Numerical Methods in Economics

MIT Press, 1998

Notes for Lecture 13: Approximation Methods

March 30, 2020

Approximation Methods

- General Objective: Given data about a function $f(x)$ (which is difficult to compute) construct a simpler function $g(x)$ that approximates $f(x)$.

- Questions:
  - What data should be produced and used?
  - What family of "simpler" functions should be used?
  - What notion of approximation do we use?
  - How good can the approximation be?
  - How simple can a good approximation be?

- Comparisons with statistical regression
  - Both approximate an unknown function
  - Both use a finite amount of data
  - Statistical data is noisy; we assume here that data errors are small
  - Nature produces data for statistical analysis; we produce the data in function approximation
  - Our approximation methods are like experimental design with very small experimental error

Local Approximation Methods

General approach:

- Use information about $f : R \to R$ only at a point, $x_0 \in R$, to construct an approximation valid near x_0
- Taylor Series Approximation

$$\begin{aligned} f(x) &\doteq f(x_0) + (x - x_0)\, f'(x_0) + \frac{(x - x_0)^2}{2}\, f''(x_0) + \cdots \\ &+ \frac{(x - x_0)^n}{n!}\, f^{(n)}(x_0) + \mathcal{O}(|x - x_0|^{n+1}) \\ &= p_n\,(x) + \mathcal{O}(|x - x_0|^{n+1}) \end{aligned} \tag{6.1.1}$$

- Power series: $\sum_{n=0}^{\infty} a_n z^n$
  - The *radius of convergence* is

$$r = \sup\{|z| : |\sum_{n=0}^{\infty} a_n z^n| < \infty\},$$

  - The series $\sum_{n=0}^{\infty} a_n z^n$ converges for all $|z| < r$ and diverges for all $|z| > r$.

- Complex analysis

  – $f : \Omega \subset C \to C$ on the complex plane C is *analytic* on Ω iff

  $$\forall a \in \Omega \ \ \exists r, c_k \left(\forall \, \|z - a\| < r \left(f(z) = \sum_{k=0}^{\infty} c_k (z - a)^k \right) \right)$$

  – A *singularity of* f is any a s. t. f is analytic on $\Omega - \{a\}$ but not on Ω.

  – If f or any derivative of f has a singularity at $z \in C$, then the radius of convergence in C of $\sum_{n=0}^{\infty} \frac{(x-x_0)^n}{n!} \, f^{(n)}(x_0)$, is bounded above by $\| \, x_0 - z \, \|$.

- Example: $f(x) = x^\alpha$ where $0 < \alpha < 1$.
  - One singularity at $x = 0$
  - Radius of convergence for power series around $x = 1$ is 1.
  - Taylor series coefficients decline slowly:

$$a_k = \frac{1}{k!}\frac{d^k}{dx^k}(x^\alpha)|_{x=1} = \frac{\alpha(\alpha-1)\cdots(\alpha-k+1)}{1\cdot 2\cdot\cdots\cdot k}.$$

Table 6.1 (corrected): Taylor Series Approximation Errors for $x^{1/4}$

	Taylor series error				$x^{1/4}$
x N:	5	10	20	50	
3.0	5(−1)	8(1)	3(3)	1(12)	1.3161
2.0	1(−2)	5(−3)	2(−3)	8(−4)	1.1892
1.8	4(−3)	5(−4)	2(−4)	9(−9)	1.1583
1.5	2(−4)	3(−6)	1(−9)	0(−12)	1.1067
1.2	1(−6)	2(−10)	0(−12)	0(−12)	1.0466
.80	2(−6)	3(−10)	0(−12)	0(−12)	.9457
.50	6(−4)	9(−6)	4(−9)	0(−12)	.8409
.25	1(−2)	1(−3)	4(−5)	3(−9)	.7071
.10	6(−2)	2(−2)	4(−3)	6(−5)	.5623
.05	1(−1)	5(−2)	2(−2)	2(−3)	.4729

Rational Approximation

- Definition: A (m, n) Padé approximant of f at x_0 is a rational function

$$r(x) = \frac{p(x)}{q(x)},$$

where degree of p $(q)$is at most m (n), and

$$0 = \frac{d^k}{dx^k}(p - f\ q)\,(x_0), \quad k = 0, \cdots, m+n.$$

- Construction

  - Usually choose $m = n$ or $m = n + 1$.
  - The $m+1$ coefficients of p and the $n+1$ coefficients of q must satisfy linear conditions

$$p^{(k)}(x_0) = (f\ q)^{(k)}(x_0), \quad k = 0, \cdots, m+n, \tag{6.1.2}$$

  - (6.1.2) plus $q(x_0) = 1$ forms $m+n+2$ linear conditions on the $m+n+2$ coefficients

Linear system may be singular; if so, reduce n or m by 1

- Example: (2,1) Pade approx. of $x^{1/4}$ at $x=1$
  - Construct degree $m+n=2+1=3$ Taylor series

$$t(x) = 1 + \frac{(x-1)}{4} - \frac{3(x-1)^2}{32} + \frac{7(x-1)^3}{128} \equiv t(x).$$

  - Find p_0, p_1, p_2, and q_1 such that

$$p_0 + p_1(x-1) + p_2(x-1)^2 - t(x)(1+q_1(x-1)) = 0 \tag{6.1.3}$$

  - Combine coefficients of like powers in (6.1.3) implies

$$\frac{21+70x+5x^2}{40+56x}. \tag{6.1.4}$$

- Pade approximation is often better; not limited by singularities

Log-Linearization

- Log-linear approximation
  - Suppose we have an equation defining x in terms of ε.

$$f\,(x, \varepsilon) = 0$$

  - Implicit differentiation implies

$$\hat{x} = \frac{dx}{x} = -\frac{\varepsilon f_{\varepsilon}}{x f_x}\frac{d\varepsilon}{\varepsilon} = -\frac{\varepsilon f_{\varepsilon}}{x f_x}\;\varepsilon,$$

Since $\hat{x} = d(\ln x)$, log-linearization implies log-linear approximation

$$\ln x - \ln x_0 \doteq -\frac{\varepsilon_0 f_{\varepsilon}(x_0, \varepsilon_0)}{x_0 f_x(x_0, \varepsilon_0)}(\ln \varepsilon - \ln \varepsilon_0). \tag{6.1.5}$$

which implies

$$x \doteq x_0 \exp\left(-\frac{\varepsilon_0 f_{\varepsilon}(x_0, \varepsilon_0)}{x_0 f_x(x_0, \varepsilon_0)}(\ln \varepsilon - \ln \varepsilon_0)\right), \tag{6.1.6}$$

- Generalization to nonlinear change of variables.
  - Suppose $Y(X)$ implicitly defined by $f(Y(X), X) = 0$.
  - Define $x = \ln X$ and $y = \ln Y$, then $y(x) = \ln Y(e^x)$.
  - $y'(x) = \frac{d \ln Y}{d \ln X}$
  - $f(Y(X), X) = 0$ is equivalent to $f(e^{y(x)}, e^x) \equiv g(y(x), x) = 0$.
  - Implicit differentiation of $g(y(x), x) = 0$ will produce the value of $y'(x)$
  - $\ln Y(X) = y(x)$ suggests the linear approximation

$$\ln Y(X) = y(x) \doteq y(x_0) + y'(x)(x - x_0), \tag{6.1.7a}$$

  - $\ln Y(X) = y(x)$ also suggests the second-order approximation

$$\ln Y(X) = y(x) \doteq y(x_0) + y'(x)(x - x_0) + y''(x_0)\frac{(x - x_0)^2}{2}, \tag{6.1.7b}$$

- Can construct Padé expansions in terms of the logarithm.
- There is nothing special about log function.
  * Take any monotonic $h(\cdot)$
  * Define $x = h(X)$ and $y = h(Y)$
  * Use the identity

$$\begin{aligned} f(Y,X) &= f(h^{-1}(h(Y)), h^{-1}(h(X))) \\ &= f(h^{-1}(y), h^{-1}(x)) \\ &\equiv g(y,x). \end{aligned}$$

to generate expansions

$$\begin{aligned} y(x) &\doteq y(x_0) + y'(x)(x - x_0) + ... \\ Y(X) &\doteq h^{-1}\left(y(h(X_0)) + y'(h(X_0))(h(X) - h(X_0)) + ...\right) \end{aligned}$$

  * $h(z) = \ln z$ is natural for economists, but others may be better globally

Global Approximation Methods

Interpolation

- Interpolation Approach: find a function from an n-dimensional family of functions which exactly fits n data items

- Lagrange polynomial interpolation

  - Data: $(x_i, y_i)\,, i = 1, .., n.$
  - Objective: Find a polynomial of degree $n-1$, $p_n(x)$, which agrees with the data, i.e.,

$$y_i = f(x_i),\ \ i = 1, .., n$$

Result: If the x_i are distinct, there is a unique interpolating polynomial

- Question: Suppose that $y_i = f(x_i)$. Does $p_n(x)$ converge to $f(x)$ as we use more points?
- Convergence Counterexample
  - Suppose
$$f(x) = \frac{1}{1+x^2}, \quad x_i : \text{uniform on } [-5,5]$$
  - Interpolation does not work well

- Hermite polynomial interpolation

  - Data: $(x_i, y_i, y_i')\,, i = 1, .., n.$
  - Objective: Find a polynomial of degree $2n-1$, $p(x)$, which agrees with the data, i.e.,

$$
\begin{aligned}
y_i &= p(x_i),\ \ i = 1, .., n \\
y_i' &= p'(x_i),\ \ i = 1, .., n
\end{aligned}
$$

  - Result: If the x_i are distinct, there is a unique interpolating polynomial

- Least squares approximation

  - Data: A function, $f(x)$.
  - Objective: Find a function $g(x)$ from a class G that best approximates $f(x)$, i.e.,

$$
g = \arg\max_{g \in G} \|f - g\|^2
$$

Orthogonal polynomials

- General orthogonal polynomials
  - Space: polynomials over domain D
  - weighting function: $w(x) > 0$
  - Inner product: $\langle f, g \rangle = \int_D f(x)g(x)w(x)dx$
  - Definition: $\{\phi_i\}$ is a family of orthogonal polynomials w.r.t $w(x)$ iff

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

  - We like to compute orthogonal polynomials using recurrence formulas

$$\begin{aligned}
\phi_0(x) &= 1 \\
\phi_1(x) &= x \\
\phi_{k+1}(x) &= (a_{k+1}x + b_k)\,\phi_k(x) + c_{k+1}\phi_{k-1}(x)
\end{aligned}$$

  - Approximation (assuming $\|\phi_i\| = 1$):

$$\begin{aligned}
f(x) &= \sum_{i=0}^{\infty} a_i \phi_i \\
a_i &= \langle f, \phi_i \rangle = \int_D f(x)\phi_i(x)\,w(x)dx, \; i \neq j
\end{aligned}$$

- Legendre polynomials
  - $[a, b] = [-1, 1]$
  - $w(x) = 1$
  - $P_n(x) = \frac{(-1)^n}{2^n n!} \frac{d^n}{dx^n} \left[(1 - x^2)^n\right]$
  - Recurrence formula:

$$
\begin{aligned}
P_0(x) &= 1 \\
P_1(x) &= x \\
P_{n+1}(x) &= \frac{2n+1}{n+1}\, x\, P_n(x) - \frac{n}{n+1}\, P_{n-1}(x),
\end{aligned}
$$

- Chebyshev polynomials
  - $[a, b] = [-1, 1]$
  - $w(x) = \left(1 - x^2\right)^{-1/2}$
  - $T_n(x) = \cos(n \cos^{-1} x)$
  - Recurrence formula:

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{n+1}(x) &= 2x\, T_n(x) - T_{n-1}(x), \end{aligned}$$

- Laguerre polynomials
  - $[a, b] = [0, \infty)$
  - $w(x) = e^{-x}$
  - $L_n(x) = \frac{e^x}{n!} \, \frac{d^n}{dx^n} \, (x^n \, e^{-x})$
  - Recurrence formula:

$$
\begin{aligned}
L_0(x) &= 1 \\
L_1(x) &= 1 - x \\
L_{n+1}(x) &= \frac{1}{n+1} \, (2n + 1 - x) \, L_n(x) - \frac{n}{n+1} \, L_{n-1}(x),
\end{aligned}
$$

- Hermite polynomials
  - $[a, b] = (-\infty, \infty)$
  - $w(x) = e^{-x^2}$
  - $H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} \left(e^{-x^2}\right)$
  - Recurrence formula:

$$
\begin{aligned}
H_0(x) &= 1 \\
H_1(x) &= 2x \\
H_{n+1}(x) &= 2x\, H_n(x) - 2n\ H_{n-1}(x).
\end{aligned}
$$

- General Orthogonal Polynomials
  - Few problems have the specific intervals and weights used in definitions
  - One must adapt interval through linear COV
    * If compact interval $[a,b]$ is mapped to $[-1,1]$ by
    $$y=-1+2\frac{x-a}{b-a}$$
    weight is $w(y)$, and $\phi_i(y)$ are orthogonal w.r.t. $w(y)$ over $y\in[-1,1]$, then $\phi_i\left(-1+2\frac{x-a}{b-a}\right)$ are orthogonal over $x\in[a,b]$ with respect to $w\left(-1+2\frac{x-a}{b-a}\right)$ iff
    * If half-infinite interval $[a,\infty]$ is mapped to $[0,\infty]$,
    $$y=\frac{x-a}{\lambda}$$
    $$w(y)=e^{-y}$$
    then $\phi_i\left(\frac{x-a}{\lambda}\right)$ are orthogonal over $x\in[a,\infty]$ w.r.t. $w\left(\frac{x-a}{\lambda}\right)$ iff $\phi_i(y)$ are orthogonal over $y\in[0,\infty]$ w.r.t. $w(y)$
    * If $[-\infty,\infty]$ is mapped to $[-\infty,\infty]$ by
    $$y=(x-\mu)\Big/\sqrt{\lambda}$$
    $$w(y)=e^{-y^2}$$
    then $\phi_i\left(\frac{x-\mu}{\sqrt{\lambda}}\right)$ are orthogonal over $x\in[a,\infty]$ w.r.t. $w\left(\frac{x-\mu}{\sqrt{\lambda}}\right)$ iff $\phi_i(y)$ are orthogonal over $y\in[0,\infty]$ w.r.t. $w(y)$

- Trigonometric polynomials and Fourier series
  - $\{\cos(n\theta), \sin(m\theta)\}$ are orthogonal on $[-\pi, \pi]$.
  - If f is continuous on $[-\pi, \pi]$ and $f(-\pi) = f(\pi)$, then

$$f(\theta) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n \cos(n\theta) + \sum_{n=1}^{\infty} b_n \sin(n\theta)$$

    where the *Fourier coefficients* are

$$\begin{aligned} a_n &= \frac{1}{\pi} \int_{\pi}^{\pi} f(\theta) \cos(n\theta) d\theta \\ b_n &= \frac{1}{\pi} \int_{\pi}^{\pi} f(\theta) \sin(n\theta)\, d\theta, \end{aligned}$$

  - A *trigonometric polynomial* is any function of the form in (6.4.4).
  - Convergence is uniform FOR PERIODIC FUNCTIONS.
  - Excellent for approximating a smooth *periodic function*, i.e., $f : R \to R$ such that for some ω, $f(x) = f(x + \omega)$.
  - Not good for nonperiodic functions
    * Convergence is not uniform
    * Many terms are needed
    * Chebyshev polynomial theory comes from a nonlinear COV

Regression

- Data: $(x_i, y_i), i = 1, .., n$.
- Objective: Find a function $f(x; \beta)$ with $\beta \in R^m$, $m \leq n$, with $y_i \doteq f(x_i), i = 1, .., n$.
- Least Squares regression:

$$\min_{\beta \in R^m} \sum (y_i - f(x_i; \beta))^2$$

Chebyshev Regression

- Approximation function: degree m polynomial
- Data: n data points, $(x_i, y_i), i = 1, ..,$ x_i are the n zeroes of $T_n(x)$ adapted to $[a, b]$
- More data than unknown coefficients: $n > m + 1$
- Objective: minimize unweighted sum of errors at nodes

Chebyshev Interpolation

- Special case of regression with $n = m + 1$.

Algorithm 6.4: Chebyshev Approximation Algorithm in $\mathbb{R}^1$

- Objective: Given $f(x)$ defined on $[a, b]$, find a m-point degree n Chebyshev polynomial approximation $p(x)$

- Step 1: Compute the $m \geq n+1$ Chebyshev interpolation nodes on $[-1, 1]$:
$$z_k = -\cos\left(\frac{2k-1}{2m} \pi\right), \quad k = 1, \cdots, m.$$

- Step 2: Adjust nodes to $[a, b]$ interval:
$$x_k = (z_k + 1)\left(\frac{b-a}{2}\right) + a, k = 1, ..., m.$$

- Step 3: Evaluate f at approximation nodes:
$$w_k = f(x_k), \quad k = 1, \cdots, m.$$

- Step 4: Compute Chebyshev coefficients, $a_i, i = 0, \cdots, n$:
$$a_i = \frac{\sum_{k=1}^{m} w_k T_i(z_k)}{\sum_{k=1}^{m} T_i(z_k)^2}$$

  to arrive at approximation of $f(x, y)$ on $[a, b]$:
$$p(x) = \sum_{i=0}^{n} a_i T_i\left(2\frac{x-a}{b-a} - 1\right)$$

Minmax Approximation

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

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

- Problem: Difficult to compute
- Chebyshev minmax property

Theorem 1 *Suppose $f : [-1, 1] \to R$ is C^k for some $k \geq 1$, and let I_n be the n-point (degree $n-1$) polynomial interpolation of f based at the zeroes of $T_n(x)$. Then*

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

- Decompose the error bound
  - $\frac{2}{\pi}\log(n+1)$: grows very slowly in n; ignore it
  - $\left(\frac{\pi}{2}\right)^k \left(\frac{b-a}{2}\right)^k$: independent of n and f
  - $\| f^{(k)} \|_\infty$: a measure of k'th order curvature
  - $\frac{(n-k)!}{n!}$: essentially n^{-k}
- Chebyshev interpolation:
  - Good properties
    * converges in L^∞
    * essentially achieves minmax approximation
    * easy to compute
  - Caution
    * does *not* necessarily approximate f'
    * if $\| f^{(k)} \|_\infty$ is large then the error may be large for moderate n.

Multidimensional approximation methods

- Lagrange Interpolation
  - Data: $D \equiv \{(x_i, z_i)\}_{i=1}^{N} \subset R^{n+m}$, where $x_i \in R^n$ and $z_i \in R^m$
  - Objective: find $f : R^n \to R^m$ such that $z_i = f(x_i)$.

- Counterexample:
  - Interpolation nodes:

$$\{P_1, P_2, P_3, P_4\} \equiv \{(1,0), (-1,0), (0,1), (0,-1)\}$$

  - Use linear combinations of $\{1, x, y, xy\}$.
  - Data: $z_i = f(P_i), i = 1, 2, 3, 4$.
  - Interpolation form $f(x, y) = a + bx + cy + dxy$
  - Defining conditions form the singular system

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{pmatrix},$$

Task: Find combinations of interpolation nodes and spanning functions to produce a nonsingular (well-conditioned) interpolation matrix.

Tensor products

- General Approach:
  - If A and B are sets of functions over $x \in R^n$, $y \in R^m$, their tensor product is

$$A \otimes B = \{\varphi(x)\psi(y) \mid \varphi \in A, \psi \in B\}.$$

  - Given a basis for functions of x_i, $\Phi^i = \{\varphi^i_k(x_i)\}_{k=0}^{\infty}$, the *n-fold tensor product* basis for functions of $(x_1, x_2, \ldots, x_n)$ is

$$\Phi = \left\{ \prod_{i=1}^{n} \varphi^i_{k_i}(x_i) \mid k_i = 0, 1, \cdots, \ i = 1, \ldots, n \right\}$$

- Orthogonal polynomials and Least-square approximation

  – Suppose Φ^i are orthogonal with respect to $w_i(x_i)$ over $[a_i, b_i]$

  Least squares approximation of $f(x_1, \cdots, x_n)$ in Φ is

$$\sum_{\varphi \in \Phi} \frac{\langle \varphi, f \rangle}{\langle \varphi, \varphi \rangle} \varphi,$$

  where the product weighting function

$$W(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} w_i(x_i)$$

  defines $\langle \cdot, \cdot \rangle$ over $D = \prod_i [a_i, b_i]$ in

$$\langle f(x), g(x) \rangle = \int_D f(x) g(x) W(x) dx.$$

Algorithm 6.4: Chebyshev Approximation Algorithm in $\mathbb{R}^2$

- Objective: Given $f(x,y)$ defined on $[a,b]\times[c,d]$, find the m-point degree n Chebyshev polynomial approximation $p(x,y)$

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

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

- Step 2: Adjust nodes to $[a,b]$ and $[c,d]$ intervals:

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

$$y_k = (z_k+1)\left(\frac{d-c}{2}\right) + c, k = 1, ..., m.$$

- Step 3: Evaluate f at approximation nodes:

$$w_{k,\ell} = f(x_k, y_\ell)\ ,\ \ k = 1, \cdots, m.\ ,\ \ \ell = 1, \cdots, m.$$

- Step 4: Compute Chebyshev coefficients, $a_{ij}, i, j = 0, \cdots, n$:

$$a_{ij} = \frac{\sum_{k=1}^{m}\sum_{\ell=1}^{m} w_{k,\ell}T_i(z_k)T_j(z_\ell)}{\left(\sum_{k=1}^{m} T_i(z_k)^2\right)\left(\sum_{\ell=1}^{m} T_j(z_\ell)^2\right)}$$

  to arrive at approximation of $f(x,y)$ on $[a,b]\times[c,d]$:

$$p(x,y) = \sum_{i=0}^{n}\sum_{j=0}^{n} a_{ij}T_i\left(2\frac{x-a}{b-a}-1\right)T_j\left(2\frac{y-c}{d-c}-1\right)$$

Complete polynomials

- Taylor's theorem for $\mathbb{R}^n$ produces the approximation

$$
\begin{aligned}
f(x) \doteq\; & f(x^0) \\
& + \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x^0)\,(x_i - x_i^0) \\
& + \frac{1}{2} \sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \frac{\partial^2 f}{\partial x_{i_1} \partial x_{i_k}}(x_0)(x_{i_1} - x_{i_1}^0)(x_{i_k} - x_{i_k}^0) \\
& \vdots
\end{aligned}
$$

  – For $k = 1$, Taylor's theorem for n dimensions used the linear functions

$$\mathcal{P}_1^n \equiv \{1, x_1, x_2, \cdots, x_n\}$$

For $k = 2$, Taylor's theorem uses

$$\mathcal{P}_2^n \equiv \mathcal{P}_1^n \cup \{x_1^2, \cdots, x_n^2, x_1 x_2, x_1 x_3, \cdots, x_{n-1} x_n\}.$$

$\mathcal{P}_2^n$ contains some product terms, but not all; for example, $x_1 x_2 x_3$ is not in $\mathcal{P}_2^n$.

- In general, the kth degree expansion uses the *complete set of polynomials of total degree k in n variables.*

$$\mathcal{P}_k^n \equiv \{x_1^{i_1} \cdots x_n^{i_n} \mid \sum_{\ell=1}^{n} i_\ell \leq k,\ 0 \leq i_1, \cdots, i_n\}$$

- Complete orthogonal basis includes only terms with total degree k or less.

- Sizes of alternative bases

degree k	$\mathcal{P}_k^n$	Tensor Prod.
2	$1 + n + n(n+1)/2$	3^n
3	$1 + n + \frac{n(n+1)}{2} + n^2 + \frac{n(n-1)(n-2)}{6}$	4^n

  - Complete polynomial bases contains fewer elements than tensor products.
  - Asymptotically, complete polynomial bases are as good as tensor products.
  - For smooth n-dimensional functions, complete polynomials are more efficient approximations

- Construction

  - Compute tensor product approximation, as in Algorithm 6.4
  - Drop terms not in complete polynomial basis (or, just compute coefficients for polynomials in complete basis).
  - Complete polynomial version is faster to compute since it involves fewer terms

Approximation Methods: Summary

- Interpolation versus regression
  - Lagrange data uses level information only
  - Hermite data also uses slope information
  - Regression uses more points than coefficients
- One-dimensional problems
  - Smooth approximations
    - Orthogonal polynomial methods for nonperiodic functions
    - Fourier approximations for periodic functions
  - Less smooth approximations
    - Splines
    - Shape-preserving splines

- Multidimensional data
  - Tensor product methods have curse of dimension
  - Complete polynomials are more efficient
  - Neural networks are most efficient
- Approximation versus Statistics
  - Similarities:
    * both approximate unknown functions
    * both use finite amount of data
  - Differences
    * approximation uses error-free data, not noisy data
    * approximation generates data, not constrained by observations