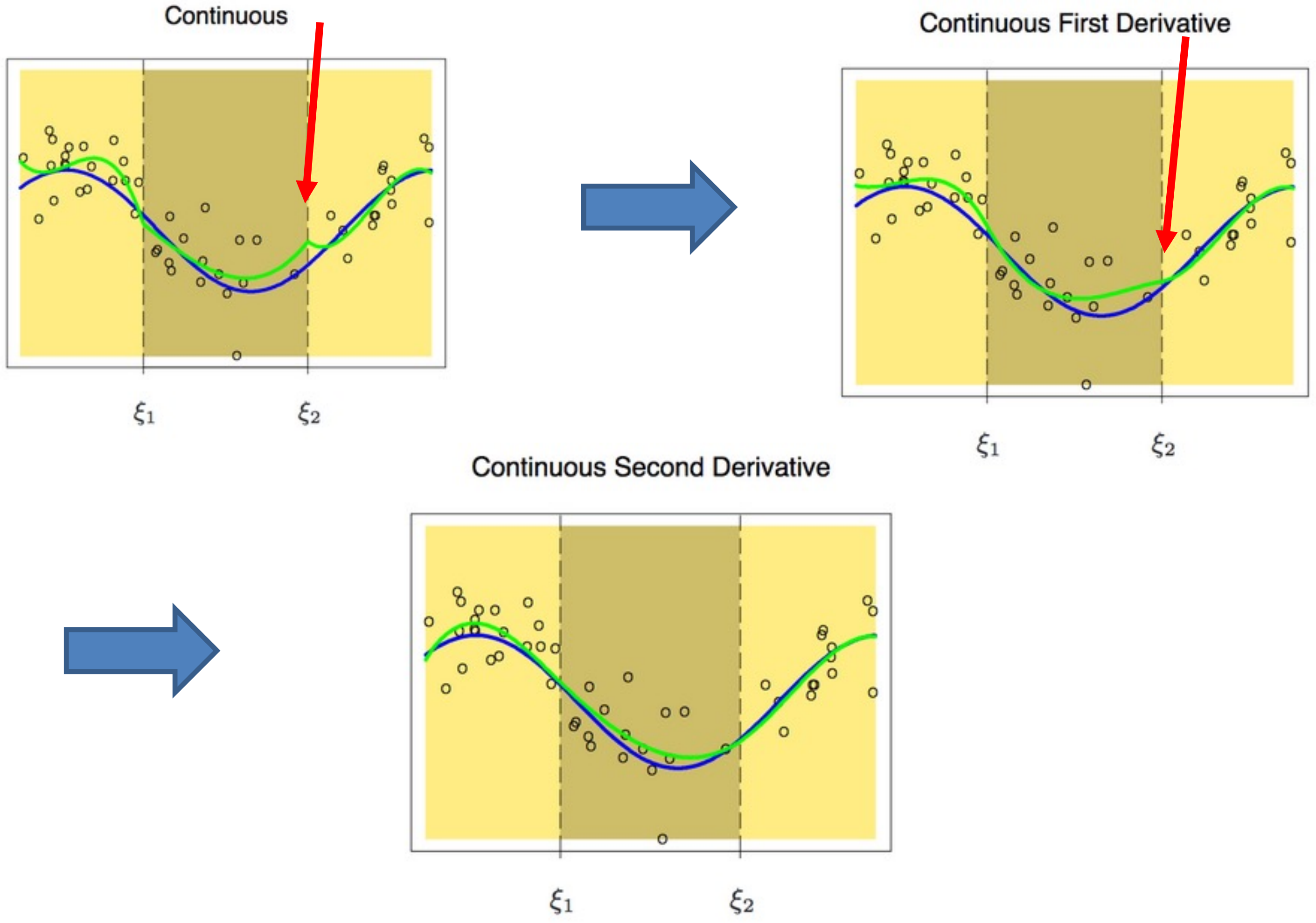
- How to address discontinuity?
  - By adding additional constraint on the model that the fitted curve must be continuous at the knots

Piecewise **cubic** fit with 1 knot & continuity constraint



Constraints & Splines

- To smoothen the polynomials at the knots, we **add extra constraints: the first and second derivatives of both the polynomials must be same.**


Continuous

Continuous First Derivative

Continuous Second Derivative

Regression Splines

- How to choose the number and locations of the knots ?
  - The regression spline is most flexible in regions that contain a lot of knots, because in those regions the polynomial coefficients can change rapidly.
  - Hence, one option is to place more knots in places where we feel the function might vary most rapidly, and to place fewer knots where it seems more stable.
  - While this option can work well, in practice it is common to place knots in a uniform fashion.

Can we avoid having to choose the knots ??

Smoothing Splines

- Mathematically more challenging than regression splines but they are more smoother and flexible as well
- Typically uses polynoms of degree 3

- Does not require the selection of the number of Knots, every (unique) value of (x_i) acts like a Knot => they have a very high flexibility (and risk of "overfitting" to the data like passing precisely through each point (resp. mean of y for each unique x) => they can be very **Rough**

- The flexibility is controlled by a **Roughness Penalty** which controls the "roughness" and the (excessive) flexibility the model would otherwise have

Constraints & Splines

- Continuity level 0: the function f has no discontinuity (no “jumps”) …
- Continuity level 1: the derivative function f' has no discontinuity (no “jumps”) …
- Continuity level 2: the second derivative f'' has no discontinuity (no “jumps”) …

… at the knot points.

Polynomials are continuous and all its derivatives are continuous over their domain, the problem is only at the know where we abruptly change from one polynomial function to another one.

f' : slope of f

$f'(x) > 0$ ⇔ graph is rising in the position x

$f'(x) < 0$ ⇔ graph is falling

$f'(x) = 0$ ⇔ graph might have a maximal or minimal point at x

A polynomial of degree 3 cannot have more than 2 such points (f' is quadratic)

f'' : curvature of f :

$f''(x) > 0$ ⇔ graph is locally (around x) convex (“left-turning”)

$f''(x) < 0$ ⇔ graph is locally concave (“right-turning”)

$f'(x) = 0$ ⇔ graph might have an inflection point at x

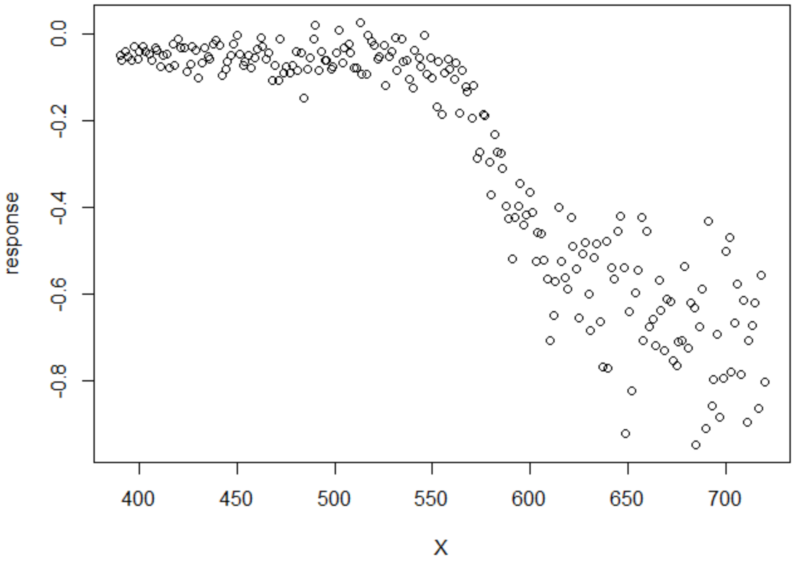
A polynomial of degree 3 cannot have more than 1 such point (f'' has degree 1)

Smoothing Splines

- In the last section we discussed regression splines, where we specified a set of knots, produced a sequence of basis functions, and used least squares to estimate the spline coefficients
- We now introduce a somewhat different approach that also produces a spline
- Similar to regression splines but result from minimizing a residual sum of squares criterion subject to a roughness penalty, which is smaller if the function is smoother

Smoothing Splines

- Aim: minimize the error function which is modified by adding a Roughness Penalty which penalizes it for Roughness (Wiggliness) and high variance.



$$DATA = SIGNAL + NOISE$$

$$y_i = \mu(x_i) + \epsilon_i$$

$$\boxed{\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt}$$

λ: tuning parameter

Constraints & Splines

f'' : curvature of f :

f''(x) > 0 ⇔ graph is locally (around x) concave (“left-turning”)
f''(x) < 0 ⇔ graph is locally convex (“right-turning”)
f'(x) = 0 ⇔ graph might have an inflection point at x

The penalty limits the variation in the curvature

(As f'' is also the slope of f') this means it limits also the change of slope of f, f tends to continue with the same slope that is locally tends to be a straight line)
The constraints imposed at the knots additionally do not allow for discontinuity in the slope or curvature at the knots, where a new polynomial takes over.

Smoothing Splines

$$\boxed{\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt}$$

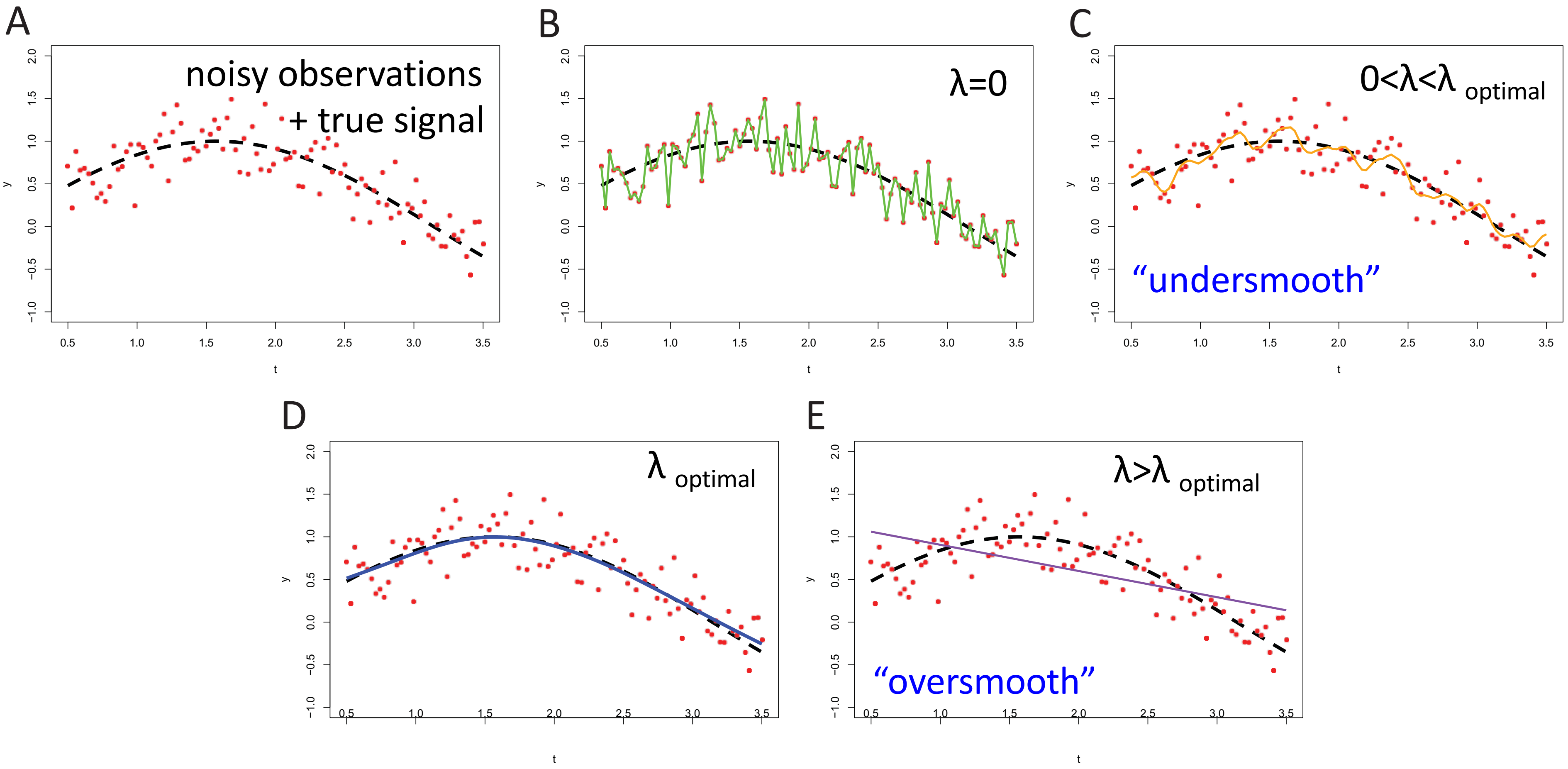
- It can be shown that the solution of the above minimization have some special properties:
  - it is a piecewise cubic polynomial,
  - with knots at unique values of x_1, . . . , x_n,
  - and continuous first and second derivatives at each knot
  - furthermore, it is linear in the regions outside of the extreme knots
  - in other words, **the solution is a cubic spline with knots at x_1,...,x_n**

Smoothing Splines

$$\boxed{\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt}$$

- What value of lambda ?
  - User defined
  - Optimisation criteria, f.ex. cross-validation best fitting

Smoothing Splines



$$\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt$$

Smoothing Splines

- Summary
  - Excellent for fitting in certain situations
  - Test for significance of parameters, best model etc. : possible
  - Use for inference, identification of important explanatory variables: work in progress