Lecture 10 Polynomial interpolation

Weinan E[1,2] and Tiejun Li[2]

[1]Department of Mathematics,
Princeton University,
weinan@princeton.edu

[2]School of Mathematical Sciences,
Peking University,
tieli@pku.edu.cn
No.1 Science Building, 1575

Outline

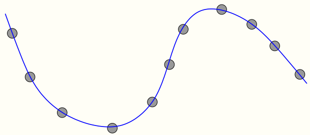
Basic motivations

- Plotting a smooth curve through discrete data points

  Suppose we have a sequence of data points

Coordinates	x_1	x_2	$\cdots$	x_n
Function	y_1	y_2	$\cdots$	y_n

- Try to plot a smooth curve (a continuous differentiable function) connecting these discrete points.



Basic motivations

- Representing a complicate function by a simple one
  Suppose we have a complicate function
  $$y = f(x),$$
  we want to compute function values, derivatives, integrations,. . . very quickly and easily.
- One strategy
  1. Compute some discrete points from the complicate form;
  2. Interpolate the discrete points by a polynomial function or piecewise polynomial function;
  3. Compute the function values, derivatives or integrations via the simple form.

Polynomial interpolation

Polynomial interpolation is one the most fundamental problems in numerical methods.

Outline

Method of undetermined coefficients

- Suppose we have $n+1$ discrete points

$$(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$$

- We need a polynomial of degree n to do interpolation ($n+1$ equations and $n+1$ undetermined coefficients $a_0, a_1, \ldots, a_n$

$$p_n(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0$$

- Equations

$$\begin{cases} p_n(x_0) = y_0 \\ p_n(x_1) = y_1 \\ \cdots \\ p_n(x_n) = y_n \end{cases}$$

Method of undetermined coefficients

- The coefficient matrix

$$V_n = \begin{vmatrix} x_0^n & x_0^{n-1} & \cdots & x_0 \\ x_1^n & x_1^{n-1} & \cdots & x_1 \\ \cdots & \cdots & \cdots & \cdots \\ x_n^n & x_n^{n-1} & \cdots & x_n \end{vmatrix}$$

  is a Vandermonde determinant, nonsingular if $x_i \neq x_j \ (i \neq j)$.

- Though this method can give the interpolation polynomial theoretically, the condition number of the Vandermonde matrix is very bad!
- For example, if

$$x_0 = 0,\ x_1 = \frac{1}{n},\ x_2 = \frac{2}{n}, \cdots, x_n = 1$$

  then $V_n \leq \frac{1}{n^n}$!

Lagrange interpolating polynomial

- Consider the interpolation problem for 2 points (linear interpolation), one type is the point-slope form

$$p(x) = \frac{y_1 - y_0}{x_1 - x_0}x + \frac{y_0 x_1 - y_1 x_0}{x_1 - x_0}$$

- Another type is as

$$p(x) = y_0 l_0(x) + y_1 l_1(x)$$

where

$$l_0(x) = \frac{x - x_1}{x_0 - x_1}, \quad l_1(x) = \frac{x - x_0}{x_1 - x_0}$$

satisfies

$$l_0(x_0) = 1, l_0(x_1) = 0; \quad l_1(x_0) = 0, l_1(x_1) = 1$$

- $l_0(x), l_1(x)$ are called basis functions. They are another base for space spanned by functions $1,\ x$.

Lagrange interpolating polynomial

- Define the basis function

$$l_i(x) = \frac{(x - x_0)(x - x_1)\cdots(x - x_{i-1})(x - x_{i+1})\cdots(x - x_n)}{(x_i - x_0)(x_i - x_1)\cdots(x_i - x_{i-1})(x_i - x_{i+1})\cdots(x_i - x_n)}$$

then we have

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

- The functions $l_i(x)$ $(i = 0, 1, \ldots, n)$ form a new basis in $\mathbb{P}_n$ instead of $1, x, x^2, \ldots, x^n$.

Lagrange interpolating polynomial

- General form of the Lagrange polynomial interpolation

$$L_n(x) = y_0 l_0(x) + y_1 l_1(x) + \cdots + y_n l_n(x)$$

then $L_n(x)$ satisfies the interpolation condition.

- The shortcoming of Lagrange interpolation polynomial: If we add a new interpolation point into the sequence, all the basis functions will be useless!

Newton interpolation

- Define the 0-th order divided difference

$$f[x_i] = f(x_i)$$

- Define the 1-th order divided difference

$$f[x_i, x_j] = \frac{f[x_i] - f[x_j]}{x_i - x_j}$$

- Define the k-th order divided difference by $k-1$-th order divided difference recursively

$$f[x_{i_0}, x_{i_1}, \dots, x_{i_k}] = \frac{f[x_{i_0}, x_{i_1}, \dots, x_{i_{k-1}}] - f[x_{i_1}, x_{i_2}, \dots, x_{i_k}]}{x_{i_0} - x_{i_k}}$$

Newton interpolation

- Recursively we have the following divided difference table

Coordinates	0-th order	1-th order	2-th order
x_0	$f[x_0]$		
x_1	$f[x_1]$	$f[x_0, x_1]$	
x_2	$f[x_2]$	$f[x_1, x_2]$	$f[x_0, x_1, x_2]$
x_3	$f[x_3]$	$f[x_2, x_3]$	$f[x_1, x_2, x_3]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Newton interpolation

- Divided difference table: an example

Discrete data points

x	0.00	0.20	0.30	0.50
$f(x)$	0.00000	0.20134	0.30452	0.52110

Divided difference table

i	x_i	$f[x_i]$	$f[x_{i-1}, x_i]$	$f[x_{i-2}, x_{i-1}, x_i]$	$f[x_0, x_1, x_2, x_3]$
0	0.00	0.00000			
1	0.20	0.20134	1.0067		
2	0.30	0.30452	1.0318	0.08367	
3	0.50	0.52110	1.0829	0.17033	0.17332

Newton interpolation

- The properties of divided difference
  1. $f[x_0, x_1, \ldots, x_k]$ is the linear combination of $f(x_0), f(x_1), \ldots, f(x_n)$.
  2. The value of $f[x_0, x_1, \ldots, x_k]$ does NOT depend on the order the coordinates $x_0, x_1, \ldots, x_k$.
  3. If $f[x, x_0, \ldots, x_k]$ is a polynomial of degree m, then $f[x, x_0, \ldots, x_k, x_{k+1}]$ is of degree $m - 1$.
  4. If $f(x)$ is a polynomial of degree n, then

$$f[x, x_0, \ldots, x_n] = 0$$

Newton interpolation

- From the definition of divided difference, we have for any function $f(x)$

$$f(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1)$$

$$+ \cdots + f[x_0, x_1, \ldots, x_n](x - x_0)(x - x_1) \cdots (x - x_{n-1})$$

$$+ f[x, x_0, x_1, \ldots, x_n](x - x_0)(x - x_1) \cdots (x - x_n)$$

- Take $f(x)$ as the Lagrange interpolation polynomial $L_n(x)$, because

$$L_n[x, x_0, x_1, \ldots, x_n] = 0$$

we have

$$L_n(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) +$$

$$+ \cdots + f[x_0, x_1, \ldots, x_n](x - x_0)(x - x_1) \cdots (x - x_{n-1})$$

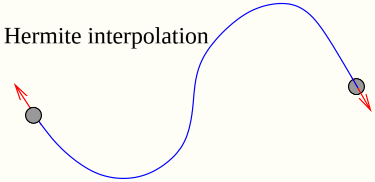
This formula is called Newton interpolation formula.

Hermite interpolation

- Hermite interpolation is the interpolation specified derivatives.
- Formulation: find a polynomial $p(x)$ such that

$$p(x_0) = f(x_0), p'(x_0) = f'(x_0), p(x_1) = f(x_1), p'(x_1) = f'(x_1)$$

- Sketch of Hermite interpolation



Hermite interpolation

- We need a cubic polynomial to fit the four degrees of freedom, one choice is

$$p(x) = a + b(x - x_0) + c(x - x_0)^2 + d(x - x_0)^2(x - x_1)$$

- We have

$$p'(x) = b + 2c(x - x_0) + 2d(x - x_0)(x - x_1) + d(x - x_0)^2$$

- then we have

$$f(x_0) = a, \quad f'(x_0) = b$$

$$f(x_1) = a + bh + ch^2, \quad f'(x_1) = b + 2ch + dh^2 \quad (h = x_1 - x_0)$$

- a, b, c, d could be solved.

Error estimates

Theorem

Suppose $a = x_0 < x_1 < \cdots < x_n = b$, $f(x) \in C^{n+1}[a, b]$, $L_n(x)$ *is the Lagrange interpolation polynomial, then*

$$E(f; x) = |f(x) - L_n(x)| \leq \frac{\omega_n(x)}{(n+1)!} M_{n+1}$$

where

$$\omega_n(x) = (x - x_0)(x - x_1) \cdots (x - x_n), \quad M_{n+1} = \max_{x \in [a,b]} |f^{(n+1)}(x)|.$$

Remark: This theorem doesn't imply the uniform convergence when $n \to \infty$.

Runge phenomenon

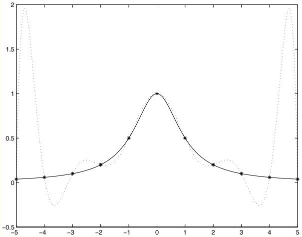
- Suppose

$$f(x) = \frac{1}{1 + 25x^2}$$

take the equi-partitioned nodes

$$x_i = -1 + \frac{2i}{n}, \ i = 0, 1, \dots, n$$

- Lagrange interpolation $(n = 10)$



Remark on polynomial interpolation

- Runge phenomenon tells us Lagrange interpolation could NOT guarantee the uniform convergence when $n \to \infty$.
- Another note: high order polynomial interpolation is unstable!
- This drives us to investigate the piecewise interpolation.

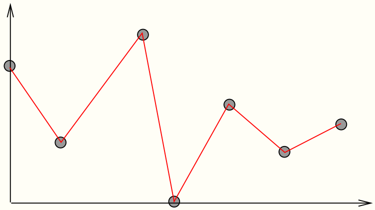
Outline

Piecewise linear interpolation

- Suppose we have $n+1$ discrete points

$$(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$$

- Piecewise linear interpolation is to connect the discrete data points as



Tent basis functions

- Define the piecewise linear basis functions as

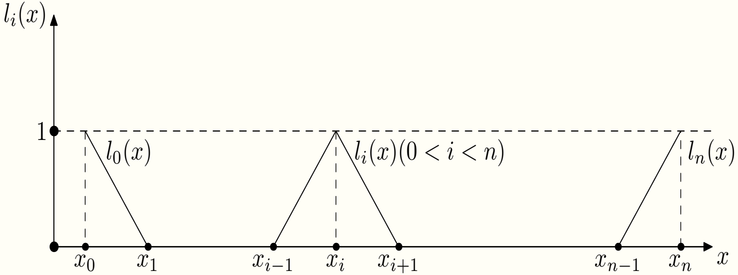
$$l_{n,0}(x) = \begin{cases} \dfrac{x - x_1}{x_0 - x_1}, & x \in [x_0, x_1], \\ 0, & x \in [x_1, x_n], \end{cases}$$

$$l_{n,i}(x) = \begin{cases} \dfrac{x - x_{i-1}}{x_i - x_{i-1}}, & x \in [x_{i-1}, x_i], \\ \dfrac{x - x_{i+1}}{x_i - x_{i+1}}, & x \in [x_i, x_{i+1}], \quad i = 1, 2, \ldots, n-1, \\ 0, & x \notin [x_{i-1}, x_{i+1}], \end{cases}$$

$$l_{n,n}(x) = \begin{cases} \dfrac{x - x_{n-1}}{x_n - x_{n-1}}, & x \in [x_{n-1}, x_n], \\ 0, & x \in [x_0, x_{n-1}]. \end{cases}$$

Tent basis functions

- The sketch of tent basis function



Piecewise linear interpolation function

- With the above tent basis function $l_{n,i}(x)$, we have

$$l_{n,i}(x_j) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

- The functions $l_{n,i}(x)$ form a basis in piecewise linear function space with nodes x_i $(i = 0, 1, \ldots, n)$.
- Piecewise linear interpolation function

$$p(x) = y_0 l_{n,0}(x) + y_1 l_{n,1}(x) + \cdots + y_n l_{n,n}(x)$$

then $p(x)$ satisfies the interpolation condition.

Cubic spline

- In order to make the interpolation curve more smooth, cubic spline is introduced.
- Formulation: Given discrete points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, find function $S_h(x)$ such that

  (1) $S_h(x)$ is a cubic polynomial in each interval $[x_i, x_{i+1}]$;

  (2) $S_h(x_i) = y_i,\ i = 0, 1, \ldots, n$;

  (3) $S_h(x) \in C^2[a, b]$.

Cubic spline

- Suppose we have n cubic polynomials in each interval, we have $4n$ unknowns totally. The interpolation condition gives $2n$ equations, $S_h(x) \in C^1$ gives $n-1$ equations, $S_h(x) \in C^2$ gives $n-1$ equations, so we have $4n-2$ equations totally, we need some boundary conditions.
- Supplementary boundary conditions:
  (1) Fixed boundary: $S'_h(x_0) = f'(x_0)$,$S'_h(x_n) = f'(x_n)$;
  (2) Natural boundary: $S''_h(x_0) = 0$,$S''_h(x_n) = 0$;
  (3) Periodic boundary:

$$S_h(x_0) = S_h(x_n),\ S'_h(x_0) = S'_h(x_n),\ S''_h(x_0) = S''_h(x_n).$$

- Each type of boundary condition gives 2 equations, thus we have $4n$ equations and $4n$ unknowns. The system could be solved theoretically.
- Problem: Why are piecewise cubic polynomials needed?)

Homework assignment

- Take interpolation points
$$x_k = -1 + \frac{2k}{n},\ k = 0, 1, \ldots, n$$
for Runge function, plot the Lagrange polynomial of degree n $(n = 1, 2, \ldots, 15)$.
- Take interpolation points
$$x_k = \cos\frac{k\pi}{n},\ k = 0, 1, \ldots, n$$
for Runge function, plot the Lagrange polynomial of degree n $(n = 1, 2, \ldots, 15)$.