Lec 16: Interpolation

15-369/769: Numerical Methods (Foundations, ML & Visual Computing)

Instructor: Minchen Li

Table of Content

- Interpolation in a Single Variable
- Multivariable Interpolation
- Theory of Interpolation

Table of Content

- Interpolation in a Single Variable
- Multivariable Interpolation
- Theory of Interpolation

Interpolation in a Single Variable

Motivation

- So far, we assumed we can evaluate $f(\mathbf{x})$ anywhere.
- In many settings, f is only known at discrete samples:
  - Photographs: pixel grid samples a continuous light field.
  - Machine learning: data collected at limited points.
  - Medicine: drug responses measured at discrete dosages.
- Goal: construct an interpolant $\tilde{f}$ such that

$$\tilde{f}(\mathbf{x}_i) = f(\mathbf{x}_i), \quad \text{and } \tilde{f} \text{ is “reasonable” (smooth, stable).}$$

Interpolation in a Single Variable

Setup

- Given k pairs (x_i, y_i) with $y_i = f(x_i)$.
- We wish to estimate $f(x)$ for $x \notin \{x_1, \ldots, x_k\}$.
- Seek a smooth interpolant $f(x)$ such that

$$f(x_i) = y_i, \quad i = 1, \ldots, k.$$

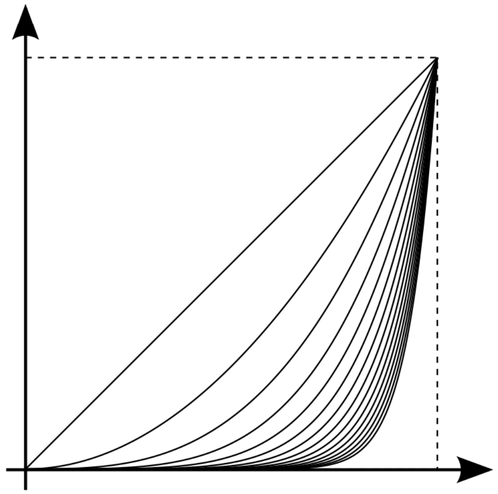
- Approach: express $f(x)$ as a linear combination of basis functions

$$f(x) = \sum_{j=1}^{k} a_j \phi_j(x).$$

- Choose basis $\{\phi_j\}$ and solve for coefficients a_j.

Interpolation in a Single Variable

Polynomial Interpolation via Monomial Basis



- Let $f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k-1}$.
- Imposing $f(x_i) = y_i$ for all i gives:

$$\begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{k-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{k-1} \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & x_k & x_k^2 & \dots & x_k^{k-1} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{k-1} \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{pmatrix}.$$

- This is the **Vandermonde system**.
- **Pros**: direct, explicit; **Cons**: numerically unstable for large k or clustered x_i.

Interpolation in a Single Variable

Alternative Polynomial Bases

- Polynomial space of degree $\leq k-1$ can be expressed in different bases:

$$\text{Monomial: } \{1, x, x^2, \ldots, x^{k-1}\}$$

$$\text{Lagrange: } \{\phi_1(x), \ldots, \phi_k(x)\}$$

$$\text{Newton: } \{\psi_1(x), \ldots, \psi_k(x)\}.$$

- All span the same space $\mathbb{P}_{k-1}$.
- Choice affects conditioning and computational cost.

Interpolation in a Single Variable

Lagrange Basis: Definition

- For sample points $\{x_1, \ldots, x_k\}$ define:

$$\phi_i(x) := \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}.$$

- Properties: $\phi_i(x_j) = \begin{cases} 1, & j = i, \\ 0, & j \neq i. \end{cases}$

- Hence, the interpolant can be written directly as

$$f(x) = \sum_{i=1}^{k} y_i \, \phi_i(x).$$

- Avoids solving a system — coefficients are y_i themselves.

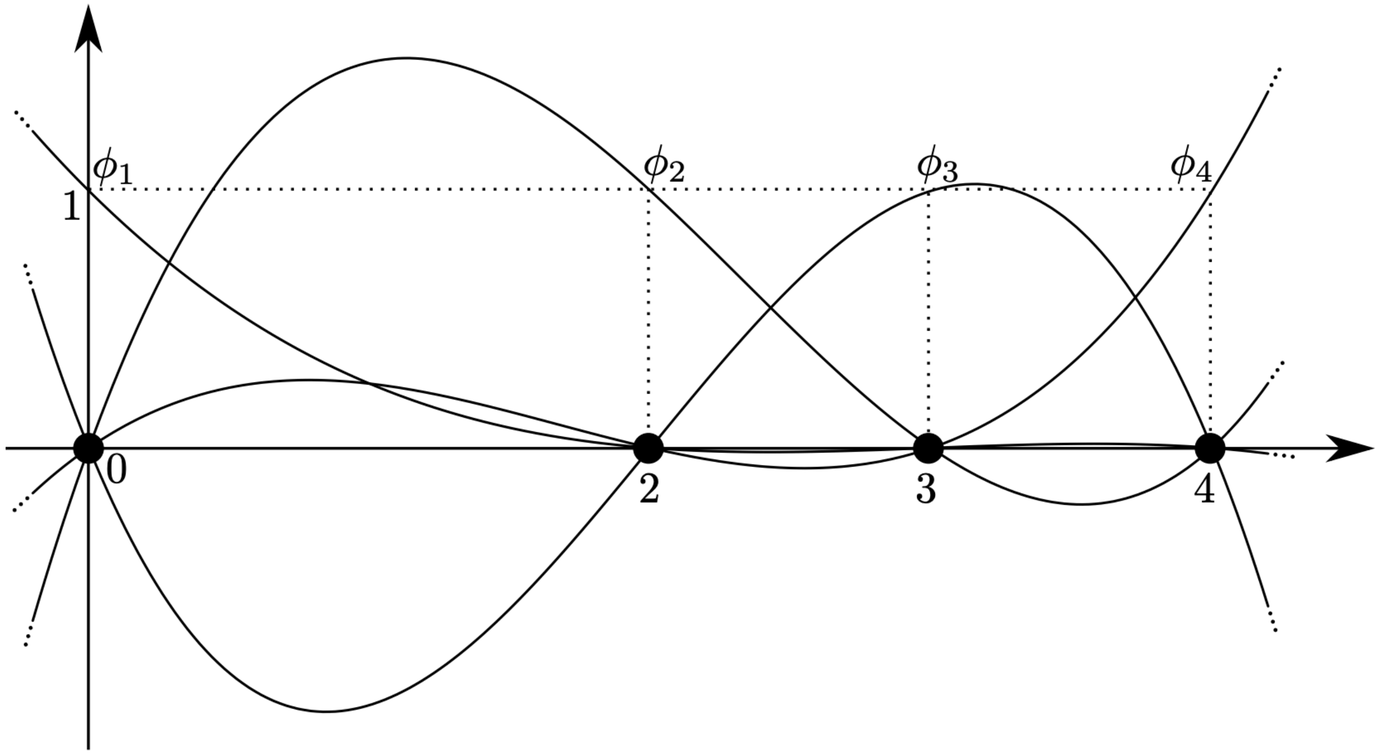
Interpolation in a Single Variable

Lagrange Basis: Example

Example: $x_1 = 0, x_2 = 2, x_3 = 3, x_4 = 4$.

$$\phi_1(x) = \tfrac{1}{24}(-x^3 + 9x^2 - 26x + 24),$$
$$\phi_2(x) = \tfrac{1}{4}(x^3 - 7x^2 + 12x),$$
$$\phi_3(x) = \tfrac{1}{3}(-x^3 + 6x^2 - 8x),$$
$$\phi_4(x) = \tfrac{1}{8}(x^3 - 5x^2 + 6x).$$

$$f(x) = \sum_{i=1}^{4} y_i \, \phi_i(x).$$



- Check: $f(x_j) = y_j$ directly.
- Evaluation cost: $O(k^2)$.
- Denominators may cause instability if x_i's are close.

Interpolation in a Single Variable

Newton Basis: Definition

- Define recursively:

$$\psi_1(x) = 1, \quad \psi_i(x) = \prod_{j=1}^{i-1}(x - x_j).$$

- Represent $f(x)$ as $f(x) = \sum_{i=1}^{k} c_i \psi_i(x)$.

- Enforcing $f(x_i) = y_i$ yields a lower-triangular system:

$$\begin{pmatrix} \psi_1(x_1) & 0 & \dots & 0 \\ \psi_1(x_2) & \psi_2(x_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \psi_1(x_k) & \psi_2(x_k) & \dots & \psi_k(x_k) \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{pmatrix}.$$

- Solve via forward substitution ($O(k^2)$).

Interpolation in a Single Variable

Newton Basis: Example

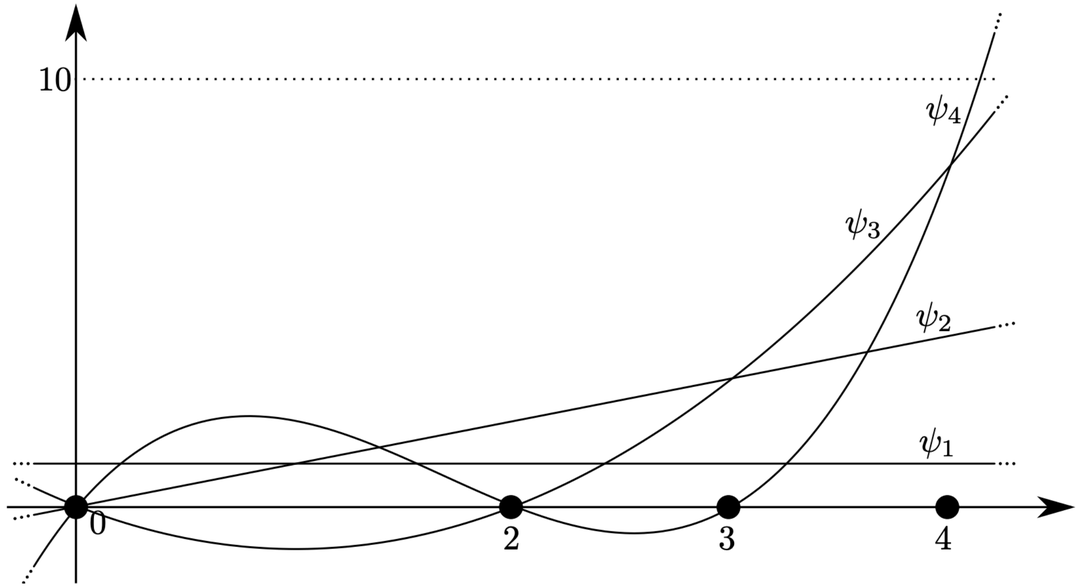
Example: $x_1 = 0,\ x_2 = 2,\ x_3 = 3,\ x_4 = 4.$

$$\psi_1(x) = 1, \quad \psi_2(x) = x, \quad \psi_3(x) = x(x-2) = x^2 - 2x, \quad \psi_4(x) = x(x-2)(x-3) = x^3 - 5x^2 + 6x.$$

$$f(x) = c_1\psi_1(x) + c_2\psi_2(x) + c_3\psi_3(x) + c_4\psi_4(x),$$

and the coefficients c_i satisfy the lower-triangular system above.

- No denominators -> numerically more stable than Lagrange.
- Easy to update if new data points are added.



Interpolation in a Single Variable

Summary: Polynomial Interpolation

Basis	Solve type	Evaluation	Stability
Monomial	Dense $k \times k$ system	$O(k)$	Poor
Lagrange	Explicit sum	$O(k^2)$	Moderate
Newton	Triangular system	$O(k^2)$	Better

- All yield the same interpolating polynomial of degree $k-1$.
- Differ only in numerical conditioning and computational cost.

Interpolation in a Single Variable

Alternative Bases for Interpolation

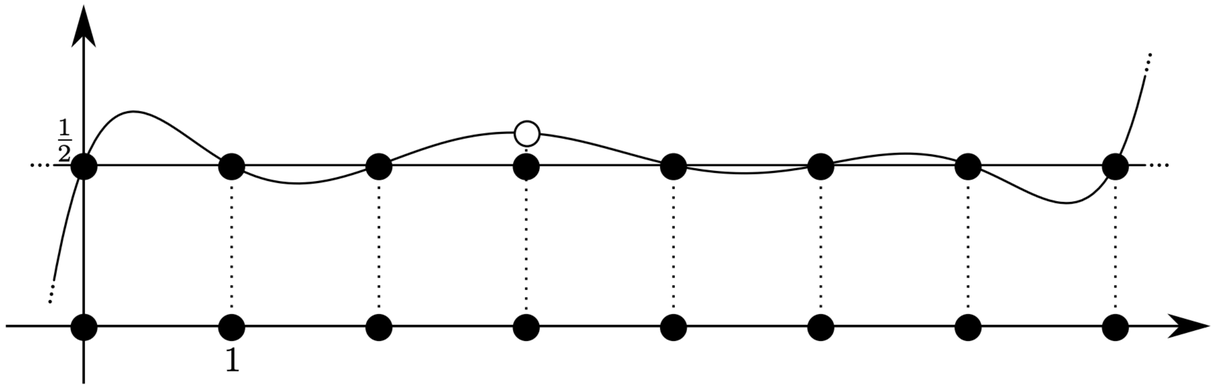
- So far, we have used *polynomial* bases, but other bases are possible.
- Fourier analysis shows that any periodic function can be approximated as a combination of $\cos(kx)$ and $\sin(kx)$.
- A Vandermonde-like matrix can be formed for trigonometric bases; the Fast Fourier Transform efficiently solves the resulting system.
- Extending beyond polynomials, one may use **rational functions**:

$$f(x) = \frac{p_0 + p_1 x + \cdots + p_m x^m}{q_0 + q_1 x + \cdots + q_n x^n}. \quad (p_i, q_i \in \mathbb{Z}\ \forall i)$$

- This allows asymptotic and singular behavior (e.g., poles), unlike polynomials, but may lead to degeneracies or ill-conditioning.

Interpolation in a Single Variable

Piecewise Interpolation Motivation



- Global polynomial interpolation is *nonlocal*: changing one y_i affects $f(x)$ everywhere.
- Desirable property: local influence — $f(x)$ near x_i depends mainly on nearby data.
- The **Weierstrass Approximation Theorem** ensures any smooth function on an interval can be approximated by polynomials, but in practice, stable interpolation requires many local pieces.
- Solution: use **piecewise interpolation** — different polynomial per interval.

Interpolation in a Single Variable

Compact Support and Local Bases

- **Definition: Compact Support.** A function $g(x)$ has **compact support** if there exists $C \in \mathbb{R}$ such that
$$g(x) = 0 \quad \text{for all } |x| > C.$$
- Compactly supported basis functions are nonzero only near specific x_i.
- Leads to **local interpolation** — efficient and stable.
- Commonly used in computer graphics and simulation (e.g., splines, finite elements).

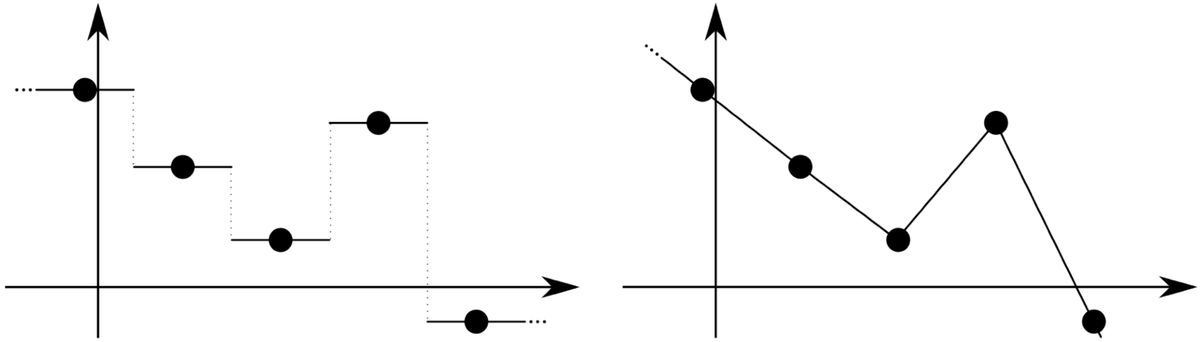
Interpolation in a Single Variable

Piecewise Polynomial Interpolation

- Divide domain $\mathbb{R}$ into intervals $[x_i, x_{i+1}]$.
- Define a local polynomial per interval.
- Two simple cases:
  - **Piecewise constant:** for each x, find nearest x_i, set $f(x) = y_i$.
  - **Piecewise linear:**

$$f(x) = y_i \left(1 - \frac{x - x_i}{x_{i+1} - x_i}\right) + y_{i+1} \frac{x - x_i}{x_{i+1} - x_i}, \quad x \in [x_i, x_{i+1}].$$

- Piecewise constant $\Rightarrow$ discontinuous; piecewise linear $\Rightarrow$ C^0.



Piecewise constant

Piecewise linear

Interpolation in a Single Variable

Increasing Smoothness

- Piecewise quadratics can be C^1, cubics can be C^2, etc.
- Higher continuity $\Rightarrow$ stronger smoothness assumptions on f.
- Not always desirable: e.g., discontinuous physical phenomena (shock waves).
- Key idea: balance continuity and locality for stability and realism.

Interpolation in a Single Variable

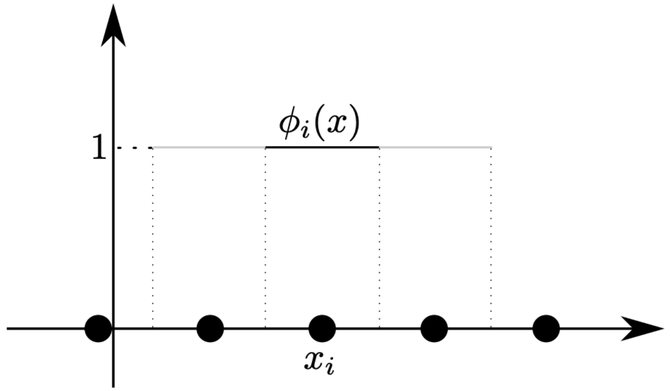
Piecewise Basis Functions

Piecewise constant basis:

$$\phi_i(x) = \begin{cases} 1, & \frac{x_{i-1}+x_i}{2} \le x < \frac{x_i+x_{i+1}}{2}, \\ 0, & \text{otherwise.} \end{cases}$$

Then

$$f(x) = \sum_i y_i \, \phi_i(x).$$



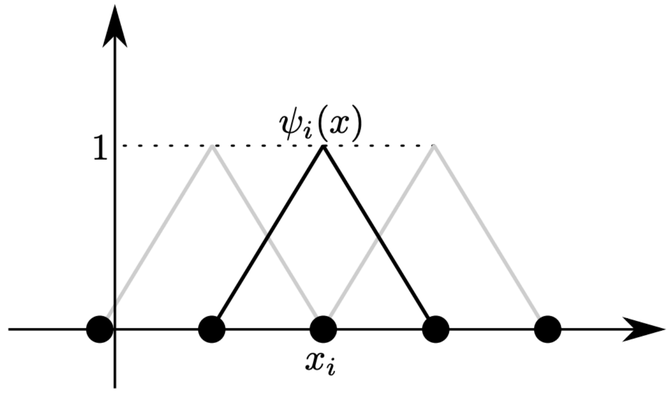
Piecewise constant basis

Piecewise linear ("hat") basis:

$$\psi_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{x_i - x_{i-1}}, & x_{i-1} < x \le x_i, \\ \dfrac{x_{i+1} - x}{x_{i+1} - x_i}, & x_i < x \le x_{i+1}, \\ 0, & \text{otherwise.} \end{cases}$$

Then

$$f(x) = \sum_i y_i \, \psi_i(x).$$



Piecewise linear basis (hat function)

Interpolation in a Single Variable

Summary: Interpolation Families

Type	Locality	Continuity	Notes
Polynomial (global)	Global	C^∞	Nonlocal, unstable for large k
Rational	Global	Variable	Handles poles, can degenerate
Piecewise Constant	Local	C^{-1}	Fast, discontinuous
Piecewise Linear	Local	C^0	Stable, simple
Splines (cubic, etc.)	Local	C^1, C^2	Smooth, widely used

Table of Content

- Interpolation in a Single Variable
- Multivariable Interpolation
- Theory of Interpolation

Multivariable Interpolation

Overview

- Extend interpolation to data pairs $(\mathbf{x}_i, y_i)$ with $\mathbf{x}_i \in \mathbb{R}^n$.
- Challenge: partitioning $\mathbb{R}^n$ into regions around the data points is less obvious than in $\mathbb{R}$.
- Interpolation strategies for higher dimensions often balance:
  - Simplicity and locality (e.g., nearest-neighbor)
  - Smoothness and differentiability (e.g., barycentric, grid-based)

Multivariable Interpolation

Nearest-Neighbor Interpolation

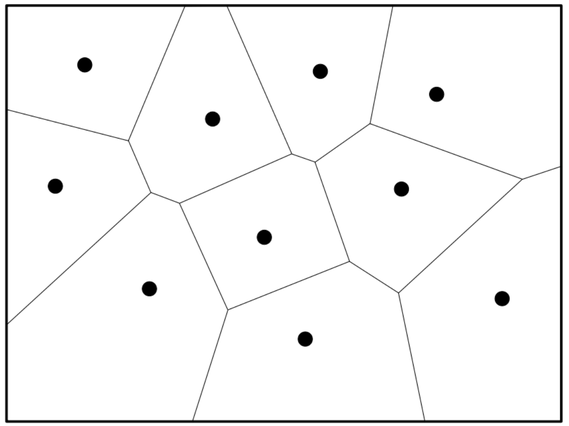
- A **low-order** interpolation method: assign $f(\mathbf{x}) = y_i$ of the nearest data point $\mathbf{x}_i$.

$$f(\mathbf{x}) = y_i \quad \text{where } i = \arg\min_j \|\mathbf{x} - \mathbf{x}_j\|_2.$$

- This yields a piecewise-constant function over $\mathbb{R}^n$.
- Computationally efficient using spatial data structures (e.g., k-d trees).
- **Definition 13.2 (Voronoi Cell)**: Given $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\} \subset \mathbb{R}^n$, the Voronoi cell of $\mathbf{x}_i$ is

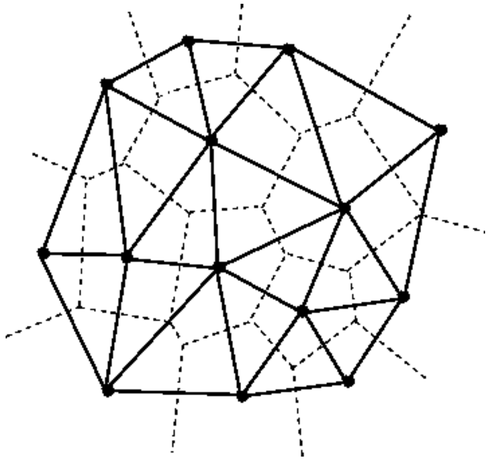
$$V_i = \{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_i\|_2 \leq \|\mathbf{x} - \mathbf{x}_j\|_2 \text{ for all } j \neq i\}.$$

  Each V_i contains all points closer to $\mathbf{x}_i$ than any other data point.



Multivariable Interpolation

Voronoi Cells and Extensions



- In $\mathbb{R}^2$, Voronoi cells are convex polygons forming a partition of space.
- The adjacency of Voronoi cells defines the **Delaunay triangulation**.
- For smoother interpolation, we can average over multiple nearest neighbors:

$$f(\mathbf{x}) = \frac{\sum_{j=1}^{k} w_j(\mathbf{x}) y_j}{\sum_{j=1}^{k} w_j(\mathbf{x})}, \quad w_j(\mathbf{x}) = \frac{1}{\|\mathbf{x} - \mathbf{x}_j\|_2^p}.$$

- This is called *k*-**nearest neighbor interpolation** with distance-based weights.

Image source: http://www.ae.metu.edu.tr/tuncer/ae546/prj/delaunay/

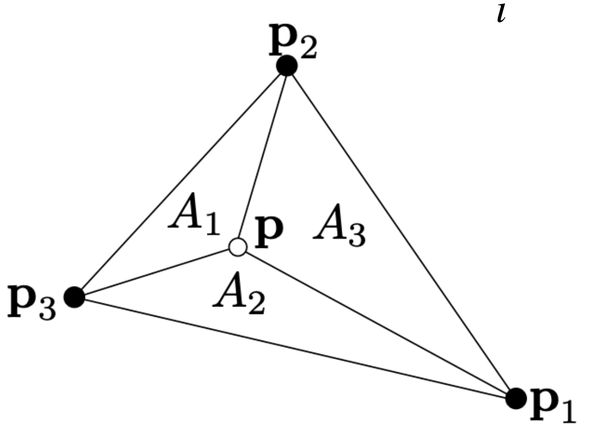
Multivariable Interpolation

Barycentric Interpolation

- Frequently used in computer graphics.
- Suppose we have $n+1$ points $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{n+1}, y_{n+1})$ with $\mathbf{x}_i \in \mathbb{R}^n$.
- Any $\mathbf{x} \in \mathbb{R}^n$ inside their convex hull can be written as a **barycentric combination**:

$$\mathbf{x} = \sum_{i=1}^{n+1} a_i \mathbf{x}_i, \quad \text{with} \quad \sum_i a_i = 1.$$

In 2D, $a_i(\mathbf{x}) = A_i(\mathbf{x}) / \sum_i A_i$:



- Define

$$f(\mathbf{x}) = \sum_i a_i(\mathbf{x}) y_i.$$

- This interpolation is **affine** (linear in $\mathbf{x}$) and exact at the data points.

Multivariable Interpolation

Solving for Barycentric Coordinates

- Given $\mathbf{x}_1, \ldots, \mathbf{x}_{n+1}$, the coefficients $a_i(\mathbf{x})$ satisfy: *(Need to solve for a_i at any $\mathbf{x}$ to be interpolated)*

$$\sum_i a_i \mathbf{x}_i = \mathbf{x}, \quad \sum_i a_i = 1.$$

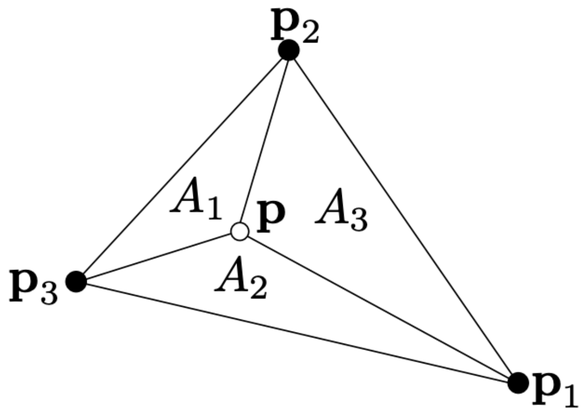
- The system has $n+1$ equations and $n+1$ unknowns, invertible unless the $\mathbf{x}_i$ are degenerate (collinear, coplanar, etc.).

- If more than $n+1$ data points are given, the system becomes **underdetermined**.
- Non-uniqueness allows additional criteria:
  - Smoothness or nonnegativity of $a_i(\mathbf{x})$
  - Optimization-based generalized barycentric coordinates

Multivariable Interpolation

Piecewise Barycentric Interpolation

- When $n+1$-point sets form a triangulation of $\mathbb{R}^2$, we interpolate per triangle.
- Within each triangle, $f(\mathbf{x})$ is affine, determined by the barycentric coordinates of $\mathbf{x}$.
- Implementation parallels piecewise-linear interpolation in 1D.

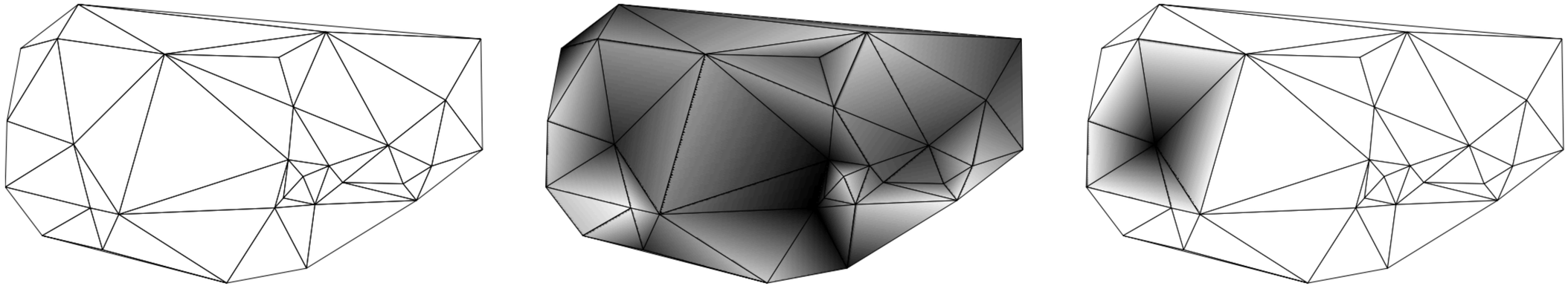


$$a_i(\mathbf{x}) = A_i(\mathbf{x}) / \sum_i A_i:$$

Multivariable Interpolation

Piecewise Barycentric Interpolation Example: Shading

- Per-vertex color interpolation in triangle meshes uses barycentric coordinates.
- Each pixel's color within a triangle is computed as a weighted combination of vertex colors.



(a) Triangle mesh (b) Barycentric interpolation (c) Hat function

Multivariable Interpolation

Grid-Based Interpolation

- When $\mathbf{x}_i$ form a regular grid, interpolation is simplified.
- Grid-based schemes apply 1D interpolation along each coordinate successively.
- Common in imaging and simulation:
  - **Image Processing:** Interpolation used to rotate or scale digital images on $\mathbb{Z}^2$.
  - **MRI:** Voxel grid values $f : \mathbb{R}^3 \to \mathbb{R}$ interpolated to extract surfaces via level sets.

Multivariable Interpolation

Bilinear Interpolation Example

Given $f(0,0) = 1, \quad f(0,1) = -3, \quad f(1,0) = 5, \quad f(1,1) = -11$. We wish to find $f\left(\frac{1}{4}, \frac{1}{2}\right)$.

1. Interpolate in x_1:

$$f\left(\frac{1}{4}, 0\right) = \frac{3}{4} f(0,0) + \frac{1}{4} f(1,0) = 2, \quad f\left(\frac{1}{4}, 1\right) = \frac{3}{4} f(0,1) + \frac{1}{4} f(1,1) = -5.$$

2. Interpolate in x_2:

$$f\left(\frac{1}{4}, \frac{1}{2}\right) = \frac{1}{2} f\left(\frac{1}{4}, 0\right) + \frac{1}{2} f\left(\frac{1}{4}, 1\right) = -\frac{3}{2}.$$

- Result is independent of interpolation order in x_1 and x_2.
- Higher-order schemes (bicubic, Lanczos) use more neighboring samples but are costlier.

Table of Content

- Interpolation in a Single Variable
- Multivariable Interpolation
- Theory of Interpolation

Theory of Interpolation

Overview

- Up to now, our interpolation methods have been heuristic.
- Theoretical analysis clarifies relationships between different interpolation bases.
- A key perspective: interpolation resides in a **linear space of functions** with notions of span, norm, and inner product.

Theory of Interpolation

Linear Algebra of Functions

- Analogous to vectors in $\mathbb{R}^n$, we can treat functions $f : \mathbb{R} \to \mathbb{R}$ as elements of a vector space.
- Define an **inner product** over an interval $[a, b]$:

$$\langle f, g \rangle = \int_a^b f(x)g(x)\,dx.$$

- This induces the **norm**:

$$\|f\|_2 = \sqrt{\langle f, f \rangle}.$$

- These constructions mirror the Euclidean dot product $\mathbf{x} \cdot \mathbf{y}$, but generalized via integration.

Theory of Interpolation

Example: Functional Inner Product of Monomials

Let $p_n(x) = x^n$ on $[0, 1]$. Then

$$\langle p_n, p_m \rangle = \int_0^1 x^{n+m}\, dx = \frac{1}{n+m+1}.$$

Thus, normalized correlation between p_n and p_m is

$$\frac{\langle p_n, p_m \rangle}{\|p_n\|_2 \|p_m\|_2} = \frac{\sqrt{(2n+1)(2m+1)}}{n+m+1}.$$

- When $n \approx m$, this ratio ≈ 1 — the monomials overlap significantly.
- Monomials are therefore **not orthogonal** on $[0, 1]$.

Theory of Interpolation

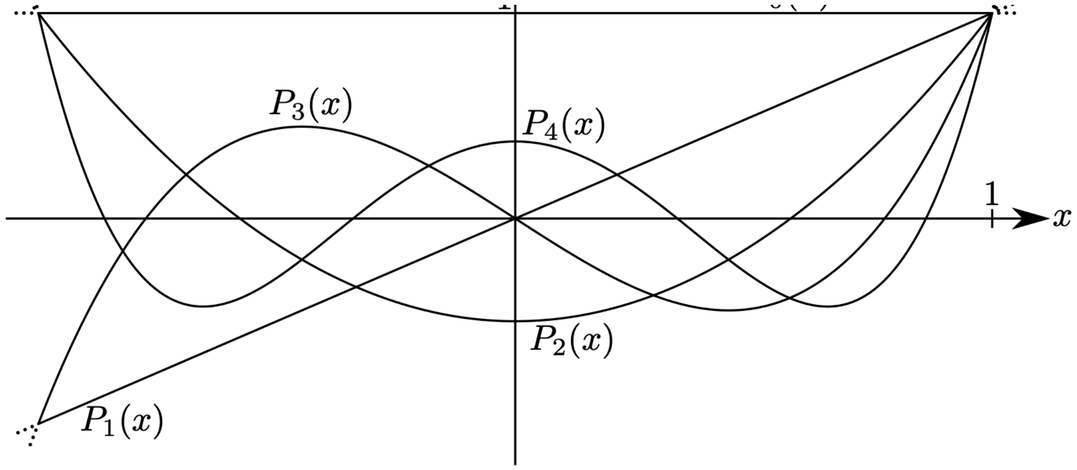
Orthogonal Polynomial Bases: Legendre Polynomials

- Apply the **Gram-Schmidt** process to monomials on $[-1, 1]$ under the inner product

$$\langle f, g \rangle = \int_{-1}^{1} f(x)g(x)\,dx.$$

- This yields the **Legendre polynomials**:

$$\begin{aligned}
P_0(x) &= 1, \\
P_1(x) &= x, \\
P_2(x) &= \tfrac{1}{2}(3x^2 - 1), \\
P_3(x) &= \tfrac{1}{2}(5x^3 - 3x), \\
P_4(x) &= \tfrac{1}{8}(35x^4 - 30x^2 + 3).
\end{aligned}$$



- They form an orthogonal basis for polynomials on $[-1, 1]$.

Theory of Interpolation

Least-Squares Approximation using Orthogonal Bases

- Approximate a function $f(x)$ by a sum: (using Legendre polynomials $P_i(x)$)

$$f(x) \approx \sum_i a_i P_i(x).$$

- The coefficients are found via orthogonal projection:

$$a_i = \frac{\langle f, P_i \rangle}{\langle P_i, P_i \rangle}.$$

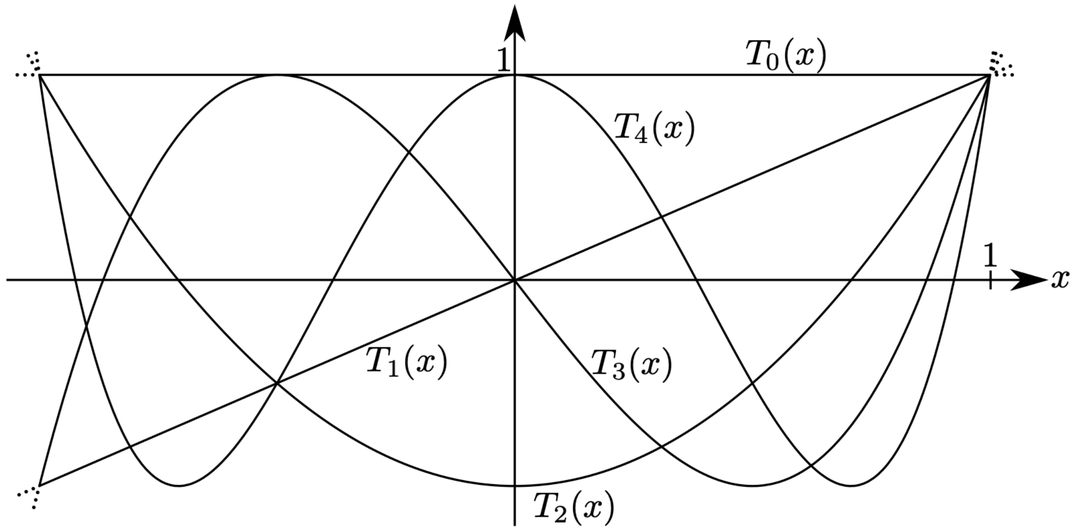
- This minimizes $\|f - \sum_i a_i P_i\|_2$, i.e., a **least-squares approximation**.
- These techniques extend naturally to numerical integration and spectral methods.

Theory of Interpolation

Weighted Inner Products and Chebyshev Polynomials

- Introduce a weighting function $w(x) > 0$: $\langle f, g \rangle_w = \int_a^b w(x) f(x) g(x)\, dx$.
- For $w(x) = \frac{1}{\sqrt{1-x^2}}$ on $[-1, 1]$, Gram–Schmidt produces the **Chebyshev polynomials**:

$$
\begin{aligned}
T_0(x) &= 1, \\
T_1(x) &= x, \\
T_2(x) &= 2x^2 - 1, \\
T_3(x) &= 4x^3 - 3x, \\
T_4(x) &= 8x^4 - 8x^2 + 1.
\end{aligned}
$$



- They satisfy the identity $T_k(x) = \cos(k \arccos x)$.
- Recurrence relation: $T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x)$.

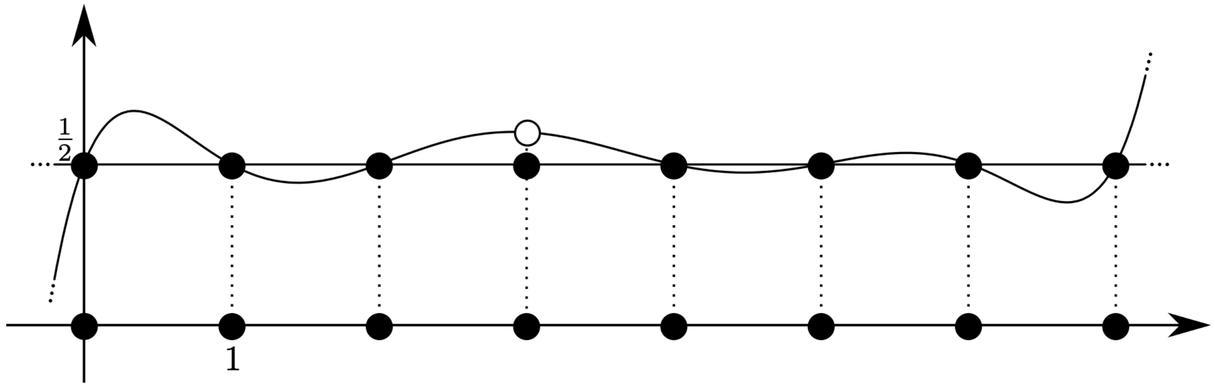
Theory of Interpolation

Chebyshev Properties and Interpolation Points

- $T_k(x)$ oscillates between $+1$ and -1.
- The extrema occur at **Chebyshev points**:

$$x_i = \cos\left(\frac{i\pi}{k}\right), \quad i = 0, 1, \ldots, k.$$

- Sampling near these points mitigates the oscillations (Runge phenomenon) seen in uniform polynomial interpolation.



Theory of Interpolation

Approximation via Piecewise Polynomials

- Consider approximating $f(x)$ on $[a, b]$ by piecewise polynomials.
- Let $\Delta x = b - a$ be the spacing between sample points.
- Approximation error depends on both Δx and the degree of the local polynomial.
- **Piecewise Constant Approximation**: If $f(x)$ is approximated by constant $c = f\left(\frac{a+b}{2}\right)$, then assuming $|f'(x)| \leq M$:

$$\max_{x \in [a,b]} |f(x) - c| \leq M\Delta x. \quad \text{(by the mean value theorem)}$$

  Hence, error scales as $O(\Delta x)$.

Theory of Interpolation

Piecewise Linear Approximation Error

Define

$$\tilde{f}(x) = \frac{b-x}{b-a} f(a) + \frac{x-a}{b-a} f(b).$$

Using Taylor expansions of $f(a)$ and $f(b)$ around x:

$$f(a) = f(x) + (a-x)f'(x) + \tfrac{1}{2}(a-x)^2 f''(x) + O(\Delta x^3),$$
$$f(b) = f(x) + (b-x)f'(x) + \tfrac{1}{2}(b-x)^2 f''(x) + O(\Delta x^3),$$

we find

$$\tilde{f}(x) = f(x) + \tfrac{1}{2}(x-a)(x-b)f''(x) + O(\Delta x^3).$$

- The leading error term $\sim O(\Delta x^2)$.
- Therefore, piecewise linear interpolation yields second-order accuracy **per interval**.

Theory of Interpolation

General Polynomial Approximation Error

- Extending this reasoning, degree-n interpolation gives error $O(\Delta x^{n+1})$ per interval.
- If f is sampled at $n+1$ points $x_0 < x_1 < \cdots < x_n$, the error satisfies

$$|f(x) - p_n(x)| \leq \frac{1}{(n+1)!} \left[\max_{x \in [x_0, x_n]} |f^{(n+1)}(x)| \right] \prod_k |x - x_k|.$$

- Higher-degree interpolants provide faster convergence *if* f is sufficiently smooth.

Table of Content

- Interpolation in a Single Variable
- Multivariable Interpolation
- Theory of Interpolation