Nonparametrics and Local Methods: Splines

C.Conlon

February 27, 2021

Applied Econometrics

Polynomial Basis

Again consider the following relationship:

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

One approach is to approximate $f(x_i)$ or $E[y_i|x_i]$ with a polynomial series.

$$y_i = a_0^k + a_1^k x_i + a_2^k x_i^2 + a_3^k x_i^3 + \varepsilon_i \text{ for } x \in [\underline{x}_k, \overline{x}_k]$$

New idea: approximate $f(x_i)$ with different functions at different intervals of $[\underline{x}_k, \overline{x}_k]$.
Hard part: maintain that $\hat{f}(x_i)$ is twice continuously differentiable...

Splines are piecewise interpolating functions

Definition
A function $s(x)$ on $[lb, ub]$ is a spline of order m IFF

1. s is $\mathbb{C}^{m-2}$ on $[lb, ub]$ and
2. there is a grid of points (nodes) $lb = x_0 < x_1 < \cdots < x_k = ub$ such that $s(x)$ is a polynomial of degree $m-1$ on each subinterval $[x_k, x_{k+1}]$, $k = 0, \ldots, K-1$

Second order $(m = 2)$ is piecewise linear.
We usually use cubic splines.

Cubic Splines

- Lagrange data set (x_i, y_i) for $i = 0, \ldots n$.
- Nodes: the x_i are the nodes of the spline
- Functional form $s(x) = a_i + b_i x + c_i x^2 + d_i x^3$ on $[x_{i-1}, x_i]$
- Unknowns $4n$ unknown coefficients
- $2n$ interpolation and continuity conditions:

$$\begin{aligned} y_i &= a_i + b_i x_i + c_i x_i^2 + d_i x_i^3 \quad i = 1, \ldots, n \\ y_i &= a_{i+1} + b_{i+1} x_i + c_{i+1} x_i^2 + d_{i+1} x_i^3 \quad i = 0, \ldots, n-1 \end{aligned}$$

- $2n - 2$ conditions from $\mathbb{C}^2$ at the interior for $i = 1, \ldots, n-1$

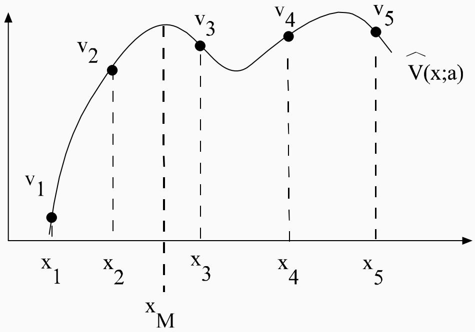
$$\begin{aligned} b_i + 2c_i x_i + 3d_i x_i^2 &= b_{i+1} + 2c_{i+1} x_i + 3d_{i+1} x_i^2 \\ 2c_i + 6d_i x_i &= 2c_{i+1} + 6d_{i+1} x_i \end{aligned}$$

Side Conditions

We have $4n-2$ linear equations and $4n$ unknowns we need two side conditions to identify the system

- Natural spline: $s''(x_0) = s''(x_n) = 0$ minimizes the total curvature $\int_{x_0}^{x_n} s''(x)^2 dx$
- Hermite spline: $s'(x_0) = y_0'$ and $s'(x_n) = y_n'$ (with extra data)
- Secant Hermite: $s'(x_0) = \frac{s(x_1)-s(x_0)}{x_1-x_0}$, $s'(x_n) = \frac{s(x_n)-s(x_{n-1})}{x_n-x_{n-1}}$
- Solvers are built in to packages like R (check documentation for which method).

Shape Issues



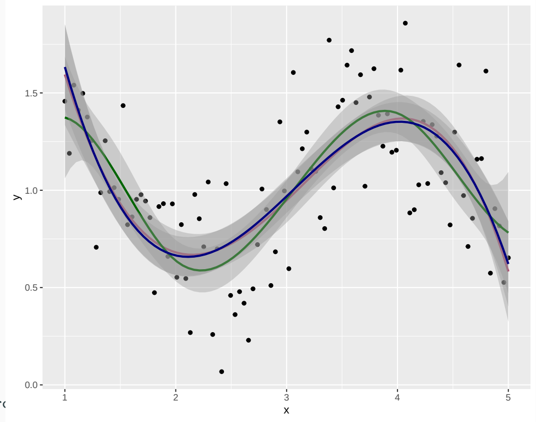
- Concave (monotone) data may lead to non concave (non monotone) approximations

Schumaker Procedure (Shape Preserving Splines)

1. Take level (and maybe slope) data at nodes x_k
2. Add intermediate nodes $z_k^+ \in [x_k, x_{k+1}]$
3. Run quadratic spline with nodes at the x and z nodes which interpolate data and preserves shape
4. Schumaker formulas tell you how to choose the z and spline coefficient
5. Detail in Judd and in companion paper (Judd and Solnick)

Spline Example

- Try two piecewise cubics (at $x = 3$)
- Try three piecewise cubics (at $x = (2, 4)$)
- Try single cubic



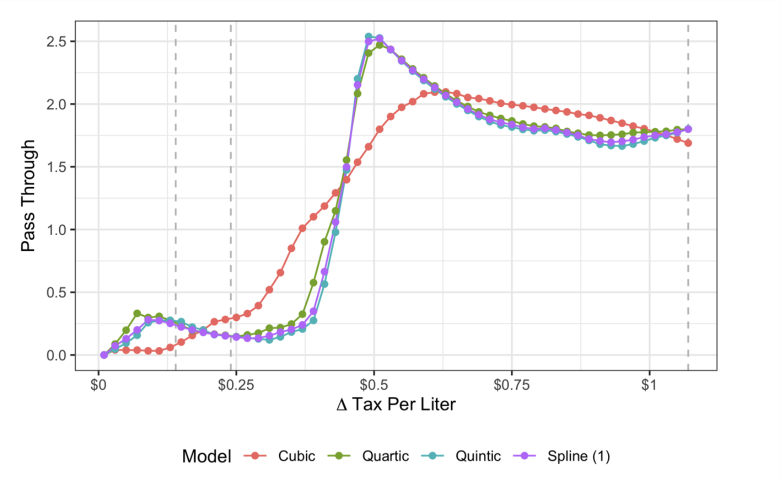

```
library(splines)
ggplot(data=NULL,aes(x, y)) + geom_point() +
  geom_smooth(method = "lm", formula = y ~ bs(x,knots=3) ,color='marc
  geom_smooth(method = "lm", formula = y ~ bs(x,knots=c(2,4)) ,color='darkgreen')+
  geom_smooth(method = "lm", formula = y ~ poly(x, 3),color='navy')
```

Spline Example

- Alternative is to use generalized additive model and fit a spline with the `s()` function.
- These can be made quite flexible (but this is simple).

```
library(mgcv)
ggplot(data=NULL,aes(x, y)) + geom_point() +
  geom_smooth(method = "lm", formula = y ~ bs(x,knots=3) ,color='maroon')+
  stat_smooth(method = gam, formula = y ~ s(x),color='navy')
```

My own example



Cubic isn't flexible enough, spline and quartic look about the same.