

Nonparametrics and Local Methods: Polynomials

C.Conlon

February 14, 2023

Applied Econometrics

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**.

$$y_i = a_0 + a_1x_i + a_2x_i^2 + a_3x_i^3 + \cdots + a_Mx_i^M + \epsilon_i$$

An important choice is to choose polynomial order M (complexity)

Idea: we can approximate **arbitrary (smooth) functions** $f(x_i)$ with a high-order polynomial.

Polynomial Example

Let's suppose we want to approximate the following function:

$$\frac{1}{3} \sin(2x) \approx \frac{2}{3}x - \frac{4}{9}x^3 + \frac{4}{45}x^5 + O(x^7)$$

This should be easy since we have a **Taylor Expansion** that is polynomial in x (also it is odd).

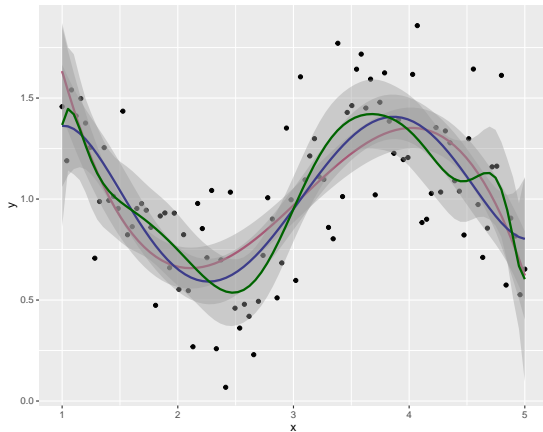
Polynomial Example

```
library(ggplot2)

# set the seed to make the results reproducible.
set.seed(3)

#### simulate some data ####
# epsilon = random error term
epsilon <- 0.25*rnorm(100)
x      <- seq(from=1, to=5, length.out=100)
y      <- 1 +sin(x*2)/3 + epsilon

# visualize the data (with a polynomial best-fit line)
ggplot(data=NULL,aes(x, y)) + geom_point() +
  geom_smooth(method = "lm", formula = y ~ poly(x, 3,raw=T),color='maroon')+
  geom_smooth(method = "lm", formula = y ~ poly(x, 5,raw=T),color='navy')+
  geom_smooth(method = "lm", formula = y ~ poly(x, 17,raw=T),color='darkgreen')
```



Polynomial Example

- ▶ $M = 5$ order polynomial should fit the data well (it does).
- ▶ We are clearly **overfitting** at $M = 17$ since this doesn't look much like $y(x) = \sin(2x)/3 + \varepsilon$.
- ▶ We used **raw polynomials**: x, x^2, x^3, \dots
- ▶ By default R uses **orthogonal polynomials** when we drop `raw=TRUE`.
(More on these later)

```
# visualize the data (with a polynomial best-fit line)
ggplot(data=NULL,aes(x, y)) + geom_point() +
  geom_smooth(method = "lm", formula = y ~ poly(x, 3),color='maroon')+
  geom_smooth(method = "lm", formula = y ~ poly(x, 5),color='navy')+
  geom_smooth(method = "lm", formula = y ~ poly(x, 17),color='darkgreen')
```

“Raw” Polynomials

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-5.421717	8.009814	-0.677	0.500
poly(x, 6, raw = T)1	18.072018	20.371055	0.887	0.377
poly(x, 6, raw = T)2	-17.392013	20.363051	-0.854	0.395
poly(x, 6, raw = T)3	7.503790	10.288970	0.729	0.468
poly(x, 6, raw = T)4	-1.561290	2.786268	-0.560	0.577
poly(x, 6, raw = T)5	0.148442	0.385418	0.385	0.701
poly(x, 6, raw = T)6	-0.004826	0.021378	-0.226	0.822

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	45.71304	19.95109	2.291	0.02423 *
poly(x, 7, raw = TRUE)1	-138.78101	59.74402	-2.323	0.02239 *
poly(x, 7, raw = TRUE)2	177.80275	72.90420	2.439	0.01665 *
poly(x, 7, raw = TRUE)3	-120.61050	47.13566	-2.559	0.01214 *
poly(x, 7, raw = TRUE)4	46.52356	17.50188	2.658	0.00926 **
poly(x, 7, raw = TRUE)5	-10.21667	3.74637	-2.727	0.00765 **
poly(x, 7, raw = TRUE)6	1.18842	0.42965	2.766	0.00686 **
poly(x, 7, raw = TRUE)7	-0.05682	0.02044	-2.780	0.00658 **

- ▶ 6th order polynomial: nothing significant
- ▶ 7th order polynomial: all terms significant (odd and even).
- ▶ Totally different coefficients!
- ▶ Both highly sensitive to small changes in x_i .

“Orthogonal” Polynomials

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.02278	0.02779	36.803	< 2e-16 ***
poly(x, 6, raw = FALSE)1	0.64772	0.27791	2.331	0.02193 *
poly(x, 6, raw = FALSE)2	0.48158	0.27791	1.733	0.08643 .
poly(x, 6, raw = FALSE)3	-2.47613	0.27791	-8.910	4.14e-14 ***
poly(x, 6, raw = FALSE)4	-0.16435	0.27791	-0.591	0.55571
poly(x, 6, raw = FALSE)5	0.79114	0.27791	2.847	0.00543 **
poly(x, 6, raw = FALSE)6	-0.06273	0.27791	-0.226	0.82190

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.02278	0.02684	38.111	< 2e-16 ***
poly(x, 7, raw = FALSE)1	0.64772	0.26837	2.414	0.01778 *
poly(x, 7, raw = FALSE)2	0.48158	0.26837	1.794	0.07602 .
poly(x, 7, raw = FALSE)3	-2.47613	0.26837	-9.227	9.68e-15 ***
poly(x, 7, raw = FALSE)4	-0.16435	0.26837	-0.612	0.54179
poly(x, 7, raw = FALSE)5	0.79114	0.26837	2.948	0.00405 **
poly(x, 7, raw = FALSE)6	-0.06273	0.26837	-0.234	0.81569
poly(x, 7, raw = FALSE)7	-0.74618	0.26837	-2.780	0.00658 **

- ▶ Odd terms are significant in both specifications
- ▶ Coefficients appear stable (!)
- ▶ Usually coefficients decline in (odd) polynomial order.
- ▶ But very high dimensional polynomials will still **overfit**.

Orthogonal Polynomials

Consider an arbitrary basis:

$$y(x_i) = a_0 + \sum_{j=1}^M a_j b_j(x_i) + \varepsilon_i$$

- ▶ $b_j(x_i) = x_i^j$ (regular polynomials)
- ▶ $\langle b_j(x), b_k(x) \rangle = 0$ for $j \neq k$ (orthogonal polynomials).
- ▶ Lots of options: Chebyshev, Legendre, Fourier, Gram Schmidt (discuss later).

What R does: Gram Schmidt

Let $a_j = x^j$ (the raw polynomial) and then **orthogonalize** as follows:

$$\hat{p}_j(x) = a_j(x) - \sum_{k=0}^{j-1} p_k(x) \frac{p_k \cdot a_j}{p_k \cdot p_k}, \quad p(x) = \frac{\hat{p}(x)}{\hat{p}(1)}$$

- ▶ We are forming the **residual** of x^j on $(x^{j-1}, x^{j-2}, \dots, x, 1)$ (where each term has already been residualized).
- ▶ Notice, we don't need to know y_i , we can simply residualized x_i against powers of itself.
- ▶ We could do this for any matrix X ! (doesn't need to be powers of x_i).

Orthogonal Polynomials

General Case

- ▶ Space: polynomials over domain D
- ▶ Weighting function: $w(x) > 0$ (positive everywhere)
- ▶ Inner product $\langle f, g \rangle = \int_D f(x)g(x)w(x)dx$
- ▶ Polynomials are orthogonal wrt to $w(x)$ IFF

$$\langle \phi_i, \phi_j \rangle = 0, \quad i \neq j$$

- ▶ Can compute orthogonal polynomials using recurrence formulas

$$\phi_0(x) = 1$$

$$\phi_1(x) = x$$

$$\phi_{k+1}(x) = (a_{k+1}x + b_k)\phi_k(x) + c_{k+1}\phi_{k-1}(x)$$

Chebyshev Polynomials

- ▶ Can compute orthogonal polynomials using recurrence formulas
- ▶ $[a, b] = [-1, 1]$ and $w(x) = (1 - x^2)^{-1/2}$
- ▶ $T_n(x) = \cos(n \cos^{-1} x)$
- ▶ Recursive Definition

$$T_0(x) = 1$$

$$T_1(x) = x$$

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

General Intervals

- ▶ Most problems aren't on the $[-1, 1]$ interval so we need a COV

$$y = -1 + 2 \frac{x-a}{b-a}$$

- ▶ Polynomials $\phi_i^*(x) \equiv \phi_i(-1 + 2 \frac{x-a}{b-a})$ are orthogonal over $x \in [a, b]$ with respect to the weight $w^*(x) \equiv (-1 + 2 \frac{x-a}{b-a})$ iff the $\phi_i(y)$ are orthogonal over $y \in [-1, 1]$ wrt $w(y)$.

Chebyshev Approximation Algorithm

1. Compute the $m \geq n + 1$ Chebyshev nodes on $[-1, 1]$

$$z_k = -\cos\left(\frac{2k-1}{2m}\pi\right), \quad k = 1, \dots, m$$

2. Adjust the nodes to $[a, b]$ interval

$$x_k = (z_k + 1)\left(\frac{b-a}{2}\right) + a, \quad k = 1, \dots, m$$

3. Evaluate f at the nodes $w_k = f(x_k)$ for $k = 1, \dots, m$
4. Compute the coefficients a_i to get the approximation $p(x)$

$$a_i = \frac{\sum_{k=1}^m w_k T_i(z_k)}{\sum_{k=1}^m T_i(z_k)^2}, \quad p(x) = \sum_{i=0}^n a_i T_i\left(2\frac{x-a}{b-a} - 1\right)$$

Minimax Approximation

- ▶ Data: (x_i, y_i) , $i = 1, \dots, n$
- ▶ Objective: L^∞ fit

$$\min_{\beta \in R^m} \max_i \|y_i - f(x_i; \beta)\|$$

- ▶ Difficult to do (minimax problems are non-convex)
- ▶ Chebyshev Approximation satisfies this property, for C^2, C^3 functions but doesn't get $f(x)$ right!

Theorem

Suppose $f: [-1, 1] \rightarrow R$ is C^k for some $k \geq 1$, and let I_n be the degree n polynomial interpolation of f based at the zeroes of $T_{n+1}(x)$ then

$$\|f - I_n\|_\infty \leq \left(\frac{2}{\pi} \log(n+1) + 1 \right) \frac{(n-k)!}{n!} \left(\frac{\pi}{2} \right)^k \left(\frac{b-a}{2} \right)^k \|f^{(k)}\|_\infty$$

Recap: Polynomials

$$y(x_i) = a_0 + \sum_{j=1}^M a_j b_j(x_i) + \varepsilon_i$$

- ▶ Thus far have looked at **global polynomial approximations**.
- ▶ Global: basis $b_j(x)$ and coefficients a_j are the same for any value of x .
- ▶ Can choose polynomial order with **cross validation**, later we will explore **penalization**.
- ▶ Special case of **sieve estimators**: we let the order $M \rightarrow \infty$ as $n \rightarrow \infty$ but not as quickly!