Lecture 3
Global Approximation Methods

Peifan Wu

UBC

Global Function Approximation

- Global Function: function interpolation over the entire domain
- Types of interpolations:
  - Spectral methods – orthogonal polynomials (e.g. Chebyshev) that use global basis
  - Finite element methods – splines (e.g. B-splines, cubic splines, Schumaker splines) that use local basis
- To approximate a known function $f(x)$ by a linear combination of basis functions ϕ

$$f(x) \equiv \sum_{j=0}^{n} w_j \phi_j(x)$$

  - Basis functions: linearly independent functions that span the family of functions chosen for the interpolation
  - In general we are interested in the space of $\mathcal{C}^0$ and $\mathcal{C}^1$ functions

Interpolation

- The problem boils down to determining the weights w_i
- Several choices:
  - Collocation: use the same number of interpolant nodes as the number of basis functions. w comes from the solution to $\Phi w = y$
  - If we have more interpolation nodes than basis functions, then we have a ”curve fitting” problem. One way is to minimize SSR by applying Least Square,

    $$w = \left(\Phi'\Phi\right)^{-1}\Phi' y$$

  - Or apply Galerkin method: define residuals

    $$r(x) = f(x) - \sum_{j=0}^{n} w_j \phi_j(x)$$

    and solve the equations

    $$\int_a^b r(x)\,\phi_j(x)\,\mathrm{d}x = 0, j = 0, 1, \ldots, n$$

- We apply collocation method for the following exercises

Spectral Methods

- We use polynomial basis
- Why polynomials? Weierstrass Theorem: if $\mathcal{C}[a,b]$ is the set of all continuous function on $[a,b]$ then for all $f \in \mathcal{C}[a,b]$ and $\epsilon > 0$ there exists a polynomial q for which

$$\sup_{x \in [a,b]} |f(x) - q(x)| < \epsilon$$

- Polynomials can approximate any continuous function over a compact domain arbitrarily well

Monomial Basis

- The most naive procedure: use monomials as basis functions

$$\phi_j(x) = x^j, j = 0, 1, \ldots, n$$

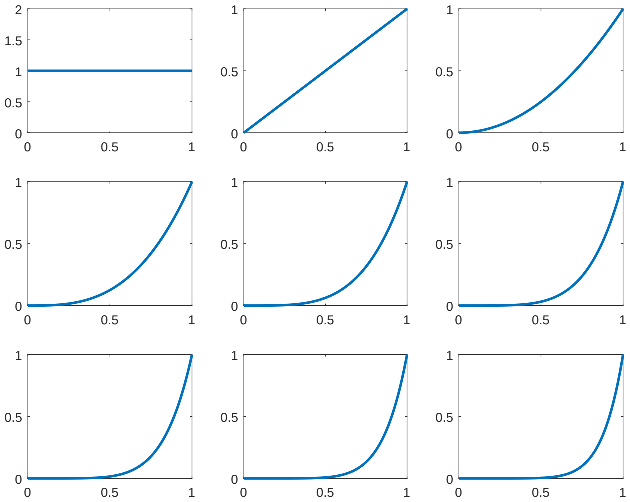
  with interpolating polynomials of the form

$$\bar{f}(x) \equiv p_n(x) = w_0 + w_1 x + w_2 x^2 + \ldots + w_n x^n$$

- Use a linear system of $n+1$ equations to solve w_i

$$\begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

Monomials in $[0, 1]$



Chebyshev Polynomials

- Chebyshev polynomials are trigonometric and can be constructed recursively,

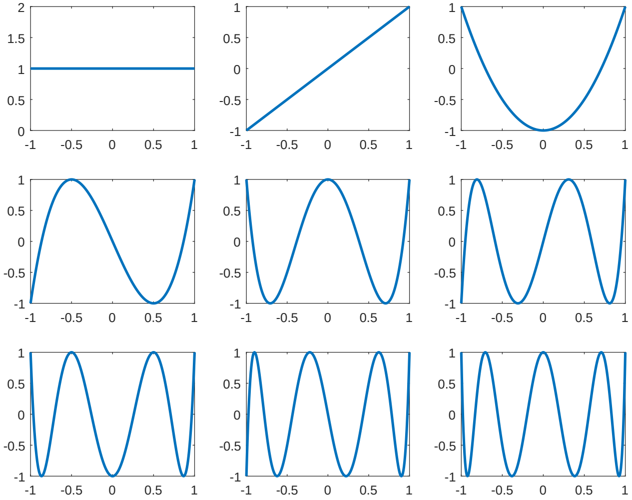
$$
\begin{aligned}
T_0(x) &= 1 \\
T_1(x) &= x \\
T_{k+1}(x) &= 2xT_k(x) - T_{k-1}(x)
\end{aligned}
$$

- $T_n(x)$ has n distinct roots in $[-1, 1]$ with expression

$$x_k = \cos\left(\frac{2k-1}{2n}\pi\right), k = 1, \ldots, n$$

- The interpolation nodes that minimize the error of Chebyshev interpolation are the zeros of the Chebyshev polynomials

Chebyshev Basis Functions in $[0, 1]$



Interpolation using Chebyshev Polynomials

- Choose x_i to be the roots of T_n

- The vector of w_i can be obtained as

$$\begin{bmatrix} T_0(x_0) & T_1(x_0) & \cdots & T_n(x_0) \\ T_0(x_1) & T_1(x_1) & \cdots & T_n(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ T_0(x_n) & T_1(x_n) & \cdots & T_n(x_n) \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}$$

- Finally the polynomial approximation of f is

$$\bar{f}(x) = \sum_{j=0}^{n} w_j T_j(x)$$

Linear B-splines

- B^1 splines implement piecewise linear interpolation

$$B_k^1(x) = \begin{cases} \frac{x - x_{k-1}}{x_k - x_{k-1}} & \text{if } x_{k-1} \leqslant x < x_k \\ \frac{x_{k+1} - x}{x_{k+1} - x_k} & \text{if } x_k \leqslant x < x_{k+1} \\ 0 & \text{elsewhere} \end{cases}$$

  it looks like ”tent-functions” with peak at x_k equal to 1

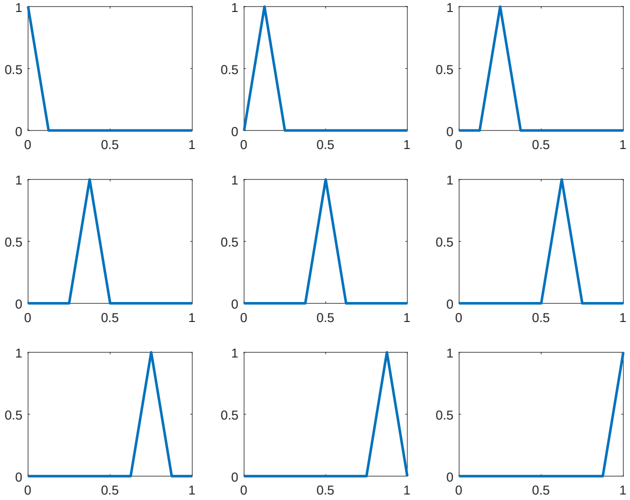
- Then the interplant is

$$\bar{f}(x) = \sum_{k=0}^{n} f(x_k) B_k^1(x)$$

- Essentially, for $x_k < x < x_{k+1}$,

$$\bar{f}(x) = f(x_k) + [f(x_{k+1}) - f(x_k)] \frac{x - x_k}{x_{k+1} - x_k}$$

Linear B-splines on $[0,1]$



Pros and Cons of Linear Splines

- Pros:
  1. Preserves monotonicity and concavity of f
  2. We can exploit information about f clustering more closely together in areas of high curvature in order to increase the accuracy
  3. Captures binding inequality constraints very well

- Cons:
  1. The approximated function is not differentiable at the knots
  2. The second derivative is 0, and the function is not smooth

Cubic Splines on $[0, 1]$

TODO

Pros and Cons of Cubic Splines

- Pros:
  1. Easy to compute: interpolant matrix very sparse, easy to invert
  2. Smooth approximation

- Cons:
  1. It may not be able to handle constraints well
  2. It does not preserve monotonicity and concavity of f

Example: Income Fluctuation Problem

Income Fluctuation Problem

$$
\begin{aligned}
V(a,y) &= \max_{c,a'} u(c) + \beta \sum_{y' \in Y} \pi(y'|y)\, V(a',y') \\
c + a' &\leqslant Ra + y \\
a' &\geqslant \underline{a}
\end{aligned}
$$

Euler equation reads

$$
u_c\left(Ra + y - a'\right) - \beta R \sum_{y' \in Y} \pi(y'|y)\, u_c\left(Ra' + y' - a''\right) \geqslant 0
$$

where the strict inequality holds when the constraint is binding

Policy Function Approximation

1. Choose a set of basis functions T_k, $k = 1, \ldots, n$ and express the policy function as

$$a' = g\left(a, \phi_y\right) = \sum_{k=0}^{n} \phi_{y,k} T_k\left(a\right)$$

2. Define the residual function from the Euler equation

$$\begin{aligned}\mathcal{R}\left(a, \phi_y\right) \equiv\ & u_c\left(Ra + y - \sum_{k=0}^{n} \phi_{y,k} T_k\left(a\right)\right) \\ & - \beta R \sum_{y' \in Y} \pi\left(y' | y\right) u_c\left(R \sum_{k=0}^{n} \phi_{y,k} T_k\left(a\right) + y' - \sum_{k=0}^{n} \phi_{y',k} T_k\left(\sum_{k=0}^{n} \phi_{y,k} T_k\left(a\right)\right)\right)\end{aligned}$$

3. Using a multidimensional root-finding algorithm to find solutions to

$$\mathcal{R}_i\left(a, \phi_y\right) = 0, i = 1, \ldots, n, y \in Y$$

Check the Constraint

- Tell the nonlinear solver that the policy function should be $a' \geqslant \underline{a}$
- In matlab the function fmincon allows you to do that