STATS 415: Splines

Prof. Liza Levina

Department of Statistics, University of Michigan

Moving Beyond Linearity

- So far we have focused on linear models.
- Linear models have a significant advantage: they are simple. Great for interpretation and inference.
- A significant limitation: not very flexible, which negatively affects prediction.
- Ridge and the Lasso improve on OLS, but they still use a linear model – can only be improved so far.

Non-Linear Methods

- Goal: relax the linearity assumption, but attempt to maintain as much interpretability as possible.
- Polynomial regression: a simple extension of the linear model
- Splines
- Generalized Additive Models (GAM).

Polynomial Regression

- Historically, the standard way to extend linear regression to non-linear data was to replace

$$y = \beta_0 + \beta_1 x + \varepsilon$$

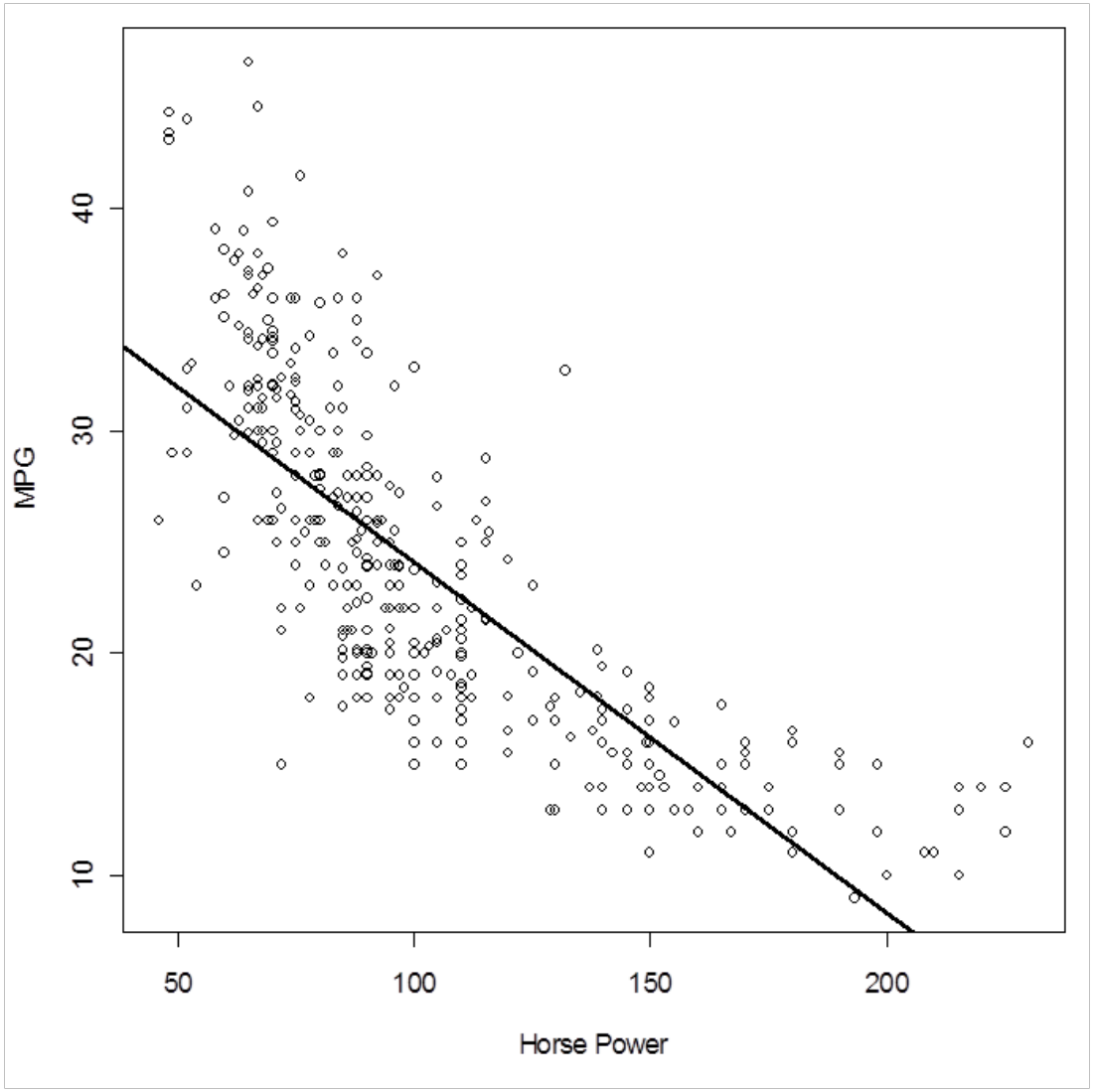
with

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d + \varepsilon$$

- For large enough d this can produce an extremely non-linear curve.
- The parameters (β) can be easily estimated using standard methods, e.g. OLS; we simply add more columns to the design matrix X.

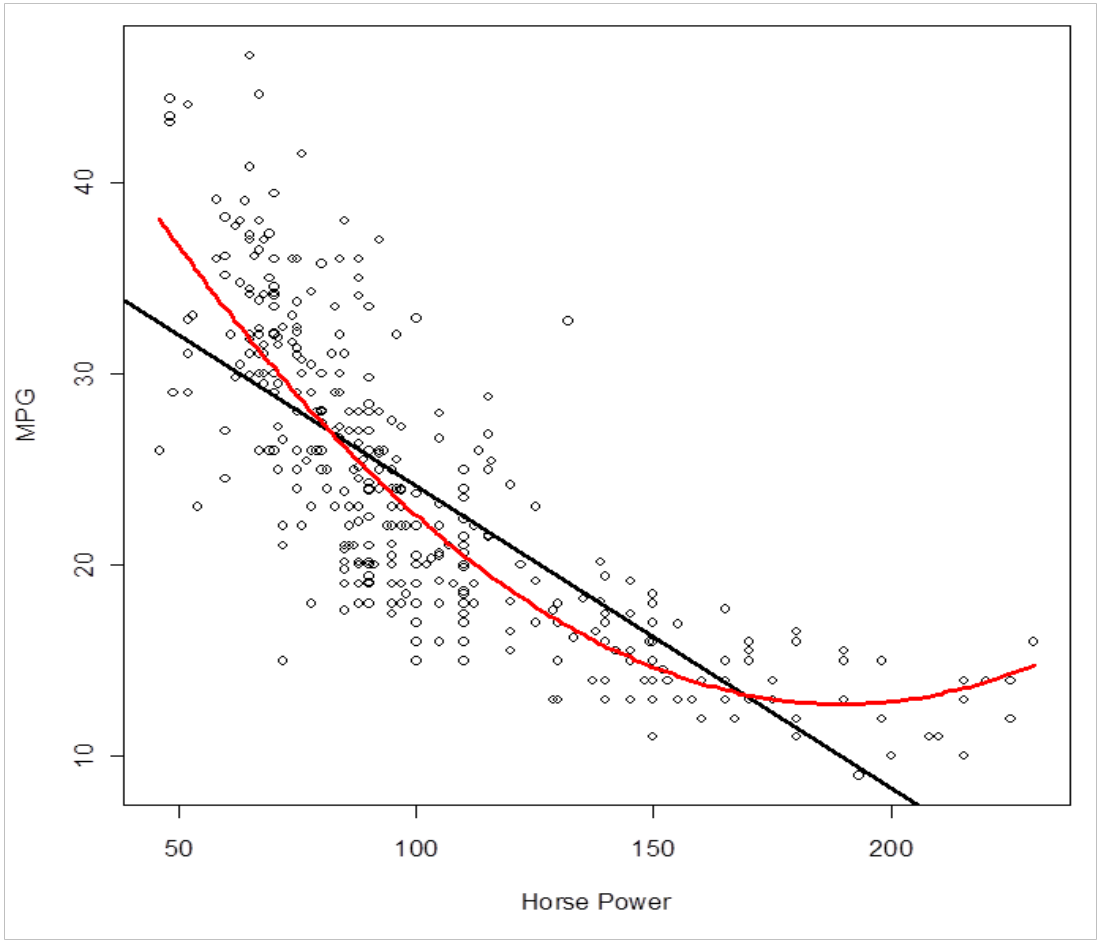
MPG vs Horse Power

- Linear regression: $y = \beta_0 + \beta_1 x + \varepsilon$



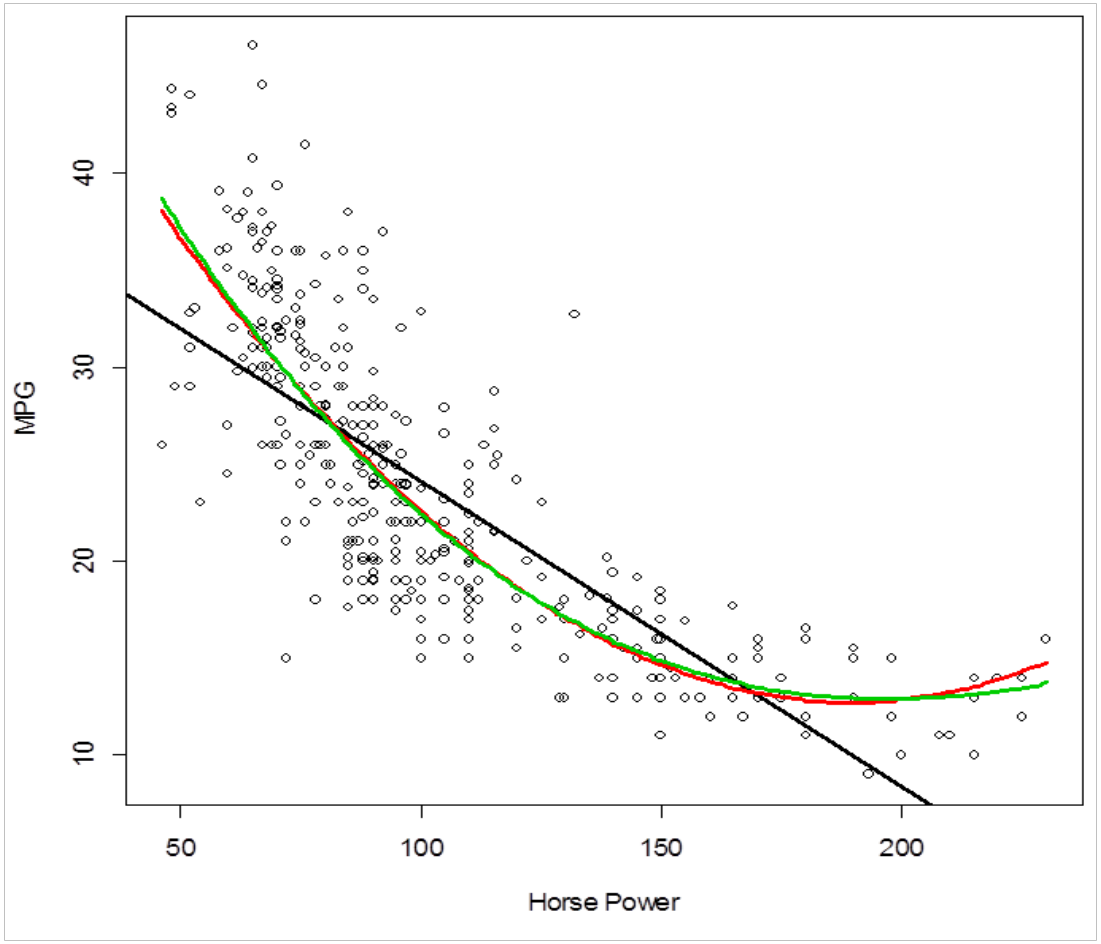
MPG vs Horse Power

- Quadratic regression: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$



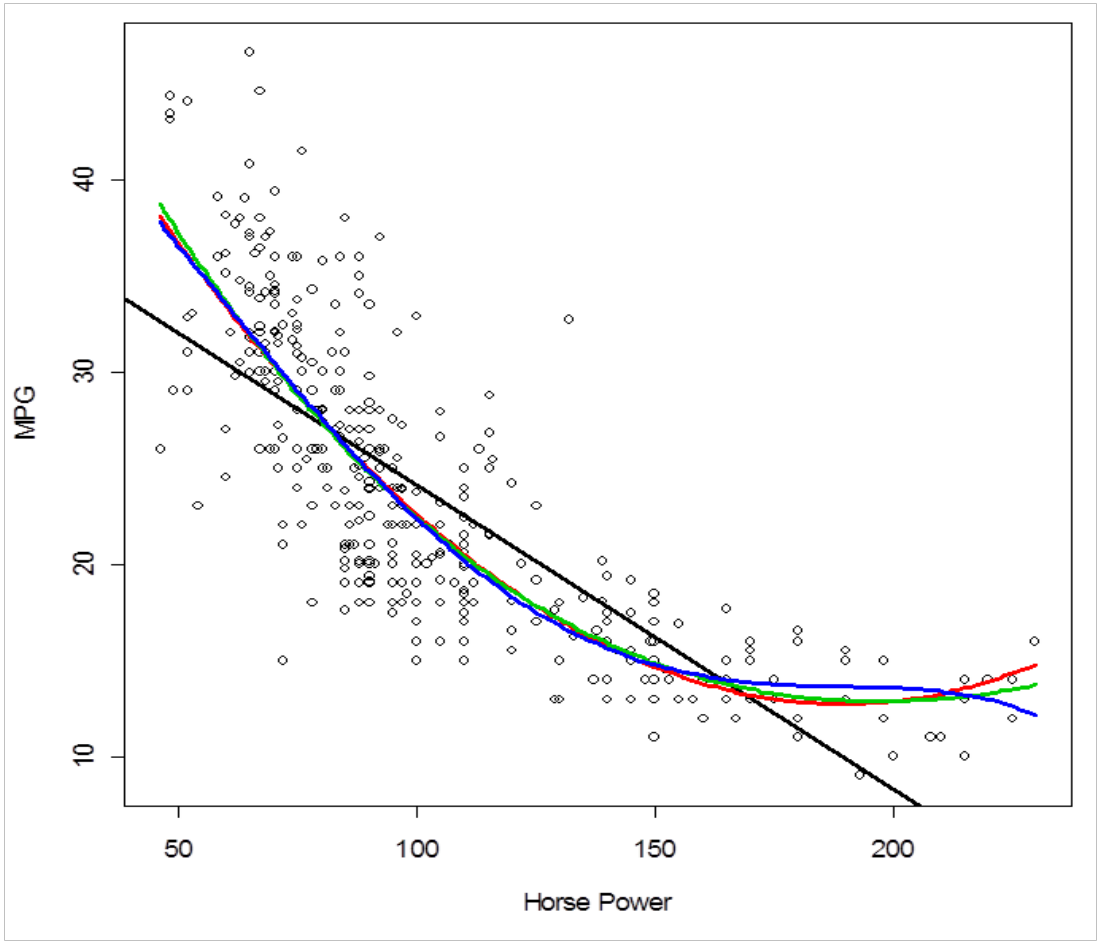
MPG vs Horse Power

- Cubic regression: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \varepsilon$



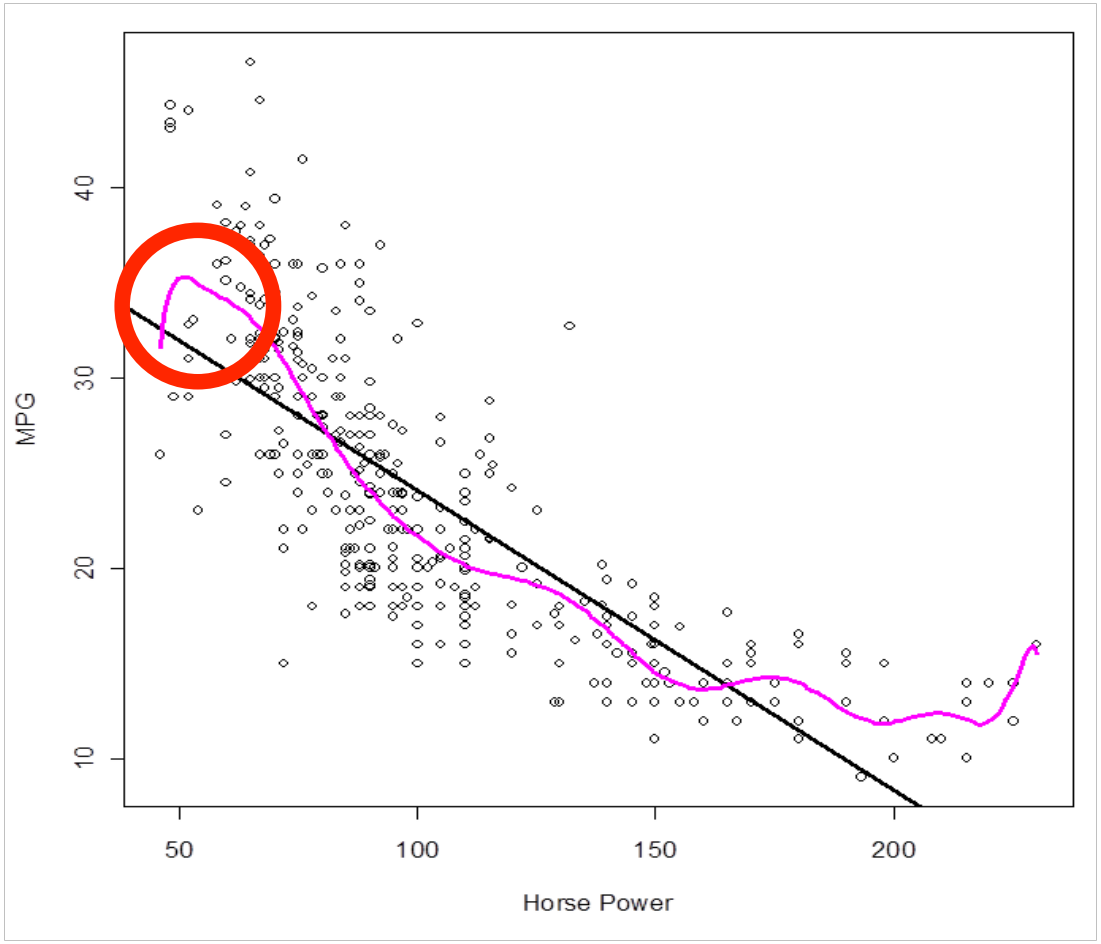
MPG vs Horse Power

- Polynomial of degree 4: $y = \beta_0 + \beta_1 x + \cdots + \beta_4 x^4 + \varepsilon$

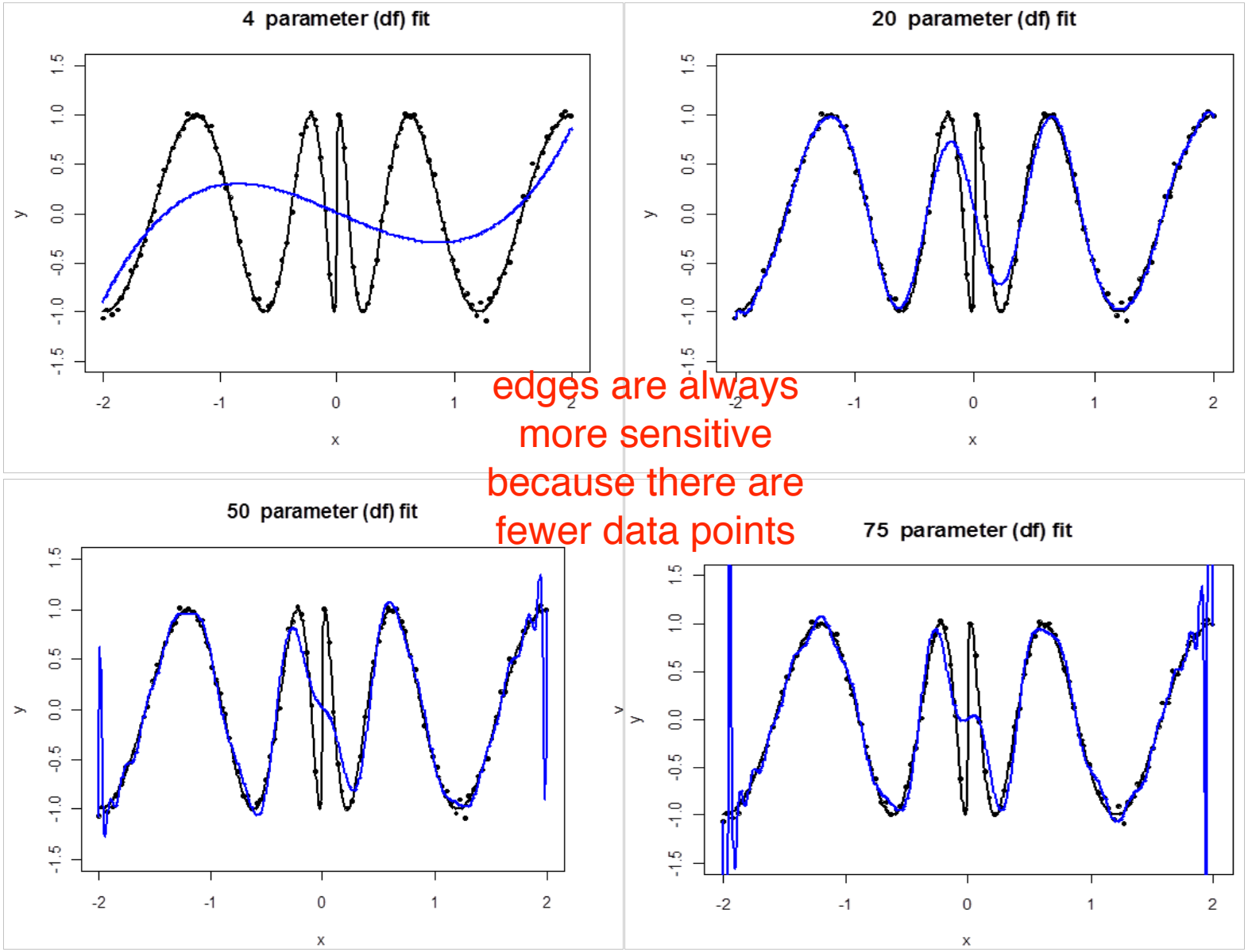


MPG vs Horse Power

- Polynomial of degree 14: $y = \beta_0 + \beta_1 x + \cdots + \beta_{14} x^{14} + \varepsilon$



A Hard Simulated Data Example



Basis Functions

- Polynomials are just one kind of basis functions.
- Polynomial regression:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d + \varepsilon$$

- More generally, we can model

$$y = \beta_0 + \beta_1 b_1(x) + \beta_2 b_2(x) + \cdots + \beta_d b_d(x) + \varepsilon$$

- For polynomial regression, where

$$b_j(x) = x^j$$

Other Basis Functions

- For any basis functions $b_j(x)$, once they are fixed, you can just use linear regression to estimate the β's
- Many different bases are used in practice (wavelets, Fourier transforms, etc).
- We will investigate one of the most common alternatives, regression splines.

Regression splines

- Key idea: fit different polynomials locally (over different regions of x).
- For example a cubic spline works by fitting a cubic $y = ax^3 + bx^2 + cx + d$ but the coefficients a, b, c and d change depending what part of the range of x we are looking at.
- The points where the coefficients change are called knots.
- The more knots, the more flexible the spline is.

A Simple Example

- Suppose x runs from 0 to 1 and we fit a cubic spline with one knot at 0.5:

$$y = \begin{cases} a_1 x^3 + b_1 x^2 + c_1 x + d_1 \text{ if } x \leq 0.5 \\ a_2 x^3 + b_2 x^2 + c_2 x + d_2 \text{ if } x > 0.5 \end{cases}$$

- Then we might have, for example,

a1*0.5^3+b1*0.5^2+c1*0.5+d1=a2*0.5^3+b2*0.5^2+c2*0.5+d2

$$a_1 = 1,\ b_1 = 4,\ c_1 = 2,\ d_1 = 0$$

$$a_2 = 2,\ b_2 = 1,\ c_2 = 0,\ d_2 = 1$$

- Then if $x = 0.25$ our prediction for y is

$$1 \times 0.25^3 + 4 \times 0.25^2 + 2 \times 0.25 + 0 = 0.766$$

- While if $x = 0.75$ our prediction for y is

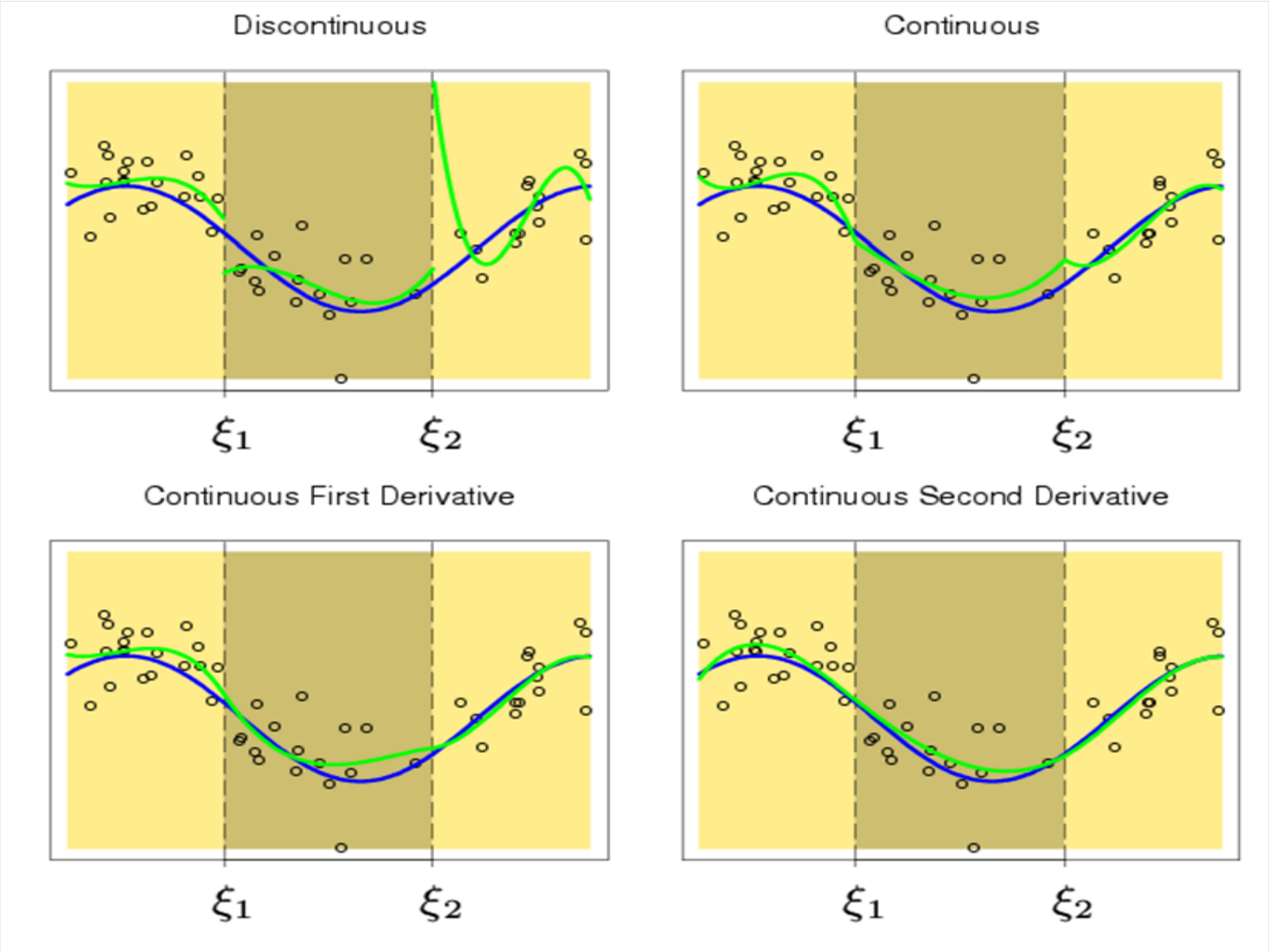
$$2 \times 0.75^3 + 1 \times 0.75^2 + 0 \times 0.75 + 1 = 2.406$$

Constraints

- It appears in this example there are 8 parameters (or degrees of freedom) for us to choose.
- However, in reality there are constraints.
- It makes sense to require that the values of different pieces match at the knots.
- For a smooth curve, we also require that the first and second derivatives match at the knots.
- How many free parameters do we have left?
- In general there are (#knots+4) free parameters to choose.

k knots=k+1 functions

Ensuring Smoothness in the Spline Fit



Splines Represented Using a Basis Function

- We can represent a spline with k knots using the basis function representation

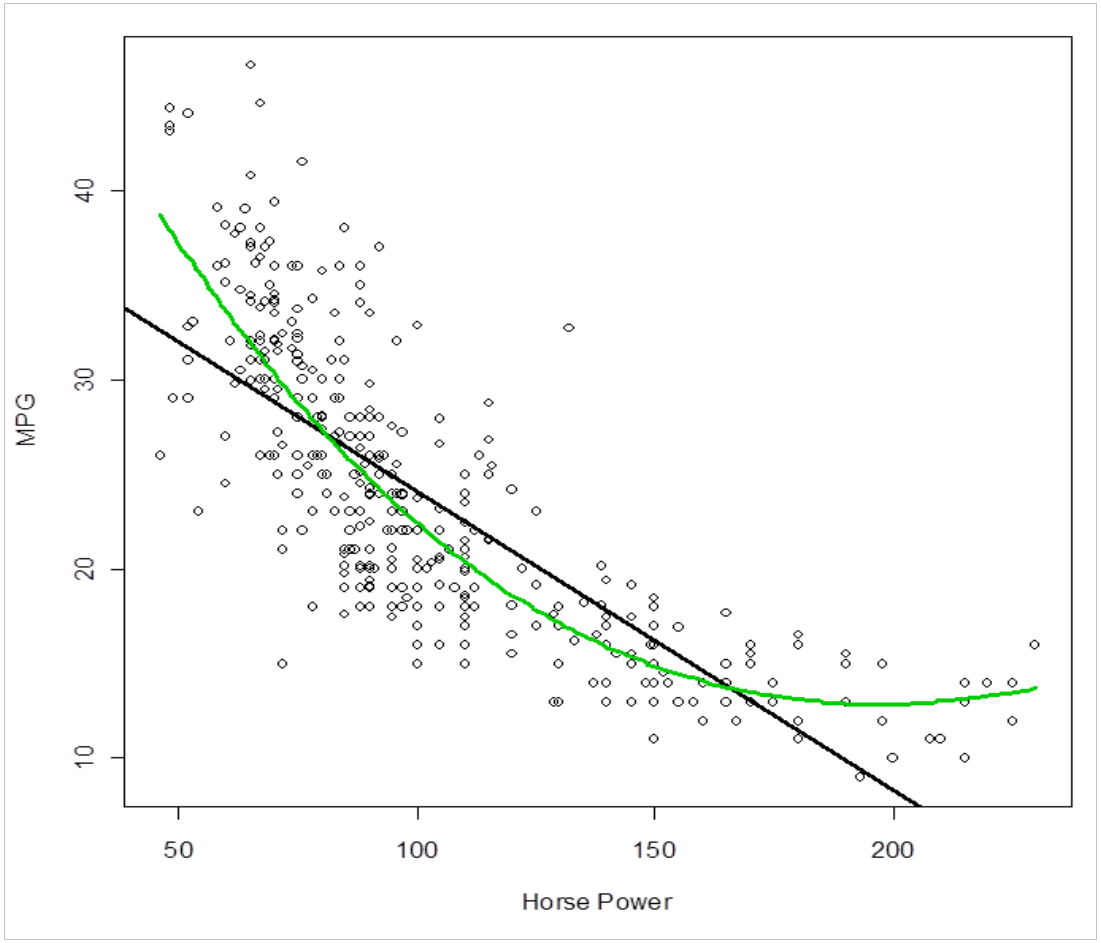
$$y = \beta_0 + \beta_1 b_1(x) + \beta_2 b_2(x) + \cdots + \beta_{k+3} b_{k+3}(x) + \varepsilon$$

- For example a spline with one knot could be written as

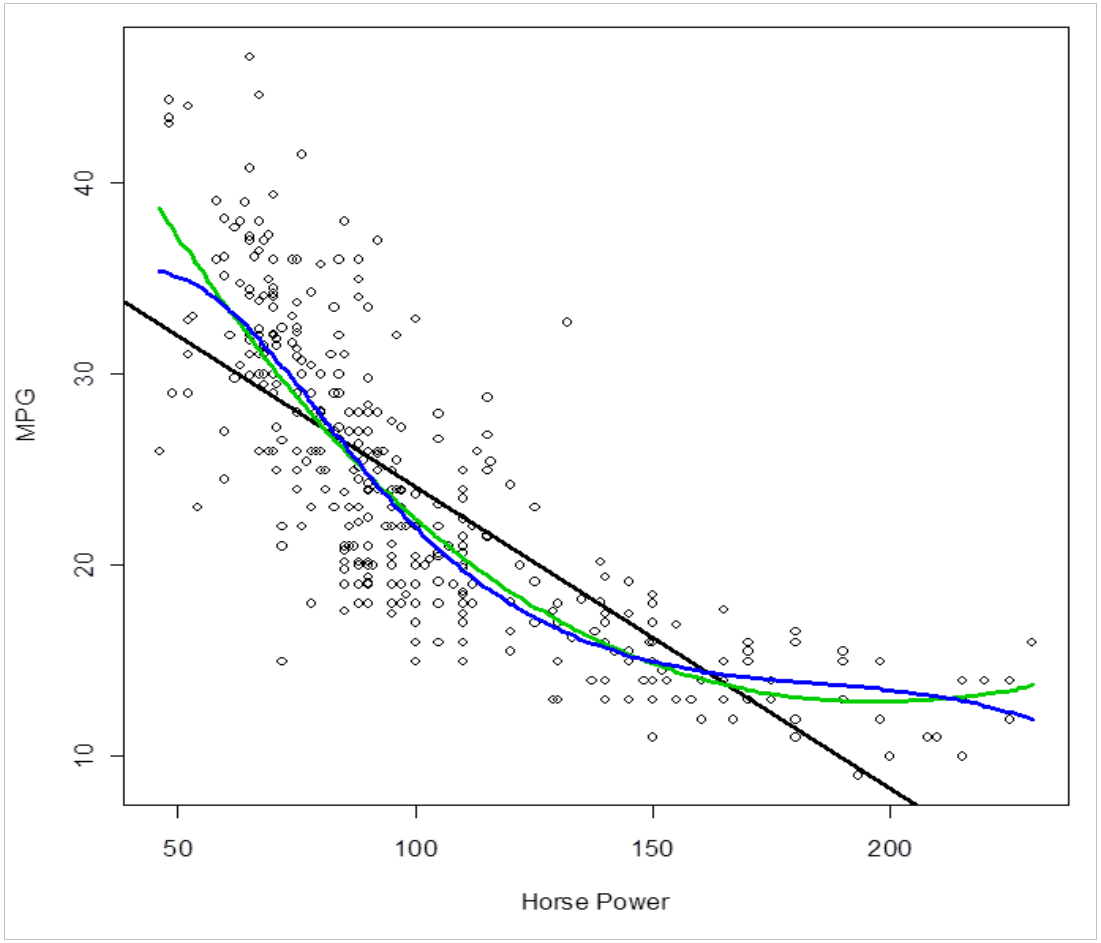
$$y = \beta_0 + \beta_1 b_1(x) + \beta_2 b_2(x) + \cdots + \beta_4 b_4(x) + \varepsilon$$

- The exact formula for $b_j(x)$ is complicated, but the computer has no trouble calculating it.
- Once we have $b_j(x_i)$ for each x_i we can use OLS to estimate the β's and hence fit the spline.

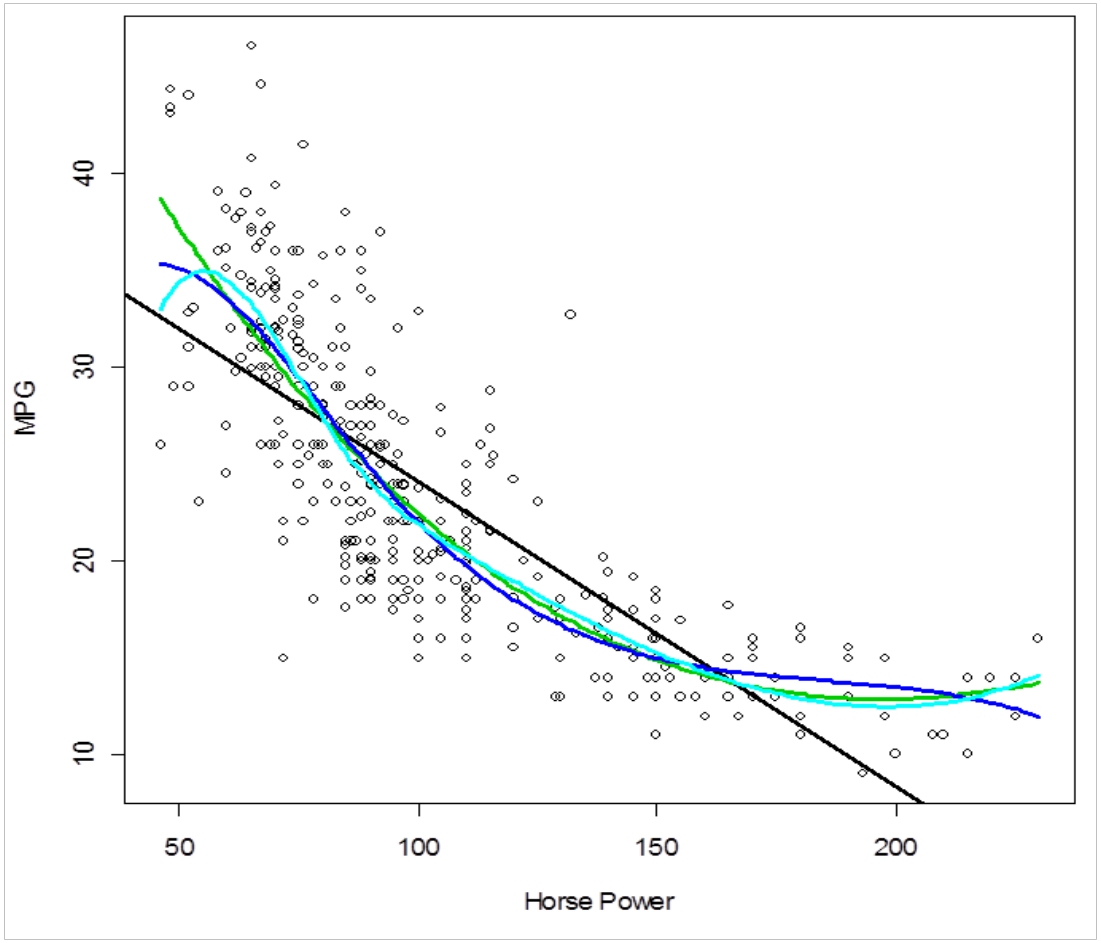
MPG vs Horse Power: 0 Knots



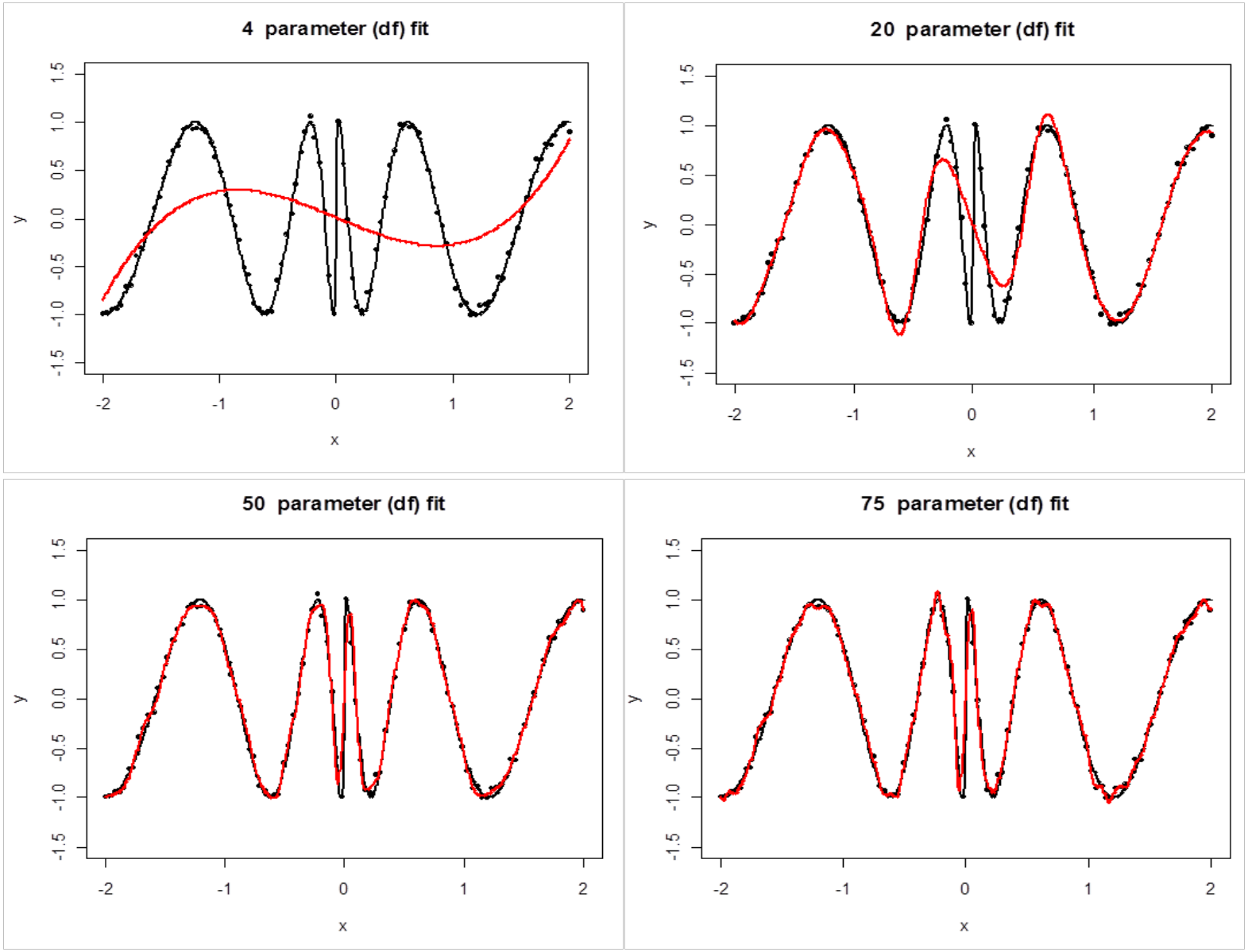
MPG vs Horse Power: 1 Knot



MPG vs Horse Power: 2 Knots

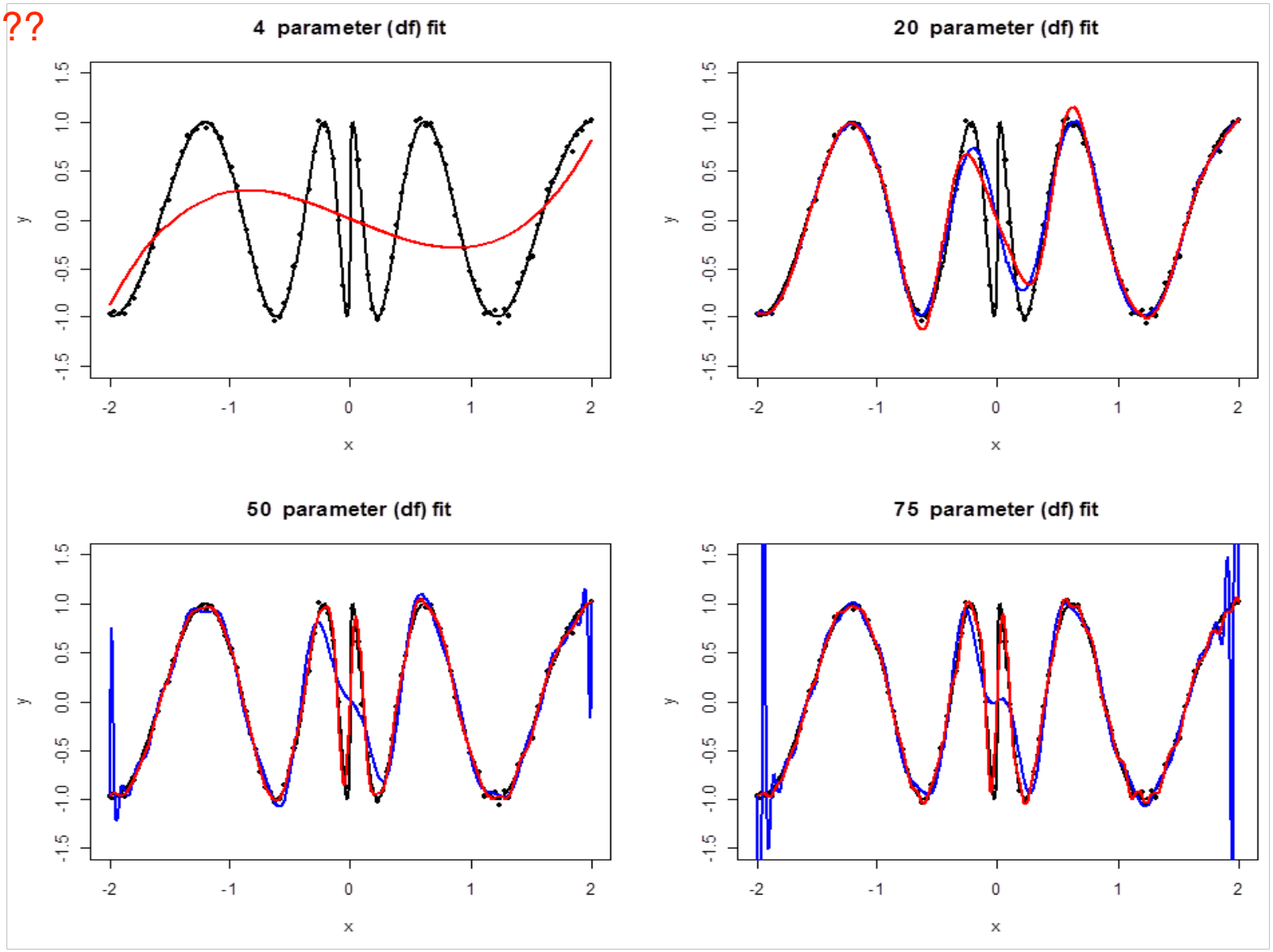


Regression Spline Simulation



Spline vs Polynomial Comparison

??

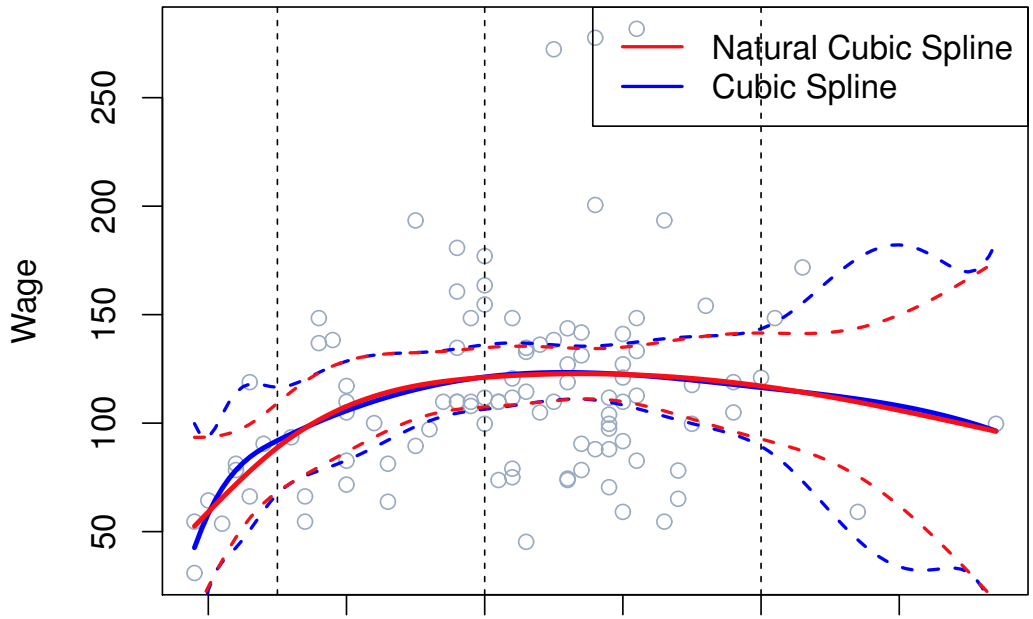


How many knots do we need?

- In principle, can use as many knots as we want
- If we know a region where the function changes rapidly, makes sense to put more knots there
- Conversely, ”flat” regions need fewer knots
- In practice, knots are often automatically placed at uniform quantiles of the data (e.g. 10%, 20%, etc).
- Can also choose the number of knots by cross-validation

Overfitting in splines

- Generally, too many knots will lead to overfitting
- The problem is especially visible at the ends (edge effects)
- Natural spline: replaces the ”end” cubic splines (one on each side) with lines.
- A natural spline cannot match derivatives, but gains stability on the boundary



Smoothing Splines

- Regression splines specify a set of knots, produce basis functions and then use least squares to estimate the spline coefficients.
- Smoothing splines are a somewhat different approach that also gives a smooth curve.
- What we really want to do is to find some function, say $g(x)$ such that it fits the observed data well, i.e.

$$\text{RSS} = \sum_{i=1}^{n} (y_i - g(x_i))^2$$

is small.

A Problem

a straight line is perfect msooth because second derivative is zero

- Without constraints on $g(x)$ then we can always set RSS equal to zero simply by choosing a $g(x_i) = y_i$ (interpolate all the y_i's).
- What we really want is a g that makes RSS small but is also smooth. How might we ensure smoothness?
- One natural approach is to find the g that minimizes

loss(fit) penalty(smoothness)

lambda->zero
lambda->inf

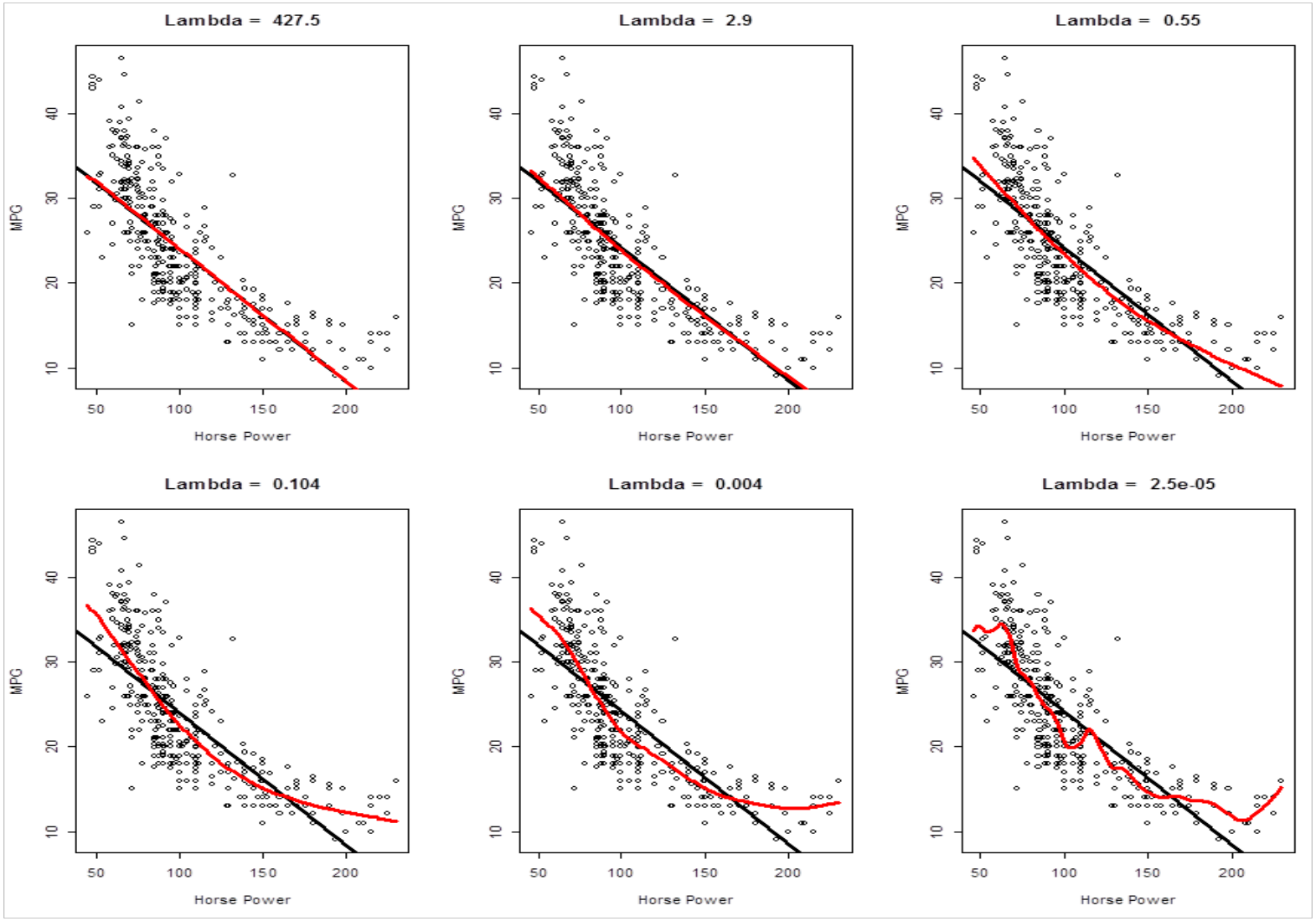
$$\sum_{i=1}^{n}(y_i - g(x_i))^2 + \lambda \int g''(t)^2 dt$$

first derivative: rate of chanfe

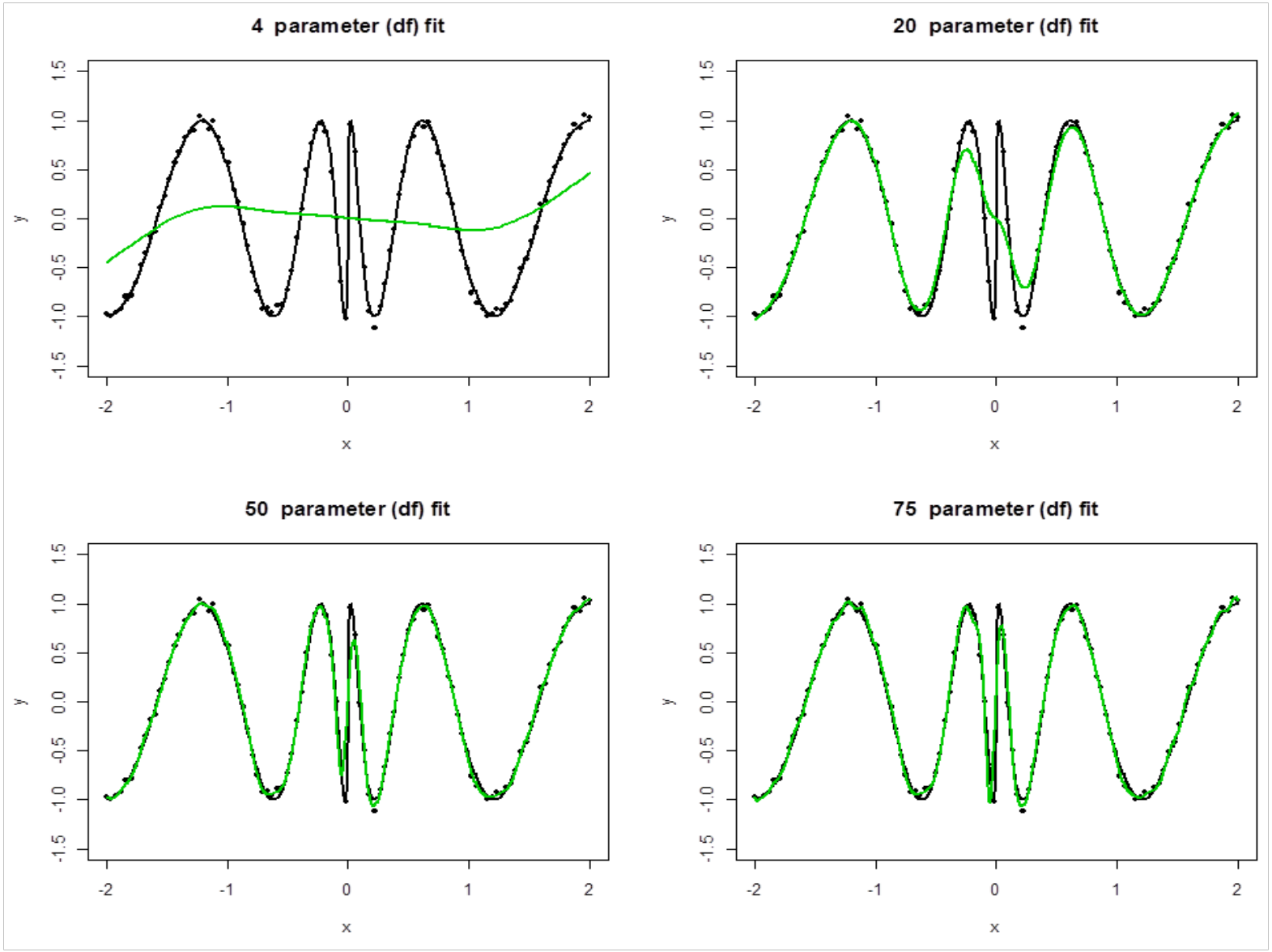
where λ is a tuning parameter greater than zero.

- Remarkably, one can show that the $g(x)$ that minimizes this quantity is a cubic spline! We call this type of spline a smoothing spline.

- As λ approaches infinity g becomes a straight line.
- As λ approaches zero g tries to interpolate all the points and becomes very wiggly.

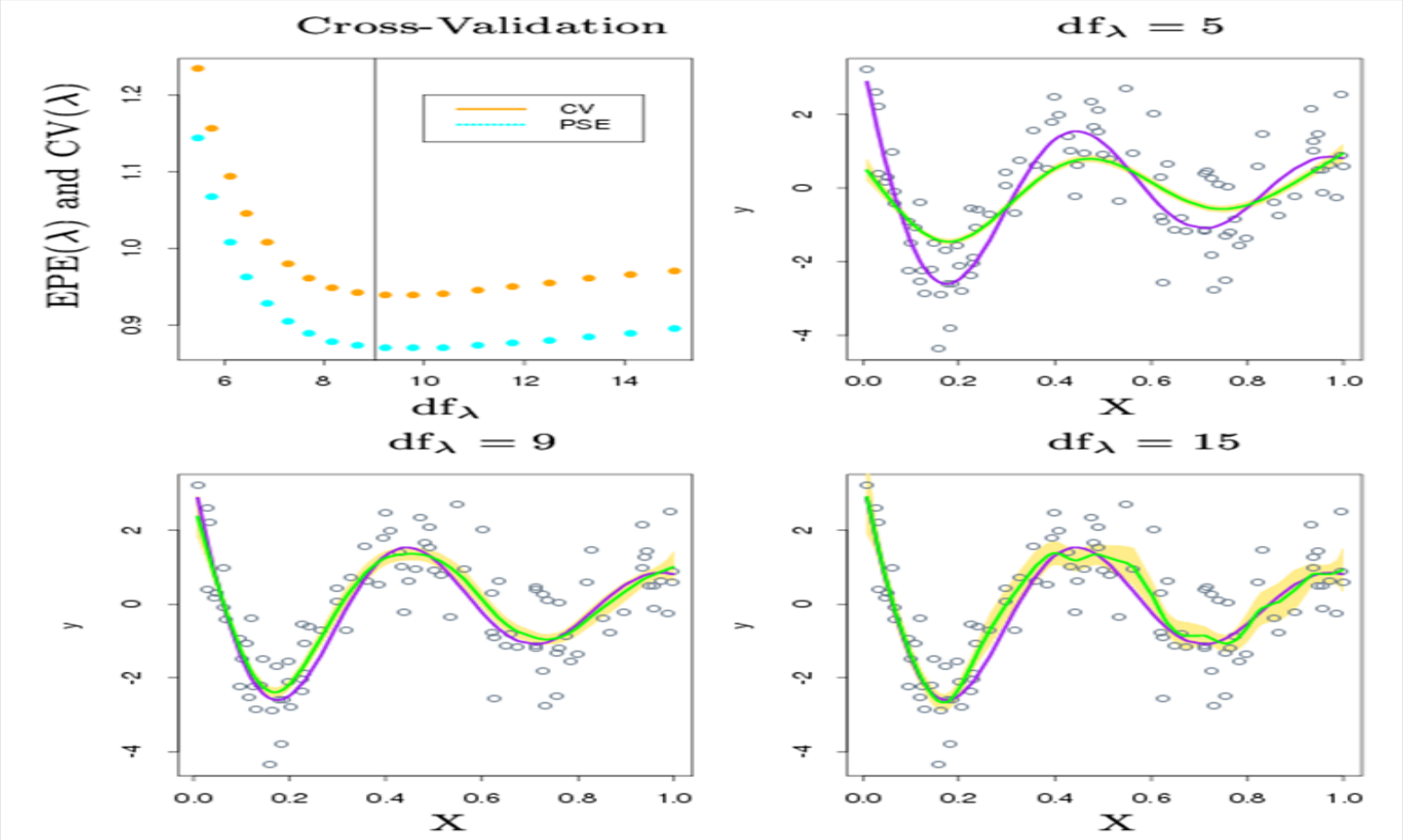


Smoothing Spline Simulation



Choosing the Smoothing Parameter

- How do we choose λ?
- We can use cross-validation again, i.e. find the value of λ that makes the cross-validated MSE as small as possible.



Summary on splines

- Locally cubic polynomials, but a lot more flexible
- Choosing the right number of degrees of freedom is important, or else can overfit - use CV
- Regression splines and smoothing splines are fit differently but often look similar