CSI 401 (Fall 2025)

Numerical Methods

Lecture 14: Interpolation Using Polynomials

Chong Liu

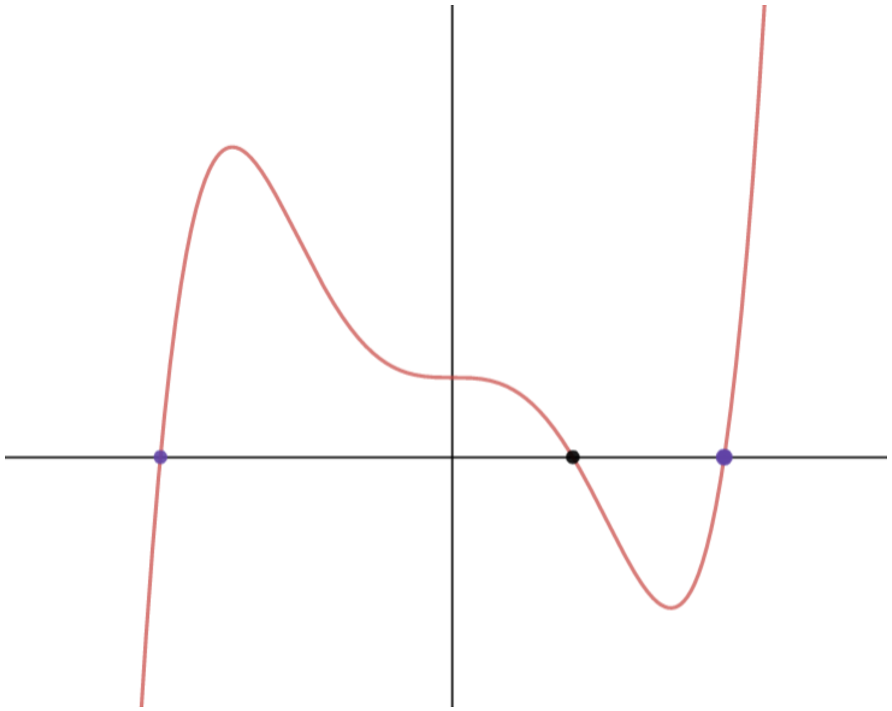
Department of Computer Science

Nov 10, 2025

Recap: Nonlinear equation solver

- Problem statement: find a root of $F(x) = 0$ within an interval $[a, b]$
    - where F is a continuous nonlinear function

    - Discussion: any ideas?



Recap: Bisection method

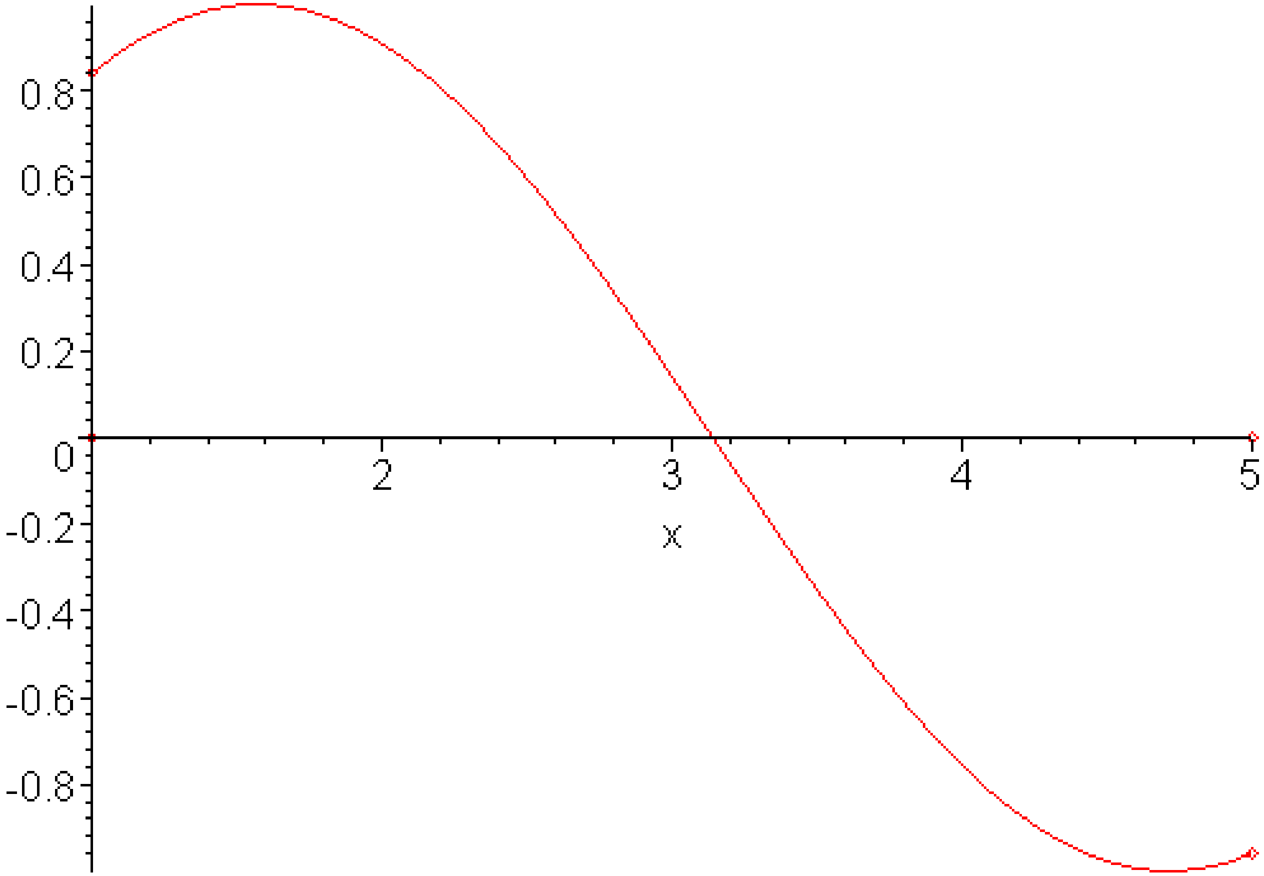
- Key idea:
  - In every iteration, we cut the interval in half while still maintaining the property that the endpoints have opposite signs. This allows us to conclude that we're getting closer and closer to a root.
- Algorithm:

1. Preprocessing: If $F(a) = 0$ or $F(b) = 0$, output whichever one was 0 and terminate. If $F(a) < 0 < F(b)$, then set $inc = 1$. Otherwise, set $inc = 0$.
2. Compute $z = \frac{a+b}{2}$, the midpoint of the interval $[a, b]$.
3. If $F(z) = 0$, return z and terminate.
4. If $inc = 0$ (so $F(a) > 0 > F(b)$):
   (a) If $F(z) < 0$, then set $b = z$.
   (b) If $F(z) > 0$, then set $a = z$.
5. If $inc = 1$ (so $F(a) < 0 < F(b)$):
   (a) If $F(z) < 0$, then set $b = z$.
   (b) If $F(z) > 0$, then set $a = z$.

After k iterations, we output the midpoint of the resulting interval.

Recap: Illustration of the bisection method

- Initial interval: [1, 5]
- 3 steps in each iteration:
  - Given a, b, find midpoint
  - Check midpoint value
  - Update a or b



Recap: Convergence of bisection method

- Key idea in analysis:
  - Let us call the initial interval $[a, b]$. After every iteration, the interval $[a, b]$ is cut in half. Thus, the length of the k-th interval is given by $l_k = \frac{b-a}{2^k}$

- **Theorem:**
  - There exists some root x of F between a and b such that the output m of the algorithm satisfies
    - $|m - x| \leqslant \frac{b-a}{2^{k+1}}$
  - In other words, if we fix some desired accuracy $\epsilon > 0$, then
    - $|m - x| \leqslant \epsilon$
  - provided that $k \geqslant \log_2\left(\frac{b-a}{\epsilon}\right) - 1$

  - Since the absolute error is divided by a positive constant in every iteration, we say that the bisection search converges linearly to the solution.

Recap: Newton’s method

- Key idea:
  - Take F, find its local linear approximation at a starting point x_0, solve for x to get x_1, and use that as our new initial point.
  - Iterate until (hopefully) convergence.
- So, how to find the local linear approximation of F at x_0?
  - First-order Taylor expansion at x_0

$$P_1(x) = F(x_0) + F'(x_0)(x - x_0).$$

$$0 = F(x_0) + F'(x_0)(x - x_0) \implies -\frac{F(x_0)}{F'(x_0)} = x - x_0 \implies x = x_0 - \frac{F(x_0)}{F'(x_0)}.$$

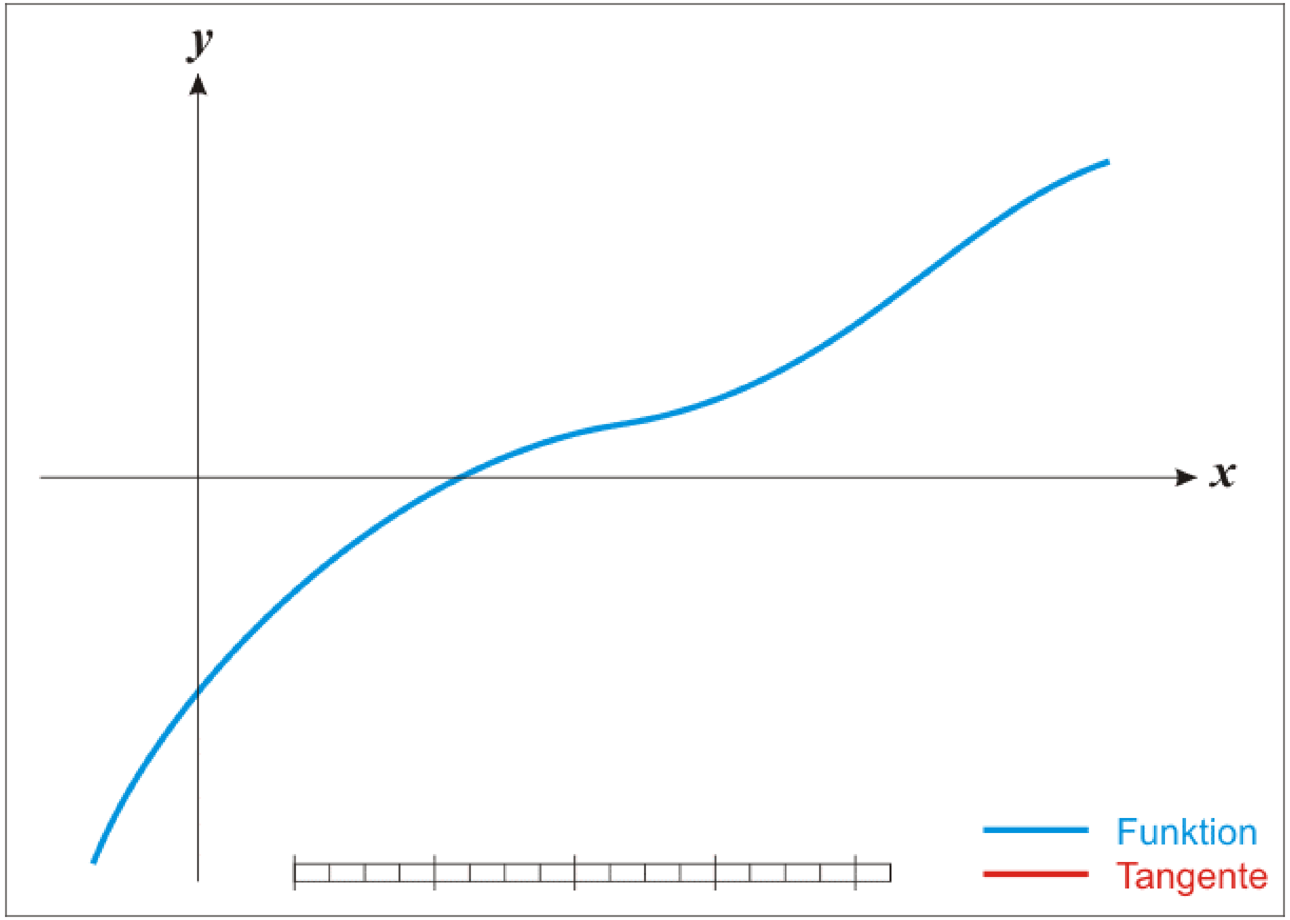
- Algorithm: (Newton update equation)

$$x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)}.$$

Recap: Illustration of the Newton's method

- In each iteration:

$$x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)}.$$



Recap: Example of Newton's method

- Without a calculator, compute $x = \sqrt{2}$
  - Newton's method with 3 iterations
- Solutions:
  - Rewrite the problem as finding the root of $F(x) = x^2 - 2 = 0$
  - Then $F'(x) = 2x$
  - So updating rule is $x_{k+1} = x_k - \frac{x_k^2 - 2}{2x_k} = \frac{x_k}{2} + \frac{1}{x_k}$
    - Suppose $x_0 = 2$
    - $x_1 = 1 + \frac{1}{2} = 1.5$
    - $x_2 = 0.75 + \frac{1}{1.5} = 1.41666667 \ldots$
    - $x_3 = 1.41421569\ldots$
- Check $x_3^2 = 2.00000602$

Recap: Convergence of Newton's method

- Newton's method doesn't always converge to the root!

- **Theorem:**
  - Newton's method converges quadratically if:
    1. $f(x)$ is continuously differentiable near r,
    2. $f'(r) \neq 0$,
    3. The starting value x_0 is sufficiently close to r.
  - If $f'(r) = 0$ or x_0 is far from the root, convergence may be slow, linear, or divergent.

Recap: Summary of nonlinear equation solvers

- Things to know about:
  - Problem statement
  - Assumptions behind each method
  - Benefits/drawbacks of each method
  - Key theorems from calculus that feature in their analysis
  - How does each method look, visually?
  - How do we code each method up in Matlab/Python?
- Technical summary table:

Methods	Bisection method	Newton's method
Assumptions	Continuity, opposite sign condition	Continuous, differentiable, initial point close to root
Associated theorem	Intermediate value theorem	Taylor's remainder theorem
Guarantee	Linear convergence	Quadratic convergence

Agenda

- Problem setup of interpolation

- Different interpolation methods:

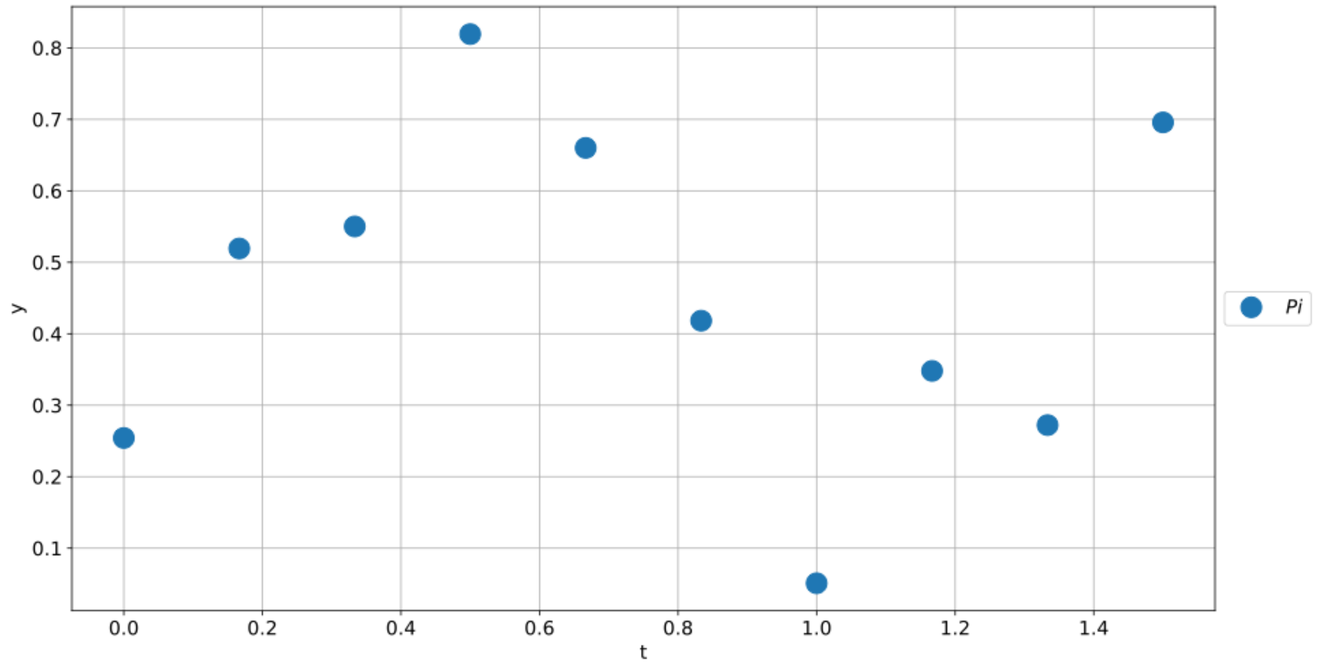
  - Basic polynomial interpolation

  - Lagrange interpolation

Example: Understanding house price change

- You observe the price change of a house in the past 15 months
  - You have 10 data points
  - t denotes time
  - y denotes price

- Discussion:
  - What can you do to study the price trend?
  - How can you define a function that describes all these price points?



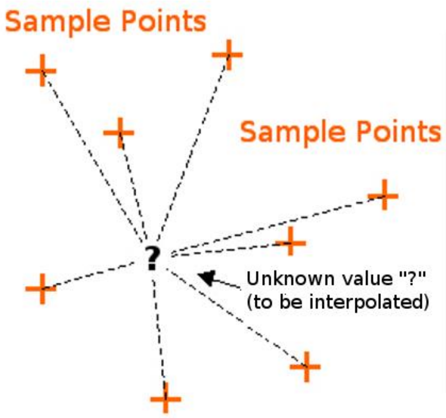
Problem setup of Interpolation

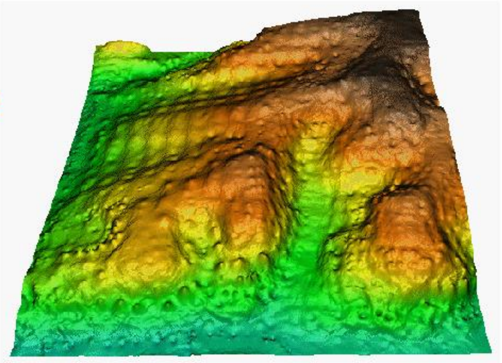
- For given data
  - $(t_1, y_1), (t_2, y_2), \dots, (t_m, y_m)$ with $t_1 < t_2 < \cdots < t_m$
- determine function $f: R \to R$ such that
  - $f(t_i) = y_i, \forall i = 1, \dots, m$
  - Exactly crossing all data points!

- f is **interpolating function**, or **interpolant**, for given data.
  - f could be function of more than one variable, but let's focus on the 1-dimensional case first.

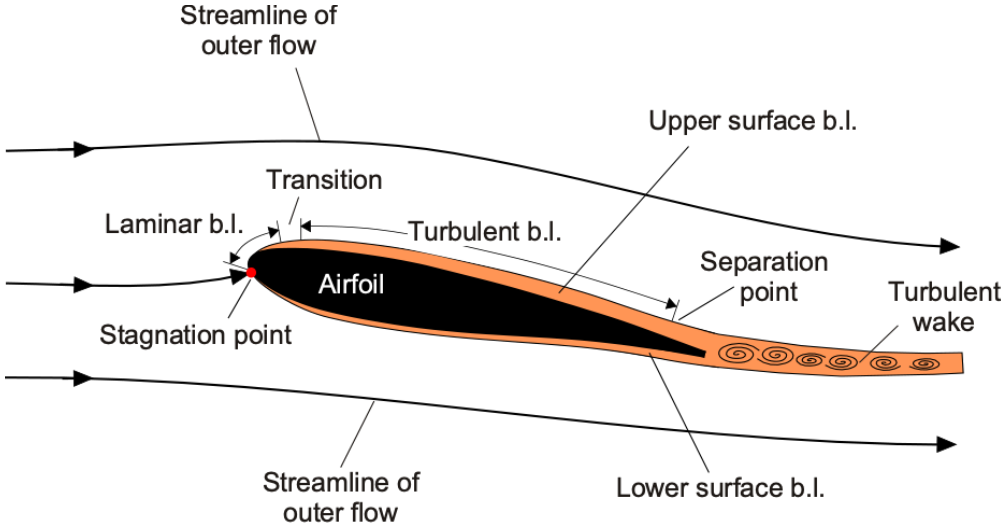
Applications of interpolation

- Geographic information systems
  - Estimating the surface of Mars
    - Limited sample points from a Mars probe
    - How does the unexplored region look like?





- Air dynamics and mechanics
  - A strong wind flowing above and below an aircraft's frame
  - Limited number of sensors



Purposes of interpolation

- Plotting smooth curve through discrete data points
- Quick and easy evaluation of mathematical function
- Replacing complicated function by simple one

Interpolation vs. Regression

- By definition, interpolating function fits given data points exactly
- Interpolation is inappropriate if data points subject to significant errors
  - Regression is a better choice in this case
- It is usually preferable to smooth noisy data
- Regression is more appropriate for special function libraries
  - Linear regression

Basis Functions

- Family of functions for interpolating:
  - Set of basis functions $\phi_1(t), \ldots, \phi_n(t)$
- Interpolating function f is chosen as linear combination of them

$$f(t) = \sum_{j=1}^{n} x_j \phi_j(t)$$

- Requiring f to interpolate data (t_i, y_i) means

$$f(t_i) = \sum_{j=1}^{n} x_j \phi_j(t_i) = y_i, \quad i = 1, \ldots, m$$

  - Discussion: What is this system?
    - A system of linear equations $Ax = y$ for n-vector x of parameters x_j, where entries of $m \times n$ matrix A are given by $a_{ij} = \phi_j(t_i)$.

Existence, Uniqueness, and Conditioning

- Existence and uniqueness of interpolant depend on number of data points m and number of basis functions n
  - If $m > n$, interpolant usually doesn't exist
  - If $m < n$, interpolant is not unique
  - If $m = n$ and data points t_i are distinct, data can be fit exactly

Basic polynomial interpolation

- Simplest and most common type of interpolation using polynomials

- Unique polynomial of degree at most $n - 1$ passes through n data points $(t_i, y_i), i = 1, \ldots, n$, where t_i are distinct

Basic polynomial interpolation

- Basis functions

$$\phi_j(t) = t^{j-1}, \quad j = 1, \ldots, n$$

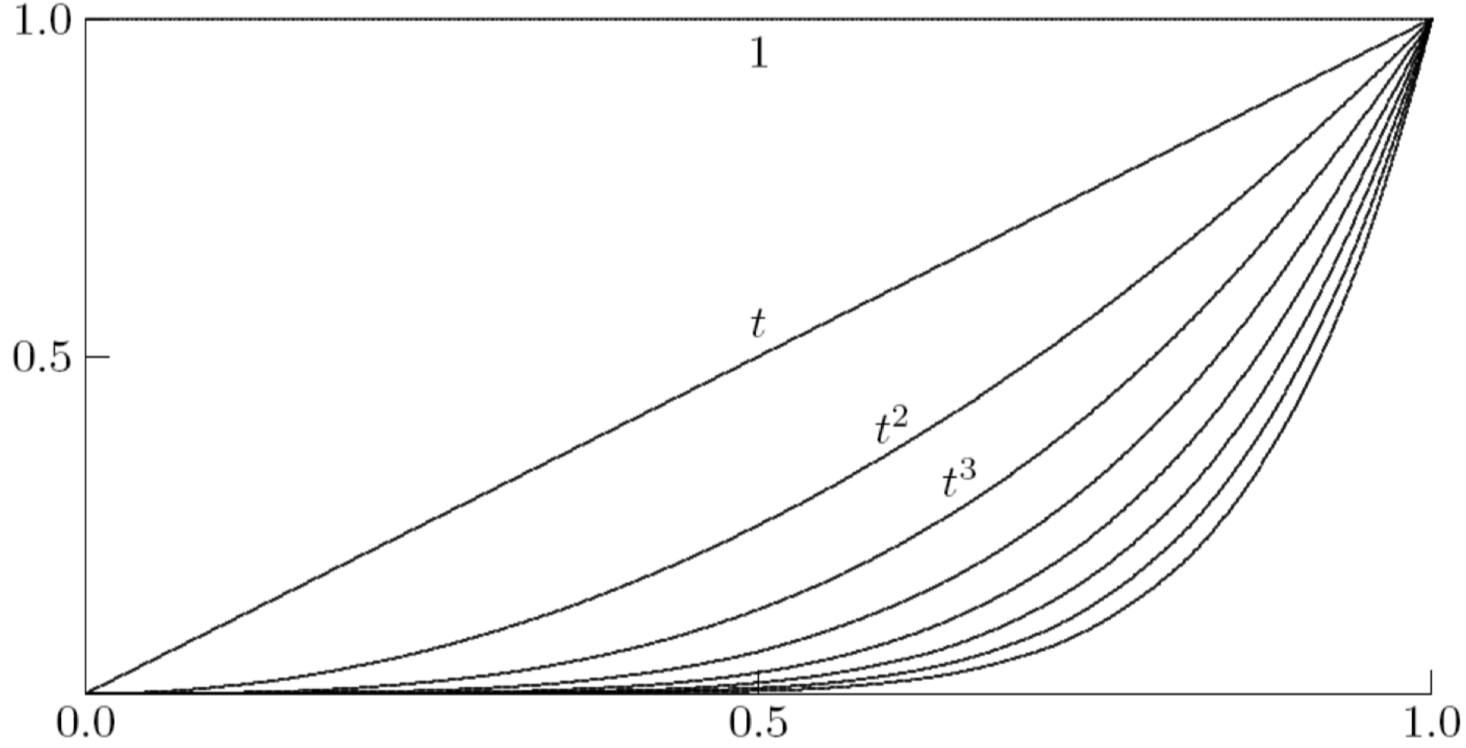
- give interpolating polynomial of form

$$p_{n-1}(t) = x_1 + x_2 t + \cdots + x_n t^{n-1}$$

- with coefficients x given by $n \times n$ linear system

$$\boldsymbol{Ax} = \begin{bmatrix} 1 & t_1 & \cdots & t_1^{n-1} \\ 1 & t_2 & \cdots & t_2^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_n & \cdots & t_n^{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \boldsymbol{y}$$

Basis functions



In-class exercise: Polynomial interpolation

- Determine polynomial of degree two interpolating three data points $(-2,-27),(0,-1),(1,0)$

$$\boldsymbol{Ax} = \begin{bmatrix} 1 & t_1 & \cdots & t_1^{n-1} \\ 1 & t_2 & \cdots & t_2^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_n & \cdots & t_n^{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \boldsymbol{y}$$

- Solution:

$$\boldsymbol{Ax} = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \boldsymbol{y}$$

$$\begin{bmatrix} 1 & -2 & 4 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -27 \\ -1 \\ 0 \end{bmatrix}$$

$$x = \begin{bmatrix} -1 & 5 & -4 \end{bmatrix}^T$$

$$p_2(t) = -1 + 5t - 4t^2$$

Basic polynomial interpolation

- For basis, matrix A is increasingly ill-conditioned as degree increases
  - $m < n$
  - Values of coefficients are poorly determined
- Discussion: Time complexity in n in solving system $Ax = y$ using standard linear equation solver to determine coefficients x?
  - $O(n^3)$
- The amount of computational work required to solve it can be improved by using different basis functions

Lagrange interpolation

- For given set of data points $(t_i, y_i), i = 1, \dots, n$, let

$$\ell(t) = \prod_{k=1}^{n}(t - t_k) = (t - t_1)(t - t_2)\cdots(t - t_n)$$

- Define weights

$$w_j = \frac{1}{\ell'(t_j)} = \frac{1}{\prod_{k=1,\, k\neq j}^{n}(t_j - t_k)}, \quad j = 1, \dots, n$$

- Lagrange basis functions are then given by

$$\ell_j(t) = \ell(t)\frac{w_j}{t - t_j}, \quad j = 1, \dots, n$$

  - From definition, $\ell_j(t)$ is polynomial of degree $n - 1$

Lagrange interpolation

- Assuming common factor $(t_i - t_j)$ in $\ell(t_j)/(t_i - t_j)$ is canceled to avoid division by zero when evaluating $\ell_j(t_i)$, then

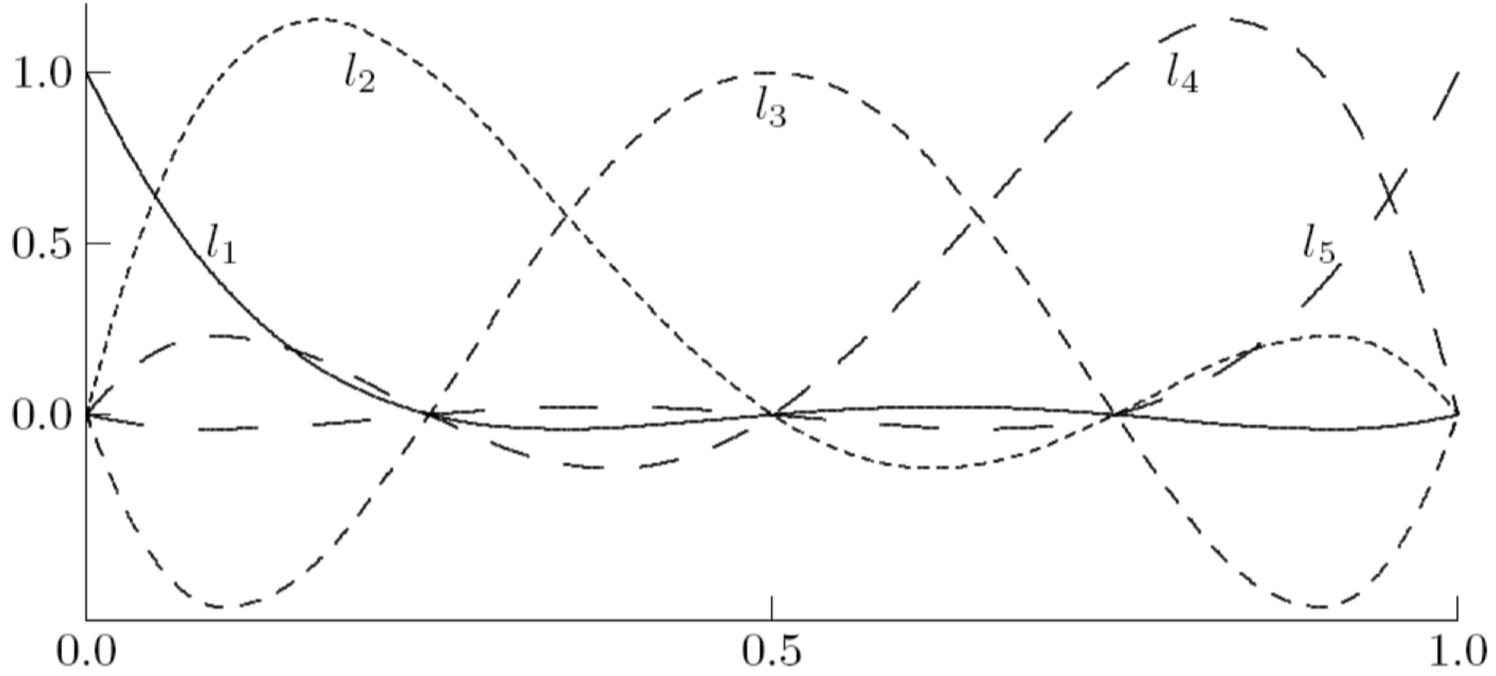
$$\ell_j(t_i) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}, \qquad i, j = 1, \ldots, n$$

  - Matrix of linear system $Ax = y$ is identity matrix I
  - Coefficients x for Lagrange basis functions are just data values y

- Polynomial of degree $n - 1$ interpolating data points $(t_i, y_i), i = 1, \ldots, n$ is given by

$$p_{n-1}(t) = \sum_{j=1}^{n} y_j \ell_j(t) = \sum_{j=1}^{n} y_j \ell(t) \frac{w_j}{t - t_j} = \ell(t) \sum_{j=1}^{n} y_j \frac{w_j}{t - t_j}$$

Lagrange Basis Functions



In-class exercise: Lagrange interpolation

- Use Lagrange interpolation to determine interpolating polynomial for three data points $(-2,-27),(0,-1),(1,0)$

- Solution:

$$\ell(t) = (t - t_1)(t - t_2)(t - t_3) = (t+2)t(t-1)$$

$$\begin{aligned} w_1 &= \frac{1}{(t_1 - t_2)(t_1 - t_3)} = \frac{1}{(-2)(-3)} = \frac{1}{6} \\ w_2 &= \frac{1}{(t_2 - t_1)(t_2 - t_3)} = \frac{1}{2(-1)} = -\frac{1}{2} \\ w_3 &= \frac{1}{(t_3 - t_1)(t_3 - t_2)} = \frac{1}{3 \cdot 1} = \frac{1}{3} \end{aligned}$$

$$p_2(t) = (t+2)t(t-1)\left(-27\frac{1/6}{t+2} - 1\frac{-1/2}{t} + 0\frac{1/3}{t-1}\right)$$

Lagrange interpolation

- Once weights w_j have been computed, which requires $O(n^2)$ operations,
  - then interpolating polynomial can be evaluated for any given argument in $O(n)$ operations

- If new data point (t_{n+1}, y_{n+1}) is added, then interpolating polynomial can be updated in $O(n)$ operations
  - Divide each w_j by $(t_j - t_{n+1}), j = 1, \ldots, n$
  - Compute new weight w_{n+1} using usual formula