Lec 25: Piecewise Interpolation – Piecewise Linear

Interpolation Problem

Problem Statement

General Interpolation Problem

Given a set of n data points $\{(x_j, y_j) \mid j \in \mathbb{N}[1, n]\}$ with $x_1 < x_2 < \ldots < x_n$, find a function $p(x)$, called the **interpolant**, such that

$$p(x_j) = y_j, \quad \text{for } j = 1, 2, \ldots, n\,.$$

The ordered pair (x_j, y_j) is called the **data point**.

- x_j is called the **abscissa** or the **node**.
- y_j is called the **ordinate**.

Polynomials

One approach is to find an interpolating *polynomial* of degree (at most) $n-1$,

$$p(x) = c_1 + c_2 x + c_3 x^2 + \cdots + c_n x^{n-1}.$$

- The unknown coefficients $c_1, \ldots, c_n$ are determined by solving the square linear system $V\mathbf{c} = \mathbf{y}$ where

$$V = \begin{bmatrix} 1 & x_1 & \cdots & x_1^{n-2} & x_1^{n-1} \\ 1 & x_2 & \cdots & x_2^{n-2} & x_2^{n-1} \\ \vdots & \vdots & & \vdots & \vdots \\ 1 & x_n & \cdots & x_n^{n-2} & x_n^{n-1} \end{bmatrix}, \quad \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

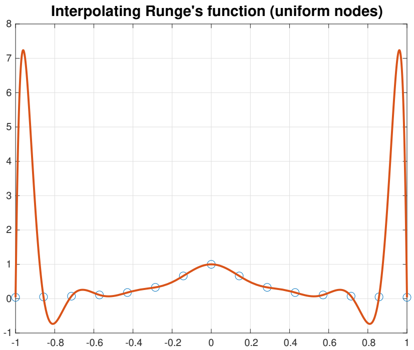
  the matrix V is called the **Vandermonde matrix**; see Lecture 13.

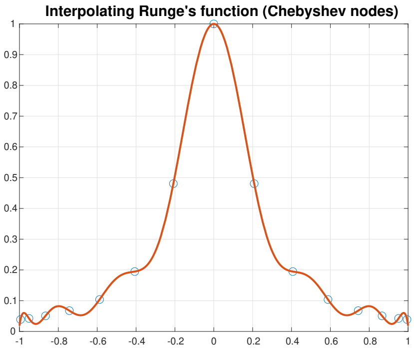
- A polynomial interpolant has severe oscillations as n grows large, unless nodes are special; see illustration in the next slide.

Illustration Runge's Phenomenon

Polynomial Interpolation of 15 data points collected from the same function

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





Piecewise Polynomials

To handle real-life data sets with large n and unrestricted node distribution:

- An alternate approach is to use a low-degree polynomial between each pair of data points; it is called the **piecewise polynomial interpolation**.
- The simplest case is **piecewise linear interpolation** (degree 1) in which the interpolant is linear between each pair of consecutive nodes.
- The most commonly used method is **cubic spline interplation** (degree 3).
- The endpoints of the low-degree polynomials are called **breakpoints** or **knots**.
- The breakpoints and the data points almost always coincide.

MATLAB Function: INTERP1

In MATLAB, piecewise polynomials are constructed using `interp1` function. Suppose the x and y data are stored in vectors `xdp` and `ydp`. To evaluate the piecewise interpolant at `x` (an array):

- By default, it finds a piecewise linear interpolant.

```
y = interp1(xdp, ydp, x);
```

- To obtain a smoother interpolant that is piecewise cubic, use `'spline'` option.

```
y = interp1(xdp, ydp, x, 'spline');
```

Demonstration: Piecewise Polynomial Interplation

To interpolate data obtained from [1]

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

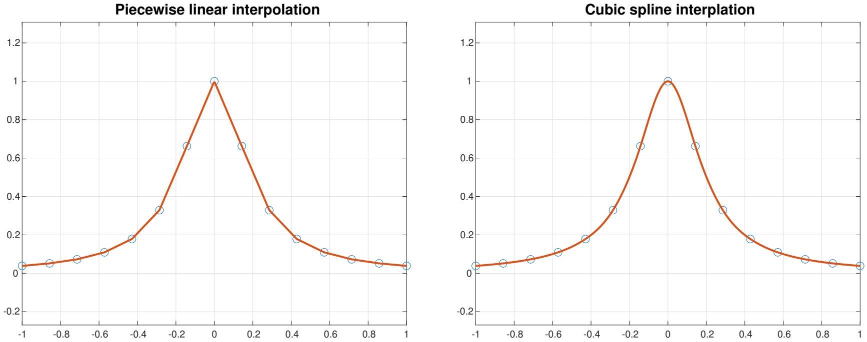
```
% Generate data and eval pts
n = 15;
xdp = linspace(-1,1,n)';
ydp = 1./(1+25*xdp.^2);
x = linspace(-1,1,400)';

% PL
plot(xdp,ydp,'o'), hold on
plot(x, interp1(xdp,ydp,x))

% Cubic spline
plot(xdp,ydp,'o'), hold on
plot(x, interp1(xdp,ydp,x,'spline'));
```

[1] This function is often called the Runge's function.

Demonstration: Piecewise Polynomial Interplation (cont')



Conditioning

Set-up for analysis.

- Let (x_j, y_j) for $j = 1, \ldots, n$ denote the data points. Assume that the nodes x_j are fixed and let $a = x_1$, $b = x_n$.
- View the interpolation problem as a mathematical function $\mathcal{I}$ with
  - Input: a vector $\mathbf{y}$ (ordinates, or y-data ponits)
  - Output: a function $p(x)$ such that $p(x_j) = y_j$ for all j.

  (That is, $\mathcal{I}$ is a *black box* that produces the interpolant from a data vector.)
- For the interpolation methods under consideration (polynomial or piecewise polynomial), $\mathcal{I}$ is *linear*:

$$\mathcal{I}(\alpha\mathbf{y} + \beta\mathbf{z}) = \alpha\mathcal{I}(\mathbf{y}) + \beta\mathcal{I}(\mathbf{z}),$$

  for all vectors $\mathbf{y}, \mathbf{z}$ and scalars α, β.

Conditioning: Main Theorem

Theorem 1 (Conditioning of General Interpolation)

Suppose that $\mathcal{I}$ is a linear interpolation method. Then the absolute condition number of $\mathcal{I}$ satisfies

$$\max_{1 \leqslant j \leqslant n} \left\| \mathcal{I}(\mathbf{e}_j) \right\|_\infty \leqslant \kappa(\mathbf{y}) \leqslant \sum_{j=1}^{n} \left\| \mathcal{I}(\mathbf{e}_j) \right\|_\infty ,$$

where all vectors and functions are measured in the infinity norm.

Conditioning: Notes

- The functional infinity norm is defined by

$$\|f\|_\infty = \max_{x\in[a,b]} \left|f(x)\right|,$$

  in a manner similar to vector infinity norm.

- The expression $\mathcal{I}(\mathbf{e}_j)$ represents the interpolant $p(x)$ which is *on* at x_j and *off* elsewhere, *i.e.*,

$$p(x_k) = \delta_{k,j} = \begin{cases} 1, & k = j \\ 0, & k \neq j \end{cases}.$$

  Such interpolants are known as **cardinal functions**.

- The theorem says that the (absolute) condition number is larger than the largest of $\left\|\mathcal{I}(\mathbf{e}_j)\right\|_\infty$, but smaller than the sum of these.

Piecewise Linear Interpolation

Piecewise Linear Interpolation

Assume that $x_1 < x_2 < \cdots < x_n$ are fixed. The function $p(x)$ defined piecewise[2] by

$$p(x) = y_j + \frac{y_{j+1} - y_j}{x_{j+1} - x_j}(x - x_j)\,, \quad \text{for } x \in [x_j, x_{j+1}],\ 1 \leqslant j \leqslant n-1$$

- is linear on each interval $[x_j, x_{j+1}]$;
- connects any two consecutive data points (x_j, y_j) and (x_{j+1}, y_{j+1}) by a straight line.

[2] Note the formula changes depending on which interval x lies in.

Hat Functions

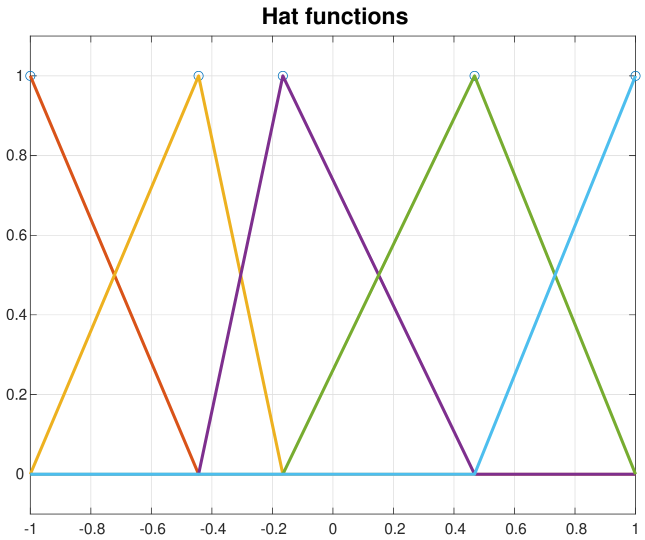
Denote by $H_j(x)$ the jth *piecewise linear* cardinal function:

$$H_j(x) = \begin{cases} \dfrac{x - x_{j-1}}{x_j - x_{j-1}}, & x \in [x_{j-1}, x_j], \\ \dfrac{x_{j+1} - x}{x_{j+1} - x_j}, & x \in [x_j, x_{j+1}], \\ 0, & \text{otherwise}, \end{cases} \qquad j = 1, 2, \ldots, n.$$

- The functions $H_1, \ldots, H_n$ are called **hat functions** or **tent functions**.
- Each H_j is globally continuous and is linear inside each interval $[x_j, x_{j+1}]$

Note: The definitions of $H_1(x)$ and $H_n(x)$ require additional nodes x_0 and x_{n+1} for x outside of $[x_1, x_n]$, which is not relevant in the discussion of interpolation.

Hat Functions (cont')



Hat Functions As Basis

- Any linear combination of hat functions is continuous and is linear inside each interval $[x_j, x_{j+1}]$.
- Conversely, *any* such function is expressible as a unique linear combination of hat functions, *i.e.*,

$$\sum_{j=1}^{n} c_j H_j(x), \quad \text{for some choice of } c_1, \ldots, c_n.$$

- No smaller set of functions has the same properties.

The hat functions form a **basis** of the set of functions that are continuous and piecewise linear relative to $\mathbf{x}$ (the vector of nodes).

Cardinality Conditions

- By construction, the hat functions are cardinal functions for piecewise linear (PL) interpolation, *i.e.*, they satisfy

$$H_j(x_k) = \delta_{j,k}. \qquad \text{(cardinality condition)}$$

- Key consequence of this property is that the piecewise linear interpolant $p(x)$ for the data values in $\mathbf{y}$ is trivially expressed by

$$p(x) = \sum_{j=1}^{n} y_j H_j(x).$$

Recipe for PL Interpolant

Piecewise Linear Interpolant

The piecewise linear polynomial

$$p(x) = \sum_{j=1}^{n} y_j H_j(x)$$

is the unique such function which passes through all the data points.

Proof: It is easy to check the interpolating property:

$$p(x_k) = \sum_{j=1}^{n} y_j H_j(x_k) = \sum_{j=1}^{n} y_j \delta_{j,k} = y_k \quad \text{for every } k \in \mathbb{N}[1,n].$$

To show uniqueness, suppose $\tilde{p}$ is another such function in the form

$$\tilde{p}(x) = \sum_{j=1}^{n} c_j H_j(x)\,.$$

Then $p(x_k) - \tilde{p}(x_k) = 0$ for all $k \in \mathbb{N}[1,n]$. This implies that $c_k = y_k$ for all k $\square$

Conditioning

Lemma

Let $\mathcal{I}$ is the piecewise linear interpolation operator and $\mathbf{z} \in \mathbb{R}^n$. Then

$$\left\|\mathcal{I}(\mathbf{z})\right\|_\infty = \|\mathbf{z}\|_\infty .$$

- It follows from the lemma that <u>the absolute condition number of piecewise linear interpolation in the infinity norm equals one</u>.

Proof of lemma. Let

$$p(x) = \mathcal{I}(\mathbf{z}) = \sum_{j=1}^{n} z_j H_j(x).$$

Let k be the index corresponding to the element of $\mathbf{z}$ with the largest absolute value, that is, $z_k = \|\mathbf{z}\|_\infty$. Since $z_k = p(x_k)$, it follows that $|p(x_k)| = \|\mathbf{z}\|_\infty$ and so $\|p\|_\infty \leqslant \|\mathbf{z}\|_\infty$.

To show the other inequality, note that

$$|p(x)| = \left| \sum_{j=1}^{n} z_j H_j(x) \right| \leqslant \sum_{j=1}^{n} |z_j| \, H_j(x) \leqslant \|\mathbf{z}\|_\infty \sum_{j=1}^{n} H_j(x) = \|\mathbf{z}\|_\infty \, ,$$

where the final step uses the fact[3] that $\sum_{j=1}^{n} H_j(x) = 1$. It implies that $\|p\|_\infty \leqslant \|\mathbf{z}\|_\infty$. Therefore, $\|p\|_\infty = \|\mathbf{z}\|_\infty$. □

[3] This property is called the *partition of unity*. Confirm it!

Convergence: Error Analysis

Set-up for analysis.

- Generate a set of data points using a "nice" function f on an interval containing all nodes, *i.e.*, $y_j = f(x_j)$. (The *niceness* of a function is described in precise terms below.)
- Then perform PL interpolation of the data to obtain the interpolant p.
- **Question.** How close is p to f?

Notation (Space of Differentiable Functions)

Let $C^n[a, b]$ denote the set of all functions that are n-times continuously differentiable on $[a, b]$. That is, if $f \in C^n[a, b]$, then $f^{(n)}$ exists and is continuous on $[a, b]$, where derivatives at the end points are taken to be one-sided derivatives.

Convergence: Error Analysis (cont')

Theorem 2 (Error Theorem for PL Interpolation)

Suppose that $f \in C^2[a,b]$. Let p_n be the piecewise linear interpolant of $(x_j, f(x_j))$ for $j = 1, \ldots, n$, where

$$x_j = a + jh \quad \text{and} \quad h = \frac{b-a}{n-1}.$$

Then

$$\|f - p_n\|_\infty \leqslant \|f''\|_\infty h^2.$$

- The theorem pertains to the interpolation on equispaced nodes.
- The significance of the theorem is that the error in the interpolant is $O(h^2)$ as $h \to 0$. (We say that PL interpolation is *second-order accurate*.)
- **Practical implication:** If n is doubled, the PL interpolant becomes about four times more accurate. A log-log graph (`loglog`) of error against n is a straight line.

Convergence: Error Analysis (cont')