Piecewise Interpolation (Introduction)

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

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

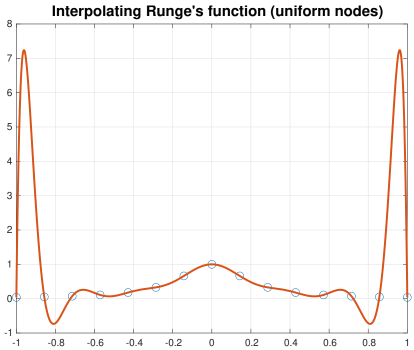
  the matrix V is called the **Vandermonde matrix**; see Lecture 13.

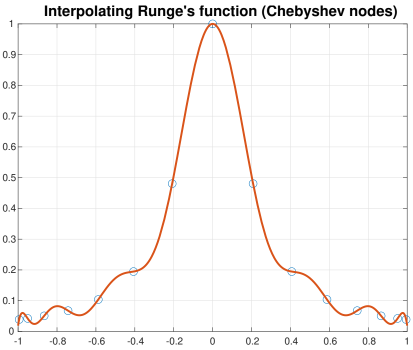
- A polynomial interpolant has severe oscillations as n grows large, unless nodes are special; see illustration in the next slide.

Illustration of 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

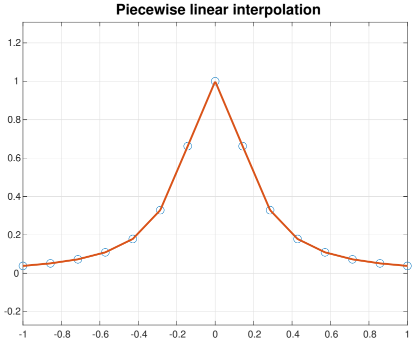
To interpolate data obtained from [1]

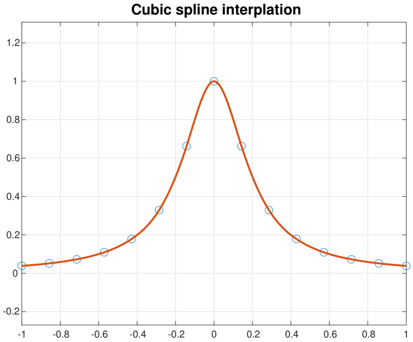
$$f(x) = \frac{1}{1+25x^2}.$$

```
% 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 points)
  - 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.